Кильдишов В.Д.

«СОЛОН»

Использование приложения MS Excel для моделирования различных задач

Практическое пособие

- решаем разнообразные задачи
- программируем
- создаем динамические модели
- задачи защиты информации
- применение в физике
- используйте возможности MS Excel на 100%

Библиотека студента

Оценка «Отлично»!

ISBN 978-5-91359-145-6

Кильдишов В. Д.

Использование приложения
MS Excel
для моделирования различных задач

Москва
СОЛОН-Пресс
2015

УДК 681.3.06
ББК 32.973.26-018.2
К 39

Кильдишов В.Д.

К39 Использование приложения MS Excel для моделирования различных задач. — М.: СОЛОН-Пресс, 2015. —156 с.: ил.

ISBN 978-5-91359-145-6

Книга является практическим руководством по моделированию задач с использованием приложения MS Excel.

Книга предназначена для школьников, студентов и преподавателей, которые хотят быть знакомы с MS Excel «на Вы», а также научиться моделированию, разработке моделей, алгоритмов и программ.

Прочитав эту книгу, Вы научитесь использовать возможности MS Excel, о которых ранее не знали.

ББК 32.973.26-018.2

Сайт издательства «Ремонт и Сервис 21»: www.remserv.ru
Сайт издательства «СОЛОН-ПРЕСС»: www.solon-press.ru

ISBN 978-5-91359-145-6

Содержание

Предисловие

В книге раскрыты возможности MS Excel, с которые обычно не рассматриваются при традиционном изучении приложения. Эти возможности позволяют создавать динамические модели и иллюстрировать решение задач. Для наглядного изображения процесса моделирования и иллюстрации задач рассмотрены практически все возможные способы работы с элементами диаграмм.

В книге представлена подборка решения задач и моделей из различных разделов естественных дисциплин. Надеемся, что это поможет более глубоко изучить возможности MS Excel, приобрести навыки моделирования, а также отойти от классического построения графиков, расчетов в таблицах, форматирования результатов вычислений. Разработка моделей значительно расширяет перечень применяемых функций, которые обычно ограничиваются основными математическими, логическими и статистическими функциями.

Следует отметить, что физика формул, применяемых в моделях, не раскрывается, а даются определенные пояснения. Надеемся, что пользователи владеют нужными знаниями и смогут самостоятельно познать необходимое через Интернет для уяснения сущности решения задач и моделей.

При создании моделей обучаемые овладевают навыками разработки алгоритмов и основами программирования. Для автоматизации процесса моделирования и из-за имеющихся ограничений при применении стандартных функций в рассматриваемых примерах используются управляющие элементы и соответствующие процедуры VBA. В ряде моделей создаются пользовательские функции и формы. Все это позволит в дальнейшем легко освоить программирование на различных языках. В этом и заключается цель книги.

Отдельные модели и примеры задач были опробованы на занятиях в Западно-Подмосковном институте туризма и в Одинцовском гуманитарном университете. При их рассмотрении они вызывали неподдельный интерес у студентов.

1. О моделировании, решении задач, программировании и MS Excel

1.1. О решении задач и моделировании

В жизни все постоянно решают задачи и моделируют. Решение задач связано с конкретными исходными данными и условиями. В результате решения задачи получаем конкретные данные. При моделировании исходные данные и условия обычно изменяются в определенных диапазонах, а в результате моделирования получаем данные, которые располагаются в определенных диапазонах, зависящих от различных начальных условий. В заключении при моделировании необходимо принимать конкретное решение, основанное на изучении полученных данных и оценки эффективности различных вариантов модели.

Вспомним о классических подходах в разъяснениях понятий о решении задач и моделировании.

Решение задачи - выполнение действий или мыслительных операций, направленных на достижение цели при заданной проблемной ситуации. Данное понятие имеет очень широкий смысл. Нам желательно его сузить и конкретизировать.

Существуют различные методы решения задач, в которые включен и метод моделирования. Круг кажется, замкнулся. Чтобы все-таки его разорвать остановимся на решении математических задачах, математическом и компьютерном моделировании. Общим для них является то, что они основываются на применении определенного языка, символов, объектов и методов математики.

Теперь сделаем еще один шаг для конкретизации. Остановимся на рассмотрении задач и моделей, которые обычно изучаются в учебных заведениях на уроках по естественным дисциплинам (математики, физики, информатики и т.п.). Есть задача о выходе путника из пункта А. Необходимо вычислить время, когда путник прибудет в пункт В при заданных условиях. Это есть классический пример решения математической задачи, когда в результате решения получаем время прибытия путника в пункт В. Но если требуется построить график движения путника, то уместно говорить о модели перемещения путника из пункта А в В. Здесь нужно учесть возможную скорость передвижения путника как свойство объекта. То есть

имеем элементарную модель перемещения путника в пространстве. Если знаний о скорости недостаточно, то можно ввести в модель дополнительные данные о качестве дороги, остановках на отдых, средствах и способах передвижения, запасах топлива и т.п. Это позволит усложнить модель и при этом уточнить точность принятия решения при анализе полученных дополнительных данных. Но не всегда в модели можно учесть все данные. Одни данные неизвестны, другие не нужны, третьи – нельзя учесть в модели из-за различных математических или технологических трудностей.

Таким образом, при моделировании учитываются существенные свойства и параметры объектов (явлений природы, событии и т.п.). Выбор свойств объектов определяется поставленными целями исследования или возможностями их учета в модели. Свойства могут быть внешними и внутренними по отношению объекта. Внешние свойства связаны с окружающей средой, а внутренние – с сущностью объекта. При этом все свойства могут изменяться при моделировании. Также свойства делятся на входные и выходные. Обычно выходные свойства зависят от входных. При этом входные свойства задаются в ходе моделирования, а выходные получаем в результате воздействия входных свойств на объект. Для более полного изучения объекта входные свойства задаются в виде диапазона данных. Хотя они могут задаваться как константы или изменяемые параметры от других входных свойств.

Следует отметить, что прежде чем начать моделирование объекта решаются задачи при конкретных значений свойств (данных) для отладки или проверки правильности модели. Не существуют модели без этапа решения задач. Контрольные просчеты нужны не только на этапе создания, отладки модели, но и для проверки достоверности полученных данных при моделировании различных ситуаций.

Классификация моделей разнообразна, но нас интересуют математические и компьютерные модели. Компьютерную модель нельзя создать без математической модели.

Математическая модель - это приближённое описание какого-либо класса явлений внешнего мира, выраженное с помощью математической символики. Почему приближенное? Так как в модели все учесть нельзя! Стараются учесть в первую очередь только важное и необходимое.

На основе математического описания разрабатывается алгоритм модели. Далее разрабатывается программа модели в определенной среде программирования. Программа размещается в компьютере, заносятся исходные данные и запускается программа на выполнение. Компьютер начинает функционировать в соответствии с программой модели, обрабатываются данные и выдаются выходные данные в виде графиков или таблиц. Таким образом, компьютерная модель есть программа алгоритма, который разработан на основе математического описания модели.

Чем привлекательны компьютерные модели? Во-первых, можно в любое время проводить исследование реальных объектов необходимое число раз. Во-вторых, смело изменять исходные данные и получать результаты, которые для реальных объектов были бы «губительны». В-третьих, спокойно вносить изменения для улучшения модели или исправления ошибок.

Современный специалист должен иметь навыки моделирования. Для этого он должен знать свою предметную область, иметь знания в области математики, уметь составлять алгоритмы, знать основы программирования. В зависимости от специфики работы объем знаний, умений и навыков по представленным составляющим может быть различным.

Вопросы алгоритмизации и программирования рассматриваются в дисциплине информатика. Здесь хочется отметить, что в информатику входят вопросы информационных технологий. Многие часто принижают роль информатики и выдвигают на первый план дисциплину информационные технологии.

Как учить и что изучать на уроках информатики? На эти вопросы четко и однозначно нельзя ответить в силу ограниченности времени и отсутствия необходимых вычислительных средств и программного обеспечения. Основными изучаемыми темами в учебных заведениях являются приложения и программирование. Причем сейчас важно не только знать возможности приложений, но и уметь программировать.

С помощью приложения MS Excel можно изучить оба вопроса и затем спокойно перейти к изучению программирования на любом языке.

При использовании приложения MS Excel обычно решают различные расчетные задачи. Расчетные задачи характеризуются наличием кон-

кретных исходных данных и неизменными условиями, о чем говорилось выше.

Однако моделирование при изучении приложения позволит более глубоко узнать возможности MS Excel и основы программирования. Приложение MS Excel имеет ряд особенностей. Но самое главное на любом ПК всегда присутствует стандартный пакет MS Office с приложением MS Excel. Не нужно выбирать язык программирования, устанавливать дополнительно среду программирования. Нужно только иметь ПК, интерес, настойчивость и усидчивость.

1.2. О преимуществах MS Excel при изучении основ программирования

Отметим следующие преимущества приложения MS Excel с точки зрения приобретения навыков программирования и разработки моделей:

- MS Excel всегда включено в состав пакета MS Office.
- MS Excel имеет достаточно большой набор различных функций.
- MS Excel представляет возможность по созданию своих пользовательских функций.
- MS Excel имеет понятную систему индикации ошибок.
- MS Excel позволяет изменять формат результатов вычислений, а также создавать свой пользовательский формат.
- MS Excel имеет простые правила составления выражений для вычислений, которые изучаются в школе.
- Вычисления выполняются автоматически после изменения данных или расчетных выражений, которые вводятся после щелчка по клавише Enter.
- Пользователь «видит» не только графику, но и результаты вычислений по ходу моделирования или решения задачи.
- Пользователь может на любом этапе решения задачи просмотреть расчетные выражения, по которым производятся вычисления, а также включить режим *Показывать формулы, а не их значения*.
- Пользователь может сформировать промежуточные вычисления, которые отражают количественные параметры модели и наблюдать за их изменениями.
- MS Excel позволяет осуществлять вложения, количество которых достаточно для реализации большинства вычислений и сопоставимо с количеством вложений в других средах программирования.
- Так как MS Excel является табличным процессором, то пользователь должен для вычислений создавать расчетные выражения (формулы) в каждой ячейке. Что очень полезно в плане запоминания стандартных функций MS Excel.

- Пользователь при использовании счетчиков может наблюдать функционирование модели на любом нужном этапе.
- MS Excel имеет среду разработки MS Visual Basic Application, которая значительно облегчает решение задач путем компонентного способа создания программ.

Эти преимущества позволяют овладеть самостоятельно навыками разработки алгоритмов и программирования на более комфортном уровне, который занимает промежуточное положение между программированием прямыми кодами и объектно-ориентированным программированием. При этом развивать «изворотливость» и умение находить выходы из трудностей при моделировании из-за ограниченных возможностей ряда функций MS Excel.

Некоторые преподаватели ратуют за подходы к программированию с точки зрения использования «кубиков», которые собираются в определенном порядке для решения задачи или моделирования. В роли «кубиков» выступают готовые подпрограммы (функции, процедуры), которые выполняют определенные действия. При этом не важно, что у них внутри. Необходимо только знать, что нужно подать на вход и что получается на выходе «кубика». Такой подход целесообразно использовать при разработке очень сложных и больших по объему программ решения задач или моделирования. Этот подход является реализацией принципа структурированного программирования.

Однако на первоначальных этапах обучения программированию реализация такого подхода пагубна. В этом случае учатся пользоваться готовым «калькулятором». То есть нажимать на кнопки и не вдаваться в подробности вычислений. В случае неисправности «калькулятора» не будет возможности его исправить, так как обучаемые не имеют представления о его «начинке».

Поэтому нужно начинать программировать с кодов или, в крайнем случае, использовать элементарные функции, сущность которых ясна. Например, функция SIN() или СУММ() приложения MS Excel.

В этом случае обучаемый приобретает ряд черт, которые нужны программисту:

- Составлять самостоятельно или «читать» чужие алгоритмы.

- «Видеть» и «чувствовать» программу.
- Уметь разбивать алгоритм на «куски», которые несут определенный физический смысл.
- Определять промежуточные параметры при вычислениях для упрощения чтения программы или облегчения контроля программы.
- Выбирать из заданного диапазона данные, которые используются для проверки правильности программы.
- При необходимости поиска ошибок «жевать программу», а не глотать ее куски.
- Выделять куски программы для нахождения ошибок или проверки правильности их функционирования.
- Подбирать нужные данные для вывода наглядного отображения результатов функционирования программы или проверки ее правильности.
- Обеспечивать сохранность полученных данных для дальнейшего анализа.

При разработке алгоритма модели (решения задачи) главное не только определиться с последовательностью вычислений и применяемыми формулами, но и четко выделить входные и выходные величины.

Выяснить зависят ли величины от пространственных координат, времени или других величин. Это важно для организации фиксации их изменений, наглядного представления результатов моделирования и определения объема выходных массивов.

При этом желательно сразу определиться с константами и параметрами. Обычно константы являются физическими величинами, которые имеют конкретные размерности. Поэтому нужно согласовать размерности констант с величинами, которые используются в формулах (выражениях) при вычислениях. Параметры могут быть связаны не только с физикой модели, но и используются для контроля отдельных этапов вычислений при моделировании.

Программист должен уметь реализовывать представленный ему алгоритм на конкретном языке и знать особенности программирования. Однако программист всегда создает свои «маленькие алгоритмы» при про-

граммировании задачи. В жизни часто программисты не остаются в стороне от вопросов алгоритмизации. Это нужно учитывать на стадии изучения программирования. Необходимо уметь разрабатывать алгоритмы с учетом особенностей конкретной среды программирования, основных положений предметной области и поставленных целей исследования.

Итак, программист при создании компьютерной модели должен:

- Уяснить цель исследования.
- Выявить особенности модели.
- Разобраться с алгоритмом модели.
- Создать программу.
- Осуществить контрольный просчет на программе.
- Убедиться в работоспособности программы в заданном диапазоне исходных данных.
- Организовать фиксацию выходных данных.
- Устранить выявленные ошибки в программе или алгоритме.
- Провести необходимые улучшения программы или алгоритма.
- Написать инструкцию эксплуатации программы модели и проверить ее правильность.
- Сохранить программу модели на заданных носителях.

1.3. Основные особенности использования MS Excel при решении задач и моделировании

Любой продукт, создаваемый с использованием MS Excel, содержит три компонента:

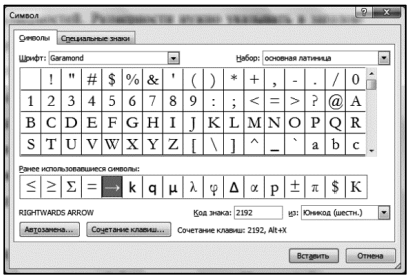
- таблицу входных данных;
- таблицу выходных данных;
- графическую компоненту.

В таблице входных данных находятся исходные данные, которые могут иметь различные форматы. Исходными данными могут быть числа, текст (символы), даты, время и т.п. Но самое главное в исходных таблицах не производят вычислений. Исходные данные могут быть представлены в таблице в виде одного или нескольких столбцов (строк), а также в виде совокупности или одиночных параметров.

В таблицах выходных данных должны присутствовать вычисления с использованием исходных данных. При этом в таблицах производятся как итоговые, так и промежуточные вычисления.

Графическая компонента позволяет наглядно изображать результаты вычислений, зависимости выходных данных от исходных данных или параметров, а также изменения динамики различных величин.

Остановимся на таблицах входных данных. В ячейки таблицы могут быть внесены числовые или текстовые данные. Текстовые данные могут состоять из букв или символов. Числовые данные могут иметь размерности. Однако для исключения неприятностей при расчетах целесообразно использовать числовые данные без размерностей. Размерности нужно указывать в заголовках столбов (строк) таблицы. Следует помнить, что все символы имеют числовые коды, которые можно увидеть в окне *Символы* при выборе символа для вставки (внизу окна).

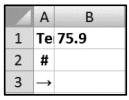


При записи текста или символа они прижимаются к левому краю ячейки по умолчанию (A1, A3). Числа всегда прижимаются к правому краю ячейки (A2). В противном случае всегда нужно обратить внимание на разделительный знак целой и дробной части числа (B2).

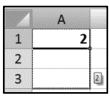
	A	B
1	Текст	75.9
2	75,9	
3	→	

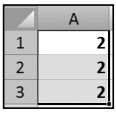
При форматировании числовых данных нужно обратиться к *Формат ячеек*, далее в окне выбрать закладку *Число* и активировать нужный формат из списка *Числовой формат*. Если отсутствует нужный формат, то можно создать свой пользовательский формат. Однако пользовательский формат целесообразно формировать для выходных данных.

При внесении исходных данных может возникнуть ситуация, когда вместо чисел в ячейке будет видна только решетка (А1). В этом случае нужно увеличить ширину столбца А1.

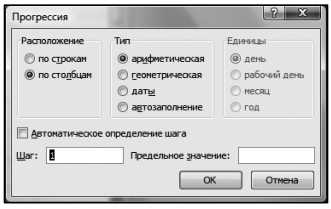


Так как исходных данных может быть достаточно много, то при их занесении нужно пользоваться механизмом автозаполнения. Простейший случай заключается в простом копировании данных занесенных в одну из ячеек. В этом случае выделяется ячейка (А1), курсор наводится на правый нижний угол ячейки до появления маркера автозаполнения в виде креста, нажимается левая клавиша мыши и протяжкой заполняем другие ячейки (А1:А3).

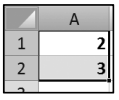


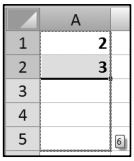


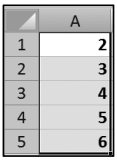
Можно после активизации ячейки (А1) протяжку осуществить правой клавишей. Затем ее отпустить и в контекстном меню выбрать пункт *Прогрессия*. В появившемся окне установить параметры для заполнения ячеек данными.



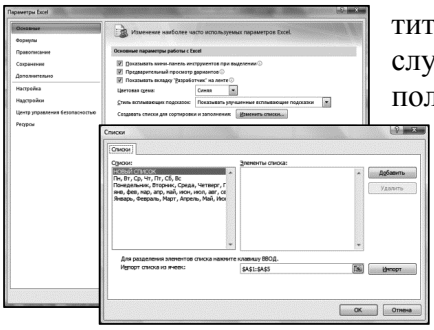
Но есть более простой и удобный способ, который позволяет контролировать значения данных в ходе автозаполнения. Заносим в первые две ячейки (А1, А2) данные и выделяем их. Курсор устанавливаем в нижний правый угол второй ячейки и левой клавишей заполняем протяжкой нужные ячейки. Программа показывает значение текущей последней ячейки при протяжке (А5). По первым двум значениям данных автоматически определяется шаг их изменения. В результате получаем пять значений исходных данных (А1:А5) в виде арифметической прогрессии от 2 до 6. В зависимости от первых двух значений прогрессии при протяжке вниз и вправо происходит увеличение (уменьшение) значений прогрессии, а при протяжке вверх или влево – уменьшение (увеличение).







Часто в исходных данных нужно отметить дни недели, месяца года и т.п. В этом случае можно с помощью механизма автозаполнения сформировать исходные данные. Для этого нужно занести названия дня недели или месяца, а затем аналогично, с помощью левой клавишей заполнить ячейки последовательными именами дней недели или месяцев из встроенных списков. При необходимости можно сформировать свои пользовательские списки, состоящие из текста или

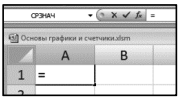


числовых данных. Для этого сначала формируем на листе пользовательский список, который полностью выделяем. Затем через кнопку *Microsoft Office* открываем окно *Параметры Excel*, из списка выделяем пункт *Основные*, а затем группу *Основные параметры для работы в Excel.* Щелкаем на кнопке *Изменить списки*. В появившемся окне *Списки* щелкаем на кнопке *Импорт* и кнопке *ОК.* Также щелкаем на кнопке ОК в окне *Параметры Excel.*

После заполнения таблицы исходных данных можно осуществить форматирование с использованием групп *Шрифт*, *Выравнивание* и *Число* на закладке *Главная*. При форматировании желательно делать так чтобы не изменялись ширина столбца и высота строк. Это позволит исключить изменения в других таблицах, которые будут находиться ниже или левее исходной таблицы. При изменении размеров ячеек с исходными данными целесообразнее в первую очередь уменьшить величину используемого шрифта.

В таблицах выходных данных должны всегда присутствовать расчетные выражения. Если вычисления отсутствуют, то нужно использовать приложение MS Word.

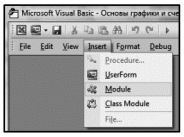
Все расчетные выражения начинаются со знака равенства. При внесении знака равенства в ячейку активизируется список формул в окне *Имя.* При использовании функций нужно помнить о разрешенном числе вложений. MS Excel позволяет вложить до 64 уровней функций. Однако вложениями не нужно увлекаться. Лучше сформировать промежуточные вычисления в виде параметров. В качестве аргументов целесообразно использовать ссылки (адреса) на данные. Это позволяет при изменении исходных данных автоматически получать результаты. При необходимости всегда нужно создавать таблицы параметров, констант и коэффициентов, которые используются в расчетных выражениях. При внесении в формулы в качестве аргументов ссылки на константы (коэффициенты) сразу их делать абсолютными с помощью клавиши F4. Если нужно сделать абсолютным адрес столбца или строки, то нужно щелкнуть по клавише несколько раз для снятия символа $ с соответствующей части адреса. Применение абсолютных адресов позволяет при применении автозаполнения для других значений исходных данных сохранять неизменными адреса констант в расчетных выражениях.

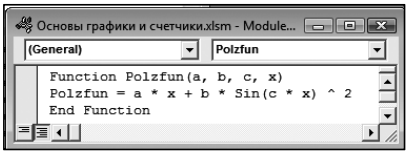


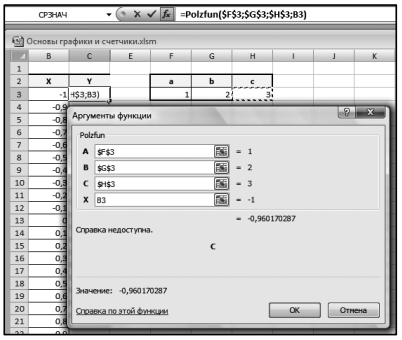
Если в выражении нужно, например, возвести в степень функцию, то следует помнить о том, что конструкция функции с аргументом неделима. То есть при табуляции функции $y = sin^2(x)$ в расчетное выражение для конкретного значения, записанного в ячейке A1, должно выглядеть следующим образом $= \sin(A1)\hat{}2$.

Иногда необходимо создать функцию, которой нет в списке. Тогда можно воспользоваться механизмом создания пользовательской функции. Пользовательскую функцию можно использовать при включении разрешения функционирования макросов или при сохранении файла с поддержкой макросов. Создадим функцию с 3-мя параметрами (коэффициентами) следующего вида $y = a * x + b * sin^2(c * x)$ для расчета при заданных значениях x.

Обратимся через закладку *Разработчик* к кнопке *Visual Basic* и вызовем окно редактора VBA. Выбираем пункт создания нового модуля *Module* в меню *Insert*. Сначала вносим слово Function, далее имя нашей функции Polzfun и обозначение входных переменных, необходимых для расчета значения функции. Формат аргументов делаем по умолчанию. Автоматически формируется текст окончание функции. Осталось внести текст программы самой функции, который понятен без объяснений. Закрываем окна редактора, сохраняем файл с поддержкой макросов. Теперь созданная функция находится в *Мастере функций* в категории *Определенные пользователем*. При необходимости вставки функции в расчетную формулу обращаемся к *Мастеру функций* и выделяем ее из списка. Заранее создаем таблицы входных данных для аргумента и параметров (коэффициентов) функции. При внесении адресов параметров с помощью клавиши F4 делаем их адреса абсолютными. После получения результатов для значения $x = -1$ с помощью механизма автозаполнения табулируем значения функции для других данных.

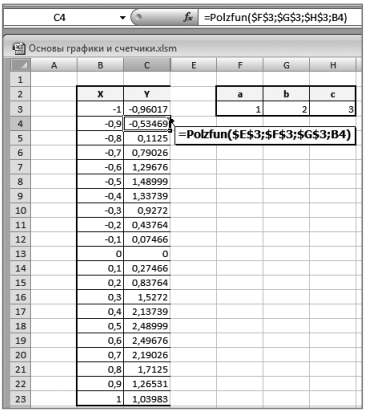






В случае отсутствия пересчета с использованием пользовательской функции для других значений аргумента нужно обратиться к *Параметры Excel*, выбрать пункт *Формулы* и в *Параметры вычислений* включить *автоматически*.

Далее форматируем таблицы и создаем примечание с расчетной формулой через контекстное меню для второго значения табулированной функции. Формулу копируем их из *Строки формулы* и вставляем в примечание. Из примечания предварительно удаляем

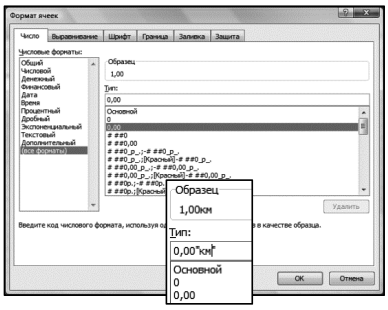


X	Y	a	b	c
-1	-0,96017	1	2	3
-0,9	-0,53469			
-0,8	0,1125			
-0,7	0,79026			
-0,6	1,29676			
-0,5	1,48999			
-0,4	1,33739			
-0,3	0,9272			
-0,2	0,43764			
-0,1	0,07466			
0	0			
0,1	0,27466			
0,2	0,83764			
0,3	1,5272			
0,4	2,13739			
0,5	2,48999			
0,6	2,49676			
0,7	2,19026			
0,8	1,7125			
0,9	1,26531			
1	1,03983			

вставку сведения о пользователе, которая является часто лишней. Не забываем при этом включить режим показа примечания через контекстное меню ячейки. Часто при печати примечания не печатаются. В этом случае необходимо включить разрешение печати примечаний. Для этого на вкладке *Разметка страницы* открыть окно *Параметры печати* и на закладке *Лист* в списке примечание выбрать *Как на листе*. При предварительном просмотре увидим созданные примечания. Таблицы табулированной функции в дальнейшем используем при рассмотрении графической компоненты.

Так как здесь упомянули о печати то, чтобы таблицы не «разваливались» по разным листам в начале работы необходимо разбивать листы на форматы через закладку *Вид* и кнопку *Разметка страницы*. Далее лучше вернуться к режиму *Обычный* и таблицы (графики) располагать в обозначенных границах листов, которые по умолчанию соответствуют формату А4.

Важно не только обеспечить правильные расчеты, но и наглядно представить результаты. При этом нужно точно указать размерности полученных чисел. Стандартные форматы чисел не всегда позволяют это сделать. Поэтому создают пользовательские форматы. Для создания пользовательского формата нужно вызвать окно *Формат ячеек*. В окне на закладке *Число* выбираем пункт *Все форматы*. В окне *Тип* выделяем формат 0,00 или какой-то другой и дописываем в апострофах, например, “км”. В этом случае получаем числовой формат с двумя знаками после запятой и подписью «км».



На рисунке показан пример применения созданного пользовательского формата. При этом сначала выделили нужные ячейки, а затем в списке *Все форматы* нашли созданный, выделили его и щелкнули на кнопке ОК.

	A	B
1	1	1,00км
2	2	2,00км
3	3	3,00км

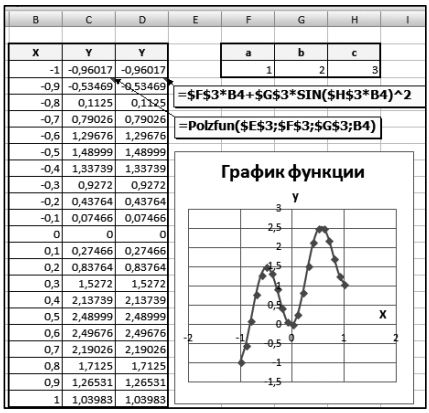
Графическая компонента необходима для наглядного отображения результатов расчетов или проведения моделирования. На графиках можно также определить уравнение регрессии и построить линию тренда как с прогнозом вперед, так и назад на заданные интервалы.

При построении графиков (диаграмм) сначала выделить столбцы (строки) в таблице исходных и (или) выходных данных. При этом обычно выделение производится протяжкой курсора по нужным ячейкам, а затем нужно обратиться к группе *Диаграмма* на закладке *Вставка* для обращения к нужному типу графика (диаграммы). Часто нужные для выделения столбцы (строки) не расположены рядом. В этом случае для выделения

несмежных столбцов (строк) нужно нажать на групповую клавишу *Ctrl* и продолжить выделение. Для графической иллюстрации решения задач и моделирования в основном будем использовать точечный тип диаграммы. При этом будем выбирать следующие виды: точечная с отрезками или гладкими кривыми, с маркерами, без маркеров или только маркеры. Тогда данные, соответствующие аргументам, будут располагаться по горизонтальной оси координат, а данные, соответствующие функции – по вертикальной оси координат.

Для повышения контрастности и точности изображения производим форматирование отдельных элементов графиков через их контекстное меню и обращения к пункту *Формат ряда данных*, *области диаграммы*, *точки данных* и т.п. Для форматирования точки данных нужно предварительно на ней дважды щелкнуть, а затем вызывать контекстное меню.

В качестве примера приведем график табулированной функции с использованием пользовательской функции и дополнительного столбца с составленным расчетным выражением (формулой) со стандартной функцией. Перед формированием дополнительного столбца вставили столбец для отделения его от таблицы коэффициентов. Создаем формулу для расчетов значения функции для первого значения x. При этом адреса (ссылки) данных формируем щелчками по соответствующим ячейкам. Если адрес должен быть абсолютным, то щелкаем на клавише *F4*. В конце формирования щелкаем на клавише *Enter*. Если этого не делать, то в расчетное выражение будут записываться адреса активных ячеек. Убеждаемся, что результаты вычислений совпадают и, используя автозаполнение, формируем формулы для остальных значений x. После дополнительного форматирования и выделения столбцов аргумента и функции обращаемся к графику *Точечная с гладкими кривыми и маркерами*. Добавляем основные линии сетки для оси аргумента через ее контекстное меню. Используя *Макеты диаграмм* вставляем название осей и диаграммы. Удаляем ненужные элементы.



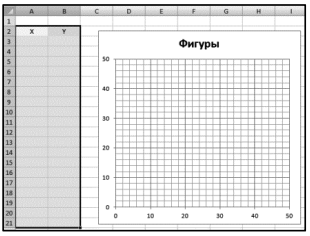
X	Y	Y
-1	-0,96017	-0,96017
-0,9	-0,53469	-0,53469
-0,8	0,1125	0,1125
-0,7	0,79026	0,79026
-0,6	1,29676	1,29676
-0,5	1,48999	1,48999
-0,4	1,33739	1,33739
-0,3	0,9272	0,9272
-0,2	0,43764	0,43764
-0,1	0,07466	0,07466
0	0	0
0,1	0,27466	0,27466
0,2	0,83764	0,83764
0,3	1,5272	1,5272
0,4	2,13739	2,13739
0,5	2,48999	2,48999
0,6	2,49676	2,49676
0,7	2,19026	2,19026
0,8	1,7125	1,7125
0,9	1,26531	1,26531
1	1,03983	1,03983

a	b	c
1	2	3

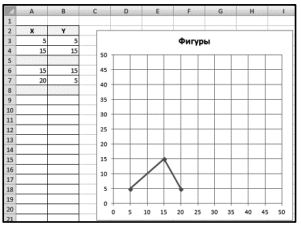
Если во втором столбце табулированный функции сделали расчеты по другой формуле, то при выделении трех столбцов получили графическое изображение двух разных кривых графиков. Хотелось напомнить об одной характерной ошибке. При обращении к типу *График* в последнем случае получили бы три кривые, а по оси аргументов были бы сформированы номера строк таблиц от 1 до 21, а не значения аргумента x.

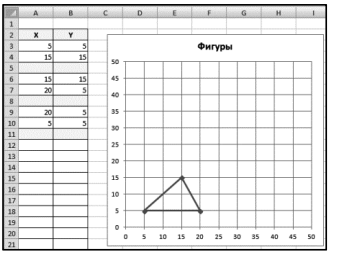
Рассмотрим методику построения простейших геометрических фигур, которая понадобиться при решении задач. Простейшим элементом фигуры для нас будет прямая линия. Прямая линия задается двумя координатами ее концов.

Обозначим заголовки столбцов таблицы координат. Для наблюдения построения фигур выделим с запасом область ячеек таблицы, в которую будем записывать координаты элементов фигур, и обратимся к типу диаграмм *Точечный график с прямыми отрезками и маркерами*. Через контекстное меню осей установим фиксированные минимальные и максимальные значения осей, добавим основные линии сетки. Растянем область диаграммы так, чтобы ячейки сетки диаграммы стали квадратными. Теперь можно начинать вносить координаты элементов фигур в таблицу и при необходимости их корректировать.

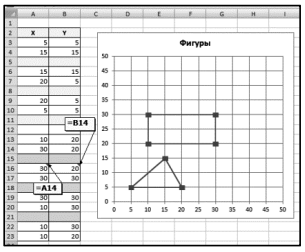


Первым построим треугольник. Координаты будем привязывать к узлам сетки. Координаты в нашем случае имеют условную размерность. Занесем координаты линий двух сторон треугольника. Как видно координаты конца одной линии совпадают с координатами начало другой. Между ячейками координат линий оставляем пустую строку. Осталось сформировать координаты третей стороны треугольника. Координата начала совпадает с координатой конца второй линии, а координата конца совпадает с началом первой линии. Получили изображение треугольника. В ходе построения иногда целесообразно откорректировать параметры сетки диаграммы.





Аналогично задаем координаты для построения прямоугольника. При этом формировать координаты линий можно с помощью ссылок на соответствующие координаты других линий (см. примечание). Это упрощает задачу построения замкнутых фигур.



Осталось построить правильный шестигранник. Выберем в качестве центра шестигранника середина ячейки сетки. Расчет координат концов линий граней будем делать по следующим формулам:

$$x = x_c + R * cos\varphi \ ,$$

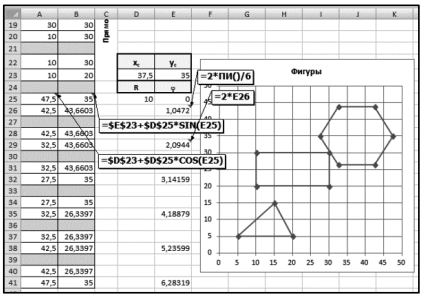
$$y = y_c + R * sin\varphi \ ,$$

где x_c , y_c - координаты центра шестиугольника,

R- радиус окружности, в которую вписан шестиугольника,

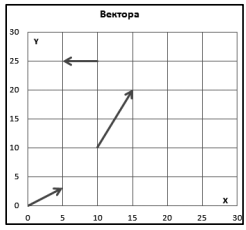
φ – угол кратный 60^0 и отсчитываемый от оси ординат против часовой стрелки.

Возьмем радиус равный 10 условным единицам. Занесем величину радиуса (D25) и координаты центра (D23,E23) в таблицу параметров. В ячейку E25 занесем начальное значение угла φ для расчета начальной координаты первой грани. Формулы расчета представлены в примечаниях. Далее в ячейке E26 формируем значение угла φ для расчета конечной координаты первой грани с использованием константы π. В функции угол подставляем в размерности радиан. Координаты начала второй грани формируем из координат конца предыдущей. Для расчета координат конца формируем значения угла φ с использованием значения угла в ячейке E26. В результате видим на графике две грани шестигранника. Аналогично формируем координаты других граней. При изменении координат центра будем наблюдать перемещение фигуры, а при изменении величины радиуса – увеличение или уменьшение размеров фигуры.



При увеличении числа граней до 15-20 в многоугольнике можно получить хорошее изображение окружности.

На графиках можно изображать векторы. Создадим таблицу координат (A3:B19), выделим ее и обратимся к диаграмме *Точечная с прямыми отрезками и маркерами*. Зададим координаты линий векторов. Произведем форматирование ряда данных. Вызываем контекстное меню линии любого вектора и выходим на пункт *Формат ряда данных*. Выбираем в окне пункт *Тип линии* и в *Параметры стрелки* выбираем прямую для *Тип начала* и стрелку - для *Тип окончания*. В *Параметры марк*ера устанавливаем *Нет* для *Тип* маркера.



Произведем сложение второго и третьего вектора. Получим третий вектор с координатами

$$x_c^2 = x_1^1 \text{ , } x_c^1 = x_1^2 + (x_2^2 - x_2^1) \text{ , } y_c^2 = y_1^1 \text{ , } y_c^1 = y_1^2 + (y_2^2 - y_2^1) \text{ ,}$$

где x_c^1 и y_c^1 – координаты начала вектора суммы векторов,

x_c^2 и y_c^2 – координаты конца вектора суммы векторов,

x_1^1 и y_1^1 – координаты начала первого вектора суммы,

x_1^2 и y_1^2 – координаты конца первого вектора суммы,

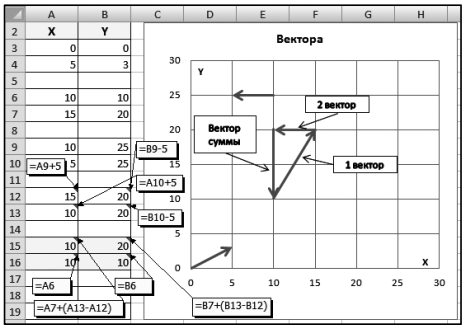
x_2^1 и y_2^1 – координаты начала второго вектора суммы,

x_2^2 и y_2^2 – координаты конца второго вектора суммы.

В соответствии с приведенными формулами, вычислим координаты суммарного вектора в таблице.

При сложении геометрически нужно сместить второй вектор суммы. Для этого преобразуем координаты третьего вектора в таблице (A9:B10) и запишем их со смещениями в ячейки A12:B13 (см. примечания). В этом случае координаты начала второго вектора суммы совмещается с концом первого вектора суммы (A6:B7).

Осталось теперь вычислить координаты суммарного вектора в ячейках A15:B16 (см. примечания). На графике видим геометрическую интерпретацию сложения векторов. Можно без труда создать модель разности двух векторов.



Следует отметить, что можно для вычислений координат результирующих векторов создать функции или процедуры для автоматизации процесса моделирования действий с векторами.

1.4. Создание счетчиков для автоматизации процесса моделирования

Счетчики необходимы для осуществления моделирования и при решении динамических задач. Счетчики могут быть детерминированными и случайными, периодическими и непериодическими, а также генерировать числа, время или даты.

Функционировать счетчики могут автоматически или автоматизировано, запускаться (останавливаться) – вручную или при выполнении заданных условий.

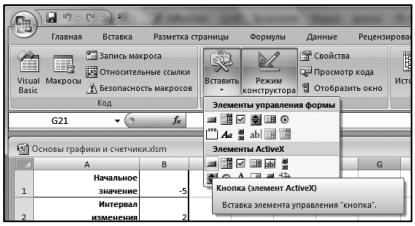
Рассмотрим создание счетчиков, которые будут в дальнейшем применяться при решении задач.

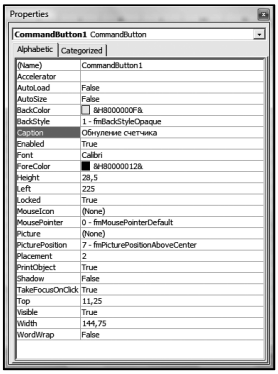
Самым простым счетчиком является выбранная ячейка, в которую периодически заносятся нужные значения с клавиатуры ПК. Очень удобно ее использовать на этапе отладки решения задачи.

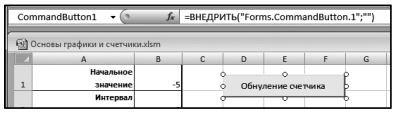
Значения счетчика можно задать в виде таблицы (столбца) входных данных, когда точно известны необходимые значения при решении задачи. В этом случае можно сразу увидеть возникающие ошибки в расчетах при определенных значениях счетчика. Но такие счетчики не очень удобны при демонстрации моделей или динамических задач.

Воспользуемся возможностями MS Excel для автоматизации функционирования счетчиков. Счетчик должен устанавливаться в начальное положение, автоматически начисляться при щелчках на кнопке и иметь возможность коррекции интервала изменения значений. Создадим счетчик целых чисел с заданным интервалом изменений. Сформируем ячейки с начальным значением, интервалом изменений и саму ячейку счетчика. Теперь можно приступить к формированию кнопок изменения значений счетчика и его «обнуления» или задания начального значения. Обращаемся к закладке *Разработчик* и в группе *Элементы управления* из списка *Вставить* выбираем кнопку из *Элементы ActiveX*.

	A	B
1	**Начальное значение**	-5
2	**Интервал изменения**	2
3	**Счетчик**	-5







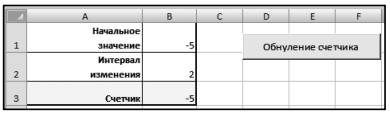
Протяжкой определяем размеры кнопок, в контекстном меню кнопки через пункт *Свойства* вызываем окно *Properties* (Свойства) и в *Caption* (Заголовок) вносим

название «Обнуление счетчика».

В режиме *Конструктора* дважды щелкаем на кнопке и вносим следующий текст программы:

```
Private Sub CommandButton1_Click()
Cells(3, 2) = Cells(1, 2)
End Sub.
```

В функции Cells указываем номер строки и номер столбца нужной ячейки. Таким образом, счетчику присваивается начальное значение. После создания программы кнопки закрываем книгу с сохранением изменений и открываем с установкой разрешения функционирования макросов. Щелкнув на кнопке присваиваем счетчику начальное значение.

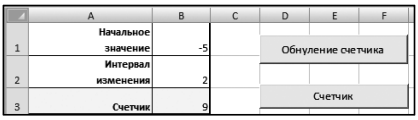


Аналогично создаем кнопку для начисления счетчика под названием «Счетчик».

Программа кнопки имеет следующий вид:

```
Private Sub CommandButton2_Click()
Cells(3, 2) = Cells(3, 2) + Cells(2, 2)
End Sub.
```

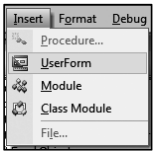
Теперь при щелчке на созданной кнопке будут увеличиваться значения счетчика.

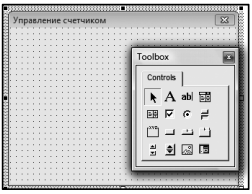


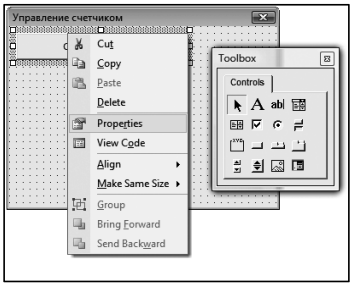
Можно сделать дополнительную кнопку для уменьшения значений счетчика. В этом случае в программе вместо плюса нужно поставить минус.

Иногда целесообразно создать диалоговое окно с помощью форм для управления счетчиком. Попробуем создать диалоговое окно на базе приведенного выше примера.

Через закладку *Разработчик* в группе *Код* щелкаем на кнопке *Visual Basic*. Появляется окно интегрированной среды разработки языка VBA. В меню *Insert* выбираем *UserForm.* В окне свойств *Alphabetic* выбираем пункт название *Caption* и вносим слово *Управление счетчиком*, в пункте *Name* вносим имя формы *Chet*. На панели инструментов Toolbox выбираем командную кнопку CommandButton и протяжкой размещаем ее на окне формы. Через контекстное меню кнопки вы-

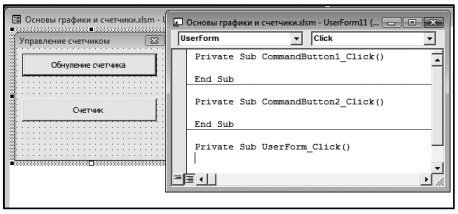




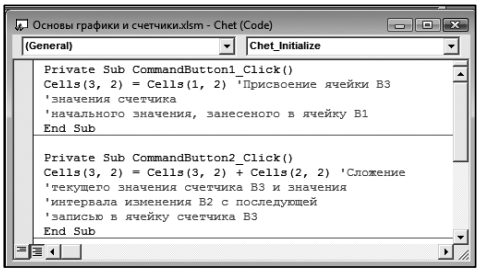


ходим на ее свойства Properties, где в пункте название записываем *Обнуление счетчика*. Аналогично создаем вторую кнопку с названием *Счетчик*. Сразу можно напомнить о том, что используя пункты свойств формы и кнопок можно без затруднений изменить их цвет, шрифт надписей на кнопках. Можно также уменьшить размер формы с кнопками.

Теперь можно перейти к программному коду кнопок и внести соответствующие им программы. Программы объектов будут запускаться по щелчку на соответствующих кнопках.



Щелкнем дважды по первой, затем по второй кнопке и окнах кодов увидим заготовки программ для кнопок и формы. Процедуры объектов начинают функционировать по щелчку *Click*. Внесем последовательно коды программ для кнопок и комментарии. Первая кнопка обеспечивает установку начального значения (обнуление) счетчика, а вторая по щелчку изменяет значение счетчика на заданный шаг.



Теперь можно заняться программой формы. Здесь нужно отметить варианты вызова формы на лист. Существуют два наиболее удачных способа вызова формы:

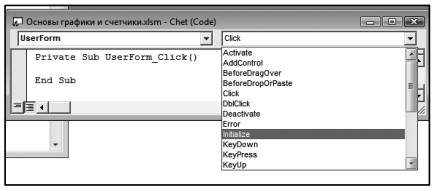
- Вызов через командную кнопку, созданную на листе.
- Вызов при открытии книги.

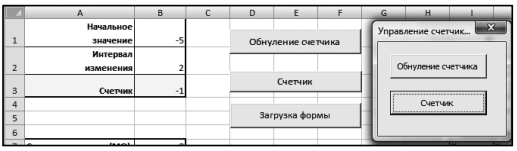
Рассмотрим вызов формы через кнопку. Для этого создаем кнопку на листе через закладку *Разработчик*, группу *Элементы управления*. В списке *Вставить* выбираем кнопку из *Элементы ActiveX*.

Протяжкой определяем размеры кнопки на листе, в контекстном меню кнопки через пункт *Свойства* вызываем окно *Properties* (Свойства) и в *Caption* (Заголовок) вносим название *Загрузка формы*. В режиме *Конструктора* дважды щелкаем на кнопке и вносим следующий текст программы:

```
Private Sub CommandButton3_Click()
Chet.Show  'Показ объекта формы с именем Chet
End Sub.
```

Далее обращаемся к программе форма и в списке событий выбираем Initialize. Вносим вместо UserForm Chet. Все сохраняем. При щелчке по кнопке *Загрузка формы* вызывается форма, а

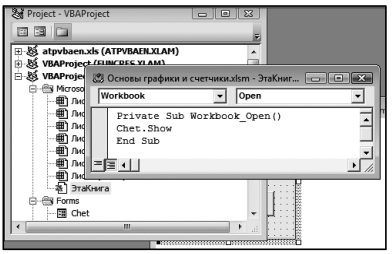




при щелчке на управляющем элементе *Закрыть* формы она закрывается.

Осталось сформировать вызов формы при открытии книги. В этом случае нужно в списке *Microsoft Excel Object* дважды щелкнуть на объекте книги *ЭтаКнига* для вызова окна кодов. Далее внести программу процедуры следующего вида:

```
Private Sub Workbook_Open()
Chet.Show
End Sub.
```



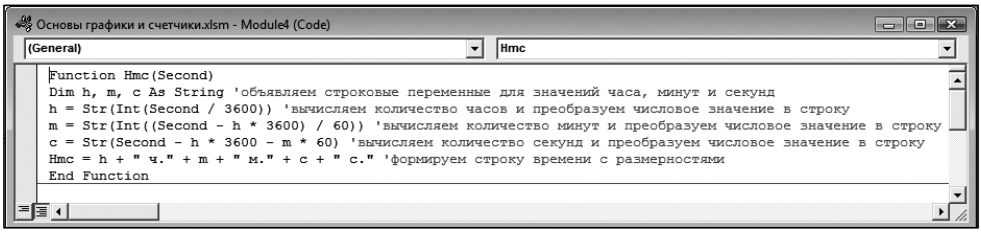
Функция Workbook_Open() автоматически запускается при открытии книги и выполняется записанная программа.

Сохранить и закрыть книгу. При вызове книги нужно разрешить функционирование макросов и элементов ActiveX. Далее появиться созданная форма управления счетчиком.

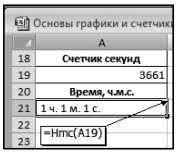
Выбор вызова формы зависит в основном от индивидуальных пристрастий пользователя ПК. Нюансы работы в интегрированной среде программирования VBA всегда можно выяснить в материалах, размещенных в Интернете.

Работа со счетчиками времени связана с небольшими особенностями. Обычно в счетчиках времени используется время в секундах от полуночи или от начала суток. Поэтому возникают задачи пересчета времени из секунд в часы, минуты и секунды, а затем обратного пересчета в секунды из заданного времени в часах, минутах и секундах. Для решения этих задач можно использовать уже имеющиеся функции или создать пользовательские.

Создадим функцию перевода секунд в часы, минуты, секунды. Исходим из того, что в 1 минуте 60 секунд, а в 1 часе 60 минут. Опять обратимся через закладку *Разработчик* к кнопке *Visual Basic* и вызовем редактор VBA. Выбираем пункт создания нового модуля *Module* в меню *Insert*. Вносим слово Function, далее имя нашей функции *Hmc* и обозначение входной переменной *Second*, с помощью которой будет передаваться значение счетчика секунд от начала суток. В расчетах будем использовать функцию определения наименьшей целой части числа Int(). Так как время целесообразно показывать с размерностями, то количество часов, минут и секунд после расчета преобразовываем в строки с помощью функции Str(). В конце формируем строку времени с указанием размерностей соответствующих частей.



После внесения текста программы функции закрываем окна редактора, сохраняем файл с поддержкой макросов. Теперь созданная функция находится в *Мастере функций* в категории *Определенные пользователем*. Пример применение пользовательской функции Hmc() пересчета счетчика секунд в часы, минуты, секунды представлен на рисунке.



При моделировании для ряда задач необходимы счетчики на базе датчиков случайных чисел. Наиболее распространенным датчиком является датчик с равномерным законом, который обладает наибольшей энтропией. Часто используются датчики с нормальным законом распределения.

Создадим счетчик на основе датчика с равномерным распределением чисел. При создании такого счетчика обычно используется функция генератора случайных чисел Rnd() и процедура Randomize. Функция Rnd (параметр) возвращает случайные числа из интервала [0,1), подчиняющиеся равномерному закону распределения. От значения параметра зависит генерация последовательности случайных чисел.

Значение параметра	Что генерируется?
Меньше нуля	Каждый раз одно и тоже число. Параметр используется как начальное значение.
Равно нулю	Последнее сгенерированное число.
Больше нуля или параметр отсутствует	Следующие случайное число в последовательности.

Перед вызовом функции Rnd() необходимо выполнить команду Randomize для инициализации (запуска) генератора случайных чисел. При этом устанавливается начальное число для генератора, которое соответствует одному из чисел в его псевдослучайной последовательности из 232 чисел. Randomize лучше использовать без параметра, чтобы запускать генератор случайного числа с начальным числом, основанным на системном таймере.

Пусть задано среднее значение диапазона чисел числом a и разброс чисел относительно среднего значения – b. Тогда для генерации целых чи-

сел в диапазоне от (a-b) до (a+b) можно использовать следующие выражение: Int((a-b)+(2*b+1)*Rnd). Прибавление 1 в коэффициенте при Rnd связано с обеспечением получения максимального возможного числа (a+ b). Так как генерируются числа от 0 (включительно) до 1, которая не входит в диапазон генерации чисел.

Формируем на листе ячейки *Среднее значение* (B7), *Разброс* (B8) , *ДСЧ-равномерный* (B9) и создаем командную кнопку со следующей программой:


```
Private Sub CommandButton5_Click()
Randomize
Cells(9, 2) = Int((Cells(7, 2) - Cells(8, 2)) + (2 * Cells(8, 2) + 1) * Rnd)
End Sub.
```

При щелчке на кнопке *ДСЧ-равномерный* получаем случайное число в диапазоне от -7 до +7.

Интересен вопрос генерации случайных чисел с нормальным распределением. Существует ряд методов создания датчиков случайных чисел с нормальным законом распределения. Остановимся на методе, основанном на центральной предельной теореме, которая позволяет вычислить математическое ожидание и дисперсию суммарного закона распределения по характеристикам суммируемых случайных величин. При отсутствии доминирующих факторов в суммируемых случайных величинах получается случайная величина, распределенная по нормальному закону.

Вычисление случайной величины производится по формуле

$$x = m + \sigma(\sum_{i=1}^{12} y_i - 6),$$

где m - математическое ожидание нормального закона,

σ - среднеквадратическое отклонение нормального закона,

y_i – случайное число с равномерным распределением в интервале от 0 до 1.

Сумма из 12 случайных величин y_i позволяет использовать датчик для решения простых задач. При необходимости повышения достоверности генерации случайных чисел, распределенных по нормальному закону, количество суммируемых случайных величин увеличивают. Формулы расчета можно найти в Интернете.

Так как обычно при моделировании используют несколько датчиков, то дополним в названия параметров равномерного закона названия параметров нормального закона (МО – математическое ожидание, СКО – среднеквадратическое отклонение) и создадим командную кнопку для генерации чисел по нормальному закону (НЗ).

Программа командной кнопки представлена ниже:

```
Private Sub CommandButton9_Click() 'ДСЧ-НЗ
```

```
Randomize 'формирование начального случайного значния для генератора случайных чисел
Y1 = Rnd: Y2 = Rnd: Y3 = Rnd: Y4 = Rnd: Y5 = Rnd: Y6 = Rnd: Y7 = Rnd
Y8 = Rnd: Y9 = Rnd: Y10 = Rnd: Y11 = Rnd: Y12 = Rnd
'12 случайных чисел с равномерным законом распределения от 0 до 1
y = Y1 + Y2 + Y3 + Y4 + Y5 + Y6 + Y7 + Y8 + Y9 + Y10 + Y11 + Y12
Cells(9, 2) = Int(Cells(7, 2) + Cells(8, 2) * (y - 6)) 'случайное величина по НЗ
End Sub.
```

Отмечаем, что точки в конце текста программы не ставятся.

Следует отметить, что случайные числа генерируются в диапазоне $\pm 3\sigma$ с вероятностью 0,993. Это нужно учитывать при задании параметров нормального закона. Пример оформления счетчика представлен на рисунке.

Здесь уместно обсудить вопрос автоматизации функционирования счетчиков. В этом случае нужно будет только обеспечить запуск и останов начисления счетчика. Вопрос автоматизации тесно связан с привязкой начисления счетчика к реальному времени. Это важно, так как часто моделирование осуществляется в реальном масштабе времени (РМВ). Начнем создание счетчика РМВ с формирования командных кнопок под названиями *Обнуление счетчика*, *Запуск счетчика РМВ* и *Останов*, которые реагируют на щелчок.

Для автоматизации используем процедуру приложения OnTime (EarliestTime, Procedure, LatestTime, Schedule). Аргумент процедуры EarliestTime определяет момент запуска нужной процедуры, аргумент Procedure - имя запускаемой процедуры. Аргумент LatestTime - если на момент запуска процедуры, Excel не может ее запустить в силу того, что выполняется другое действие. Аргумент LatestTime определяет последнее время запуска процедуры. Если этот аргумент опущен, то Excel будет ждать до тех пор, пока не сможет выполнить эту процедуру. Аргумент Schedule может иметь значения True и False. При True выполнение процедуры откладывается на сутки, а при False процедура не выполняется. То есть нужная процедура под именем Procedure начинает выполняться в заданный момент реального времени EarliestTime.

Определим ячейки: *Счетчик РМВ* (E1), *Временной интервал счетчика* (E4), *Количество тактов* (E2), *Величина тактов* (E3). Временной интервал в простейшем случае равен 1 секунде. При необходимости замедлении темпа начисления счетчика нужно увеличить временной интервал. Целесообразно предусмотреть подсчет количества тактов начисления

счетчика, которое можно использовать для ускорения или замедления времени моделирования.

В программе командной кнопки *Обнуление счетчика* предусмотрено обнуление только ячеек *Счетчик РМВ* и *Количество тактов:*

```
Sub CommandButton6_Click() 'Обнуление ячеек счетчика по щелчку
Cells(1, 5) = Null 'Обнуление ячейки счетчика времени
Cells(2, 5) = Null 'Обнуление ячейки тактов счетчика времени
End Sub.
```

В программах других кнопок вызываем соответствующие процедуры

```
Sub CommandButton4_Click() 'Запуск счетчика по щелчку
Call Bklshetchika 'Вызов процедуры включения счетчика
End Sub
Sub CommandButton5_Click() 'Останов счетчика по щелчку
Call Stopchet 'Вызов процедуры остановки счетчика
End Sub.
```

Процедура включения счетчика выглядит следующим образом:

```
Dim Zadaniecledtime As Date
Sub Bklshetchika() 'Запуск счетчика с заданным интервалом
Zadaniecledtime = Now + Cells(4, 5)  'Запись значения времени для запуска счетчика с учетом интервала
Application.OnTime Zadaniecledtime, "Zapistime" 'Вызов процедуры записи счетчика времени в заданное время
End Sub.
```

Для формирования времени запуска процедуры zapistime используем переменную Zadaniecledtime, в которую записываем текущее время и прибавляем заданный интервал в ячейке E4. Предварительно ячейке E4 задаем временной формат ч:м:с.

При наступлении заданного времени Zadaniecledtime производится запись текущего значения времени и количество тактов:

```
Sub Zapistime() 'Процедура записи текущего значения счетчика и тактов
Cells(1, 5) = Now 'Запись текущего значения времени
Cells(2, 5) = Cells(2, 5) + Cells(3, 5) 'Начисление тактов счетчика времени
Call Bklshetchika
End Sub.
```

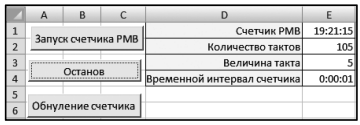
После записи текущего значения времени опять обращаемся к Bklshetchika для формирования следующего момента времени начисления счетчика.

Для отображения в ячейки счетчика только времени (без даты) задаем формат времени.

Для прекращения начисления счетчика присваиваем параметру Schedule:=False для остановки запуска процедуры Zapistime:

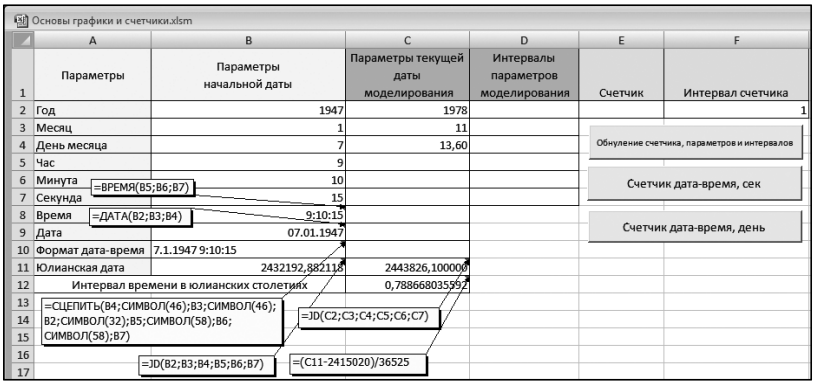
```
Sub Stopchet() 'Процедура остановки счетчика
Application.OnTime EarliestTime:=Zadaniecledtime, Procedure:="Zapistime", Schedule:=False 'Формирование запрета вызова процедуры записи значения времени счетчика
End Sub.
```

В начале моделирования сначала нужно задать интервалы времени и тактов, затем обнулить ячейки и щелкнуть на кнопке *Запуск счетчика PMB*. При необходимости в процессе моделирования можно изменять значения интервалов. Иногда возникает необходимость фиксации времени начала моделирования. Это можно сделать без особых затруднений.



Осталось рассмотреть применение дат и времени при моделировании. Самое главное здесь нужно знать точки отсчета дат, которые используются в формулах задачи и при генерации в ПК. Это обусловлено тем, что при моделировании обычно используют не даты, а количество суток или время от точки отсчета. Поэтому первое действие, которое нужно осуществить это из заданной даты вычесть дату нужную дату отсчета, а затем определить количество суток, часов или минут и т.д. при необходимости. Следует отметить, что стандартные функции даты в Excel не очень удобны. Лучше воспользоваться функциями в VBA. В VBA тип даты формируется от 1 января 100 г. до 31 декабря 9999 года. Функции вычислений с датами позволяют получать результат в годах (лет) до секунд. При этом в функции указывается аргумент, который может иметь следующие значения: "yyyy" год, "q" квартал, "m" месяц, "y" день года, "d" день, "w" день недели, "ww" неделя, "h" час, "n" минута, "s" секунда. Если производим форматирование, то количество символов в составляющих аргумента связано с количеством отображаемых чисел параметра даты (времени).

Самое главное, что является основой счетчика: год, количество дней, часы, минуты или секунды. При этом нужно обеспечить вычисление текущей даты моделирования по изменяемому значению счетчика, которое по иерархии может быть в самом сложном случае счетчиком секунд. При этом может понадобиться не только текущее значение даты моделирования, но и интервал (промежуток) даты (времени) от начальной даты. Следовательно, в начале создадим ячейки для записи отдельных параметров начальной даты и затем соберем их в одну ячейку с форматом даты. В ячейки параметров заносим данные от значения года до секунд.



Далее используя стандартные функции (см. примечания) и формируем ячейки со стандартным форматированием для времени и даты. При создании ячейки для отображения даты и времени применяем функцию сцепки СЦЕПИТЬ(). Для разделения данных используем функцию символ СИМВОЛ () с кодами: 46 – точка; 32 – пробел; 58 – двоеточие. Коды символов легко определить с помощью функции КОДСИМВ(), где в качестве аргумента записывается нужный символ.

Перейдем к созданию счетчика для формирования текущей даты (времени) моделирования и интервалов параметров, которые могут понадобиться при моделировании. Необходимо отметить, что не все параметры могут понадобиться при моделировании конкретной задачи. Поэтому отдельные параметры нашего счетчика могут быть исключены из модели.

Сначала озаглавим и отформатируем ячейки для параметров, интервалов и счетчика. Далее в режиме *Конструктор* создадим командные кнопки со следующими названиями: *Обнуление счетчика, параметров и интервалов*; *Счетчик дата-время, сек*; *Счетчик дата-время, день*.

Программа для обнуления:

```
Private Sub CommandButton2_Click()
Range("c2:d10", "e2") = Null 'Обнуление области параметров, интервалов и счетчика
Cells(1, 5) = "Счетчик" 'Запись надписи без размерности
Cells(1, 6) = "Интервал счетчика" 'Запись надписи без размерности
End Sub.
```

содержит операторы для записи заголовков счетчика и интервалов счетчика без размерностей. Это сделано для того, чтобы при использовании счетчика секунд или дней, а вернее суток, сформировать соответствующие названия.

Программа для счетчика дней (суток) имеет следующий вид:

```
Private Sub CommandButton3_Click()
Dim Tymdhms As Date 'Объявление переменной для начисления счетчика
```

```
Cells(2, 5) = Cells(2, 5) + Cells(2, 6) 'Начисления счетчика дней
Tymdhms = DateAdd("d", Cells(2, 5), Cells(10, 2)) 'Увеличение начальной даты на величину счетчика дней
Cells(2, 3) = Year(Tymdhms) 'Определение текущего года
Cells(3, 3) = Month(Tymdhms) 'Определение текущего месяца
Cells(4, 3) = Day(Tymdhms) 'Определение текущего дня месяца
Cells(5, 3) = Hour(Tymdhms) 'Определение текущего часа
Cells(6, 3) = Minute(Tymdhms) 'Определение текущей минуты
Cells(7, 3) = Second(Tymdhms) 'Определение текущей секунды
Cells(8, 3) = Format(Tymdhms, "h:m:s") 'Форматирование текущего времени
Cells(9, 3) = Format(Tymdhms, "d/m/yyyy") 'Форматирование текущей даты
Cells(10, 3) = Format(Tymdhms, "d/m/yyyy h:m:s") 'Форматирование текущего даты и времени
Cells(1, 5) = "Счетчик, день" 'Запись надписи с размерностью день
Cells(1, 6) = "Интервал счетчика, день" 'Запись надписи с размерностью день
Cells(2, 4) = DateDiff("yyyy", Cells(10, 2), Cells(10, 3)) 'Формирование интервала в годах
Cells(3, 4) = DateDiff("m", Cells(10, 2), Cells(10, 3)) 'Формирование интервала в месяцах
Cells(4, 4) = DateDiff("d", Cells(10, 2), Cells(10, 3)) 'Формирование интервала в днях
Cells(5, 4) = DateDiff("h", Cells(10, 2), Cells(10, 3)) 'Формирование интервала в часах
Cells(6, 4) = DateDiff("n", Cells(10, 2), Cells(10, 3)) 'Формирование интервала в минутах
Cells(7, 4) = DateDiff("s", Cells(10, 2), Cells(10, 3)) 'Формирование интервала в секундах
End Sub.
```

В программу включена процедура DateAdd("d", Cells(2, 5), Cells(10, 2)) сложения ячеек счетчика дней и начальной даты. Параметр "d" указывает на сложение в днях. Переменная Tymdhms имеет стандартный формат даты. Далее выделяются параметры даты и записываются в соответствующие ячейки. Так как в стандартном формате переменной Tymdhms не отражаются секунды, то с помощью процедуры Format() производим форматирование. Следует напомнить, что если укажем аргумент дня с двумя символами dd, то дни будут изображаться двумя цифрами и в соответствующих случаях отображаться с нулем. Это положение распространяется на другие параметры даты.

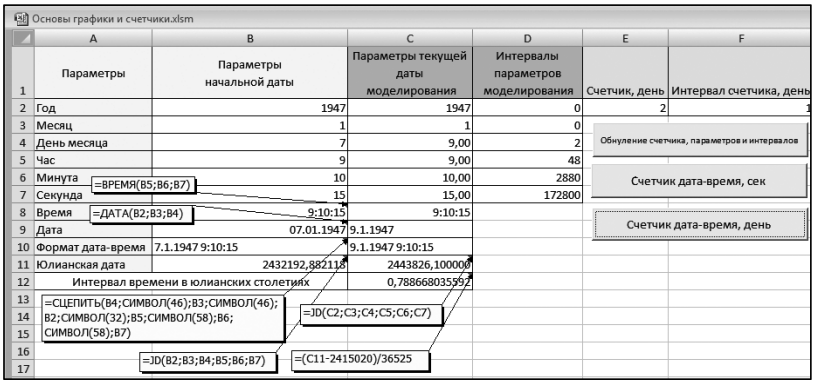
Для вычисления интервалов параметров используется процедура DateDiff(), где от начальной даты вычитается текущая дата моделирования с указанием соответствующего признака параметра (год, месяц и т.д.).

В программе записываются название счетчика и интервала с указанием размерности день.

Для моделирования с использованием счетчика секунд используется аналогичная программа командной кнопки:

```
Private Sub CommandButton1_Click()
Dim Tymdhms As Date 'Объявление переменной для начисления счетчика
Cells(2, 5) = Cells(2, 5) + Cells(2, 6) 'Начисления счетчика секунд
Tymdhms = DateAdd("s", Cells(2, 5), Cells(10, 2)) 'Увеличение начальной даты на величину счетчика секунд
Cells(2, 3) = Year(Tymdhms)
Cells(3, 3) = Month(Tymdhms)
Cells(4, 3) = Day(Tymdhms)
Cells(5, 3) = Hour(Tymdhms)
Cells(6, 3) = Minute(Tymdhms)
Cells(7, 3) = Second(Tymdhms)
Cells(8, 3) = Format(Tymdhms, "h:m:s")
Cells(9, 3) = Format(Tymdhms, "d/m/yyyy")
Cells(10, 3) = Format(Tymdhms, "d/m/yyyy h:m:s")
Cells(1, 5) = "Счетчик, сек"
Cells(1, 6) = "Интервал счетчика, сек"
Cells(2, 4) = DateDiff("yyyy", Cells(10, 2), Cells(10, 3))
Cells(3, 4) = DateDiff("m", Cells(10, 2), Cells(10, 3))
Cells(4, 4) = DateDiff("d", Cells(10, 2), Cells(10, 3))
Cells(5, 4) = DateDiff("h", Cells(10, 2), Cells(10, 3))
Cells(6, 4) = DateDiff("n", Cells(10, 2), Cells(10, 3))
Cells(7, 4) = DateDiff("s", Cells(10, 2), Cells(10, 3))
End Sub.
```

В начале моделирования щелкаем на кнопке обнуления, заносим значение интервала моделирования, а затем щелкаем по соответствующей кнопке счетчика. Следует отметить, что в нашем варианте счетчика отсутствует начальное нулевое значение. Это нужно учитывать при разработке модели. На рисунке показано применение счетчика дней.



Следует напомнить, что при расчетах координат планет используется интервал времени в юлианских столетиях от конкретной даты T. Для расчета интервала необходимо вычислить текущую юлианскую дату JD от Гринвичского полудня всемирного времени, который соответствует полдню 1 января 4713 г. до н.э юлианского календаря, и знать юлианскую дату начальной эпохи JD0, которая используются в формулах вычисления положения планет. Чаще всего начальная дата JD0 привязывается к полуночи 1-го января 1900 года и равняется 2415020,0. Юлианские сутки начинаются в полдень, а не в полночь как календарные сутки.

Вычисление интервала времени в юлианских столетиях производится по выражению $T = (JD\text{-}JD0)/36525{,}0$, где 36525,0 – величина юлианского столетия в солнечных сутках.

Текущее значение юлианской даты вычисляется по формуле

$$JD = INT(365{,}25 * y) + INT\big(30{,}6001 * (m + 1)\big) + d + t + 1720994{,}5,$$

где y - значение года при номере месяца больше 2 и значение года на 1 меньше для января и февраля,

m – параметр, который равен номеру месяца при условии, что значение месяца больше 2, и равен сумме 12 и номера месяца для января и февраля.

d - номер дня в месяце,

t - время в долях суток от полуночи,

1720994,5 - юлианская дата.

Так как время должно быть привязано к Гринвичскому меридиану, то при вводе Московского времени нужно вычитать 3 часа.

Если дата больше 15 октября 1582 года, которая является началом григорианского календаря, то к текущей юлианской дате нужно вычислить добавку b по формуле

$$b = 2 - a + INT(a/4),$$

где $a = INT(y/100)$.

Если время задаем с учетом секунд, то необходимо при приведении к долям суток помнить что в сутках 86400 секунд.

Создадим пользовательскую функцию для вычисления юлианской даты с аргументами, которые соответствуют параметрам дат в нашей таблице. Программа вычисления юлианской даты представлена ниже:

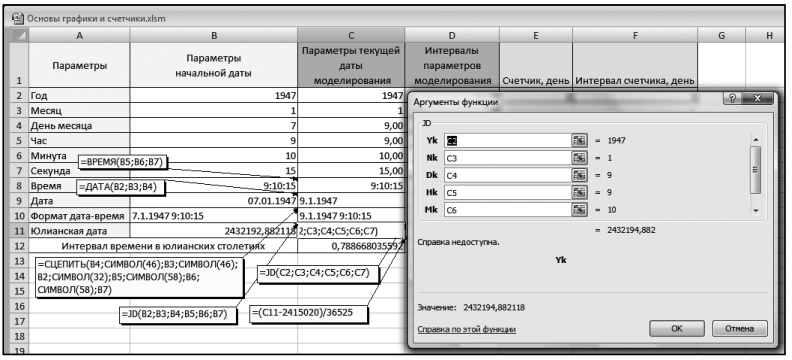
```
Function JD(yk, nk, dk, hk, mk, sk) 'вычисление юлианской даты по календарным датам больше 15 октября 1582 года
If nk > 2 Then
y = yk: m = nk
Else
y = yk - 1: m = nk + 12
End If
a = Int(y / 100): b = 2 - a + Int(a / 4)
t = (hk * 3600 + mk * 60 + sk) / 86400 'расчет доли суток
JD = Int(365.25 * y) + Int(30.6001 * (m + 1)) + dk + t + 1720994.5 + b
End Function.
```

Следует отметить, что при задании исходной даты дни представлены с дробной частью суток, то параметры времени нужно задать нулевые (см. C2:C4).

Для расчета юлианской даты в ячейки B11 и C11 вставили пользовательскую функцию JD() с параметрами календарных дат. Для определения интервала в юлианских столетиях T внесли расчетное выражение в ячейку C12 (см. примечание).

Для более плавного перехода к программированию в других средах рассмотрим в отдельных моделях использование форм с рядом управляющих элементов.

На этом закончим рассмотрение вопроса создания счетчиков для моделирования. При решении задач или моделировании рассмотренные вопросы создания счетчиков будем опускать с надеждой, что читатель самостоятельно сможет создать необходимый счетчик для конкретной модели.



2. Моделирование с использованием MS Excel

2.1. Математика

2.1.1. Вращение графиков

Задан отрезок прямой и спираль Архимеда. Нужно построить их графики и обеспечить вращение относительно центра прямоугольных координат.

Начнем с массива координат прямой и спирали с начальными условиями. Затем во вспомогательном массиве будем осуществлять пересчет координат с учетом текущего угла поворота, который определяется скоростью изменения и счетчиком времени. Этот массив координат будет служить для построения графиков прямой и спирали.

Отрезок прямой зададим координатами ее концов, а спираль Архимеда уравнением в полярной системе координат как

$$R = a * \varphi,$$

где R – радиус,

a - параметр, который определяет скорость изменения радиуса при изменении угла,

φ - угол.

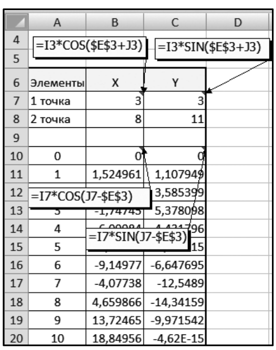
Задаем координаты прямой в ячейках G3:H4 и параметр спирали в ячейке M3. Размерность координат задаем в условных единицах, а углов – в радианах. Для реализации модели вращения пересчитываем прямоугольные координаты концов прямой в полярные по формулам

	G	H	I	J	K	L	M
1	=(G3^2+H3^2)^0,5 Начальные прямоугольные координаты концов отрезка		=ATAN(H3/G3) Начальные полярные координаты			=ПИ()/5 Шаг угла спирали, рад	Параметр спирали
2	X	Y	R	φ, рад			
3	3	3	4,242641	0,785398		0,628319	3
4	8	11	13,60147	0,942			

$$R_i = (x_i^2 + y_i^2)^{0,5},$$

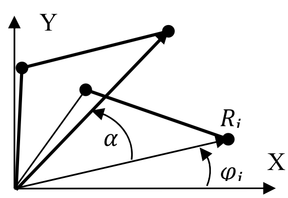
$$\varphi_i = arctg\, y_i / x_i,$$

где i=1,2.



	A	B	C	D
4	=I3*COS($E$3+J3)		=I3*SIN($E$3+J3)	
5				
6	Элементы	X	Y	
7	1 точка	3	3	
8	2 точка	8	11	
9				
10	0	0	0	
11	1	1,524961	1,107949	
12	=I7*COS(J7-$E$3)		3,585399	
13	[illegible]	-1,74743	5,378098	
14	4	=I7*SIN(J7-$E$3)	[illegible]	
15	5	[illegible]	[illegible]	
16	6	-9,14977	-6,647695	
17	7	-4,07738	-12,5489	
18	8	4,659866	-14,34159	
19	9	13,72465	-9,971542	
20	10	18,84956	-4,62E-15	

Полярные координаты прямой запишем в ячейки I3:J4. Формируем массив координат графиков для изображения через диаграммы (A7:C20). В ячейках B7:C8 вычисляем прямоугольные координаты концов прямой с учетом угла поворота по формулам (см. рисунок)

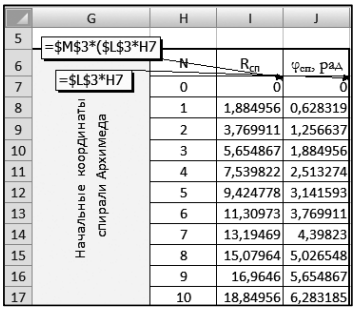


$$x_i = R_i * \cos(\varphi_i + \alpha),$$

$$y_i = R_i * \sin(\varphi_i + \alpha),$$

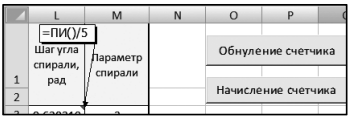
где α – угол поворота графика.

Угол поворота вычисляется (E3) с учетом угловой скорости (A3) и значения счетчика (C3). Значение счетчика можно изменять путем записи с клавиатуры. Для автоматизации процесса моделирования поворота графиков создадим ячейку, которая определяет шаг изменения значения счетчика (D3). Следует отметить, что поворот прямой осуществляется против часовой стрелки при положительном значении угловой скорости (A3). В массиве A10:C20 формируем 11 точек для изображения спирали Архимеда, где пересчитываем полярные координаты в прямоугольные (см. примечания). Можно изменить количество точек. При пересчете используем массив полярных координат спирали без учета угла поворота (I8:J17). Координаты точек вычисляем с учетом номера точки, который определяет полярный угол точки спирали (см.примечания). Так как при пересчете координат в массиве A10:C20 вычитаем значение угла поворота, то спираль вращается по часовой стрелке. Итак, прямая вращается по часовой стрелке, а спираль – против. Можно изменить вращение графиков на противоположный путем задания отрицательного значения угловой скорости.



Выделяем массив A7:C20 и через закладку *Вставка*, в группе *Диаграмма* выбираем график *Точечная с гладкими кривыми и маркерами*. Целесообразно зафиксировать в пункте *Формат оси* контекстного меню оси максимальные и минимальные значения осей. Получим изображение графиков, которые вращаются при изменении значения счетчика.

Автоматизируем изменение значение счетчика. Для этого создадим кнопки *Обнуление счетчика* и *Начисление счетчика*. На закладке *Разработчик*, в группе *Элементы управления*, в списке *Вставить* выбираем в *Элементы ActiveX* элемент *Кнопка*. Протяжкой определяем место и размер изображения кнопки на листе. Не закрывая объект *Кнопка,* через контекстное меню выходим на *Свойства* и в пункте *Caption* заносим название кнопки.



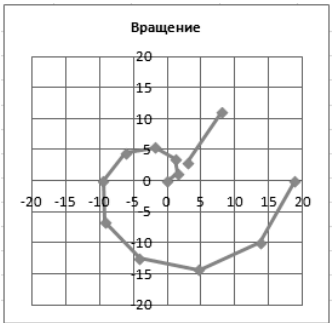
Далее щелкнув дважды на соответствующей кнопке, заносим следующие тексты программ:

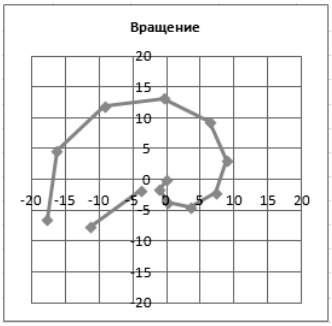
```
Private Sub CommandButton1_Click()
Cells(3, 3) = 0 'Обнуление счетчика'
End Sub
Private Sub CommandButton2_Click()
Cells(3, 3) = Cells(3, 3) + Cells(3, 4) 'Начисление счетчика'
```

End Sub.

Точку не надо ставить в тексте программы кнопок. Стиль ссылок в программах кнопок соответствует R1C1. В этом случае сначала указывается номер строки, а затем номер столбца в функции Cells. Обнуление и начисление счетчика осуществляется путем щелчков на соответствующих кнопках. При моделировании можно изменить шаг начисления счетчика (D3 или Cells(3, 4)).

Обязательно сохраните файл с расширением *Книга Excel с поддержкой макросов*, а при открытии файла включить разрешение на функционирование макросов. На рисунках представлены графики в начальном положении и после начисления счетчика.





В нижней таблицы представлены уравнения кривых в полярной системе координат, которые могут быть использованы для моделирования.

Уравнение	Название
$R = a * \cos(b * \varphi + \varphi_0)$	полярная роза
$R = a + b * \varphi$	спираль Архимеда
$R = a * \sin(2 * \varphi)$	двулистник
$R = a * sin(4 * \varphi)$	четырехлистник
$R^2 = a^2 * \cos(2 * \varphi)$	лемниската Бернулли
$R = a * (1 + \cos(\varphi))$	кардиоида, a >0
$R = (3 * a * \cos(\varphi) * \sin(\varphi))/(cos^3(\varphi) + sin^3(\varphi))$	лист Декарта, a >0
$R = 2 * a * \cos(\varphi) + b$	улитка Паскаля
$R = 1 + \cos(3 * \varphi) + sin^2(3 * \varphi)$	лист щавеля
$R = a * \exp(b * \varphi)$	логарифмическая спираль
$R = a * cos^{0,5}(2 * \varphi)$	лемниската Бернулли

2.1.2. Комплексное нахождение корней уравнений

Задано уравнение. Нужно найти точное значение корней уравнения. Формулы вычисления корней уравнения отсутствуют.

Точное вычисление корней произведем с использованием процедуры *Поиск решения*. Уравнения часто имеют сложный вид. Поэтому целесообразно предварительно определить наличие корней уравнения, области их нахождения и приближенные значения.

При нахождении корней сначала применим графоаналитический метод нахождения корней уравнения, условного форматирования и затем процедуры *Поиска решения.*

Графоаналитический метод позволяет наглядно определить наличие корней, их количество и приближенные значения.

С помощью условного форматирования графически выделяем на графике интервалы нахождения корней.

Процедура *Поиск решения* обеспечивает точное определение значений всех корней уравнения.

В качестве примера рассмотрим вычисление корней на определенном интервале следующего уравнения

$$7 * \cos(3 * x) + 3 = x^2.$$

Для предварительного определения значений корней и их количества применяем графоаналитический метод. Преобразуем исходное уравнение. Перенесем все члены из правой части в левую часть уравнения. В правой части уравнения целесообразно оставить значение нуль

$$7 * \cos(3 * x) - x^2 + 3 = 0.$$

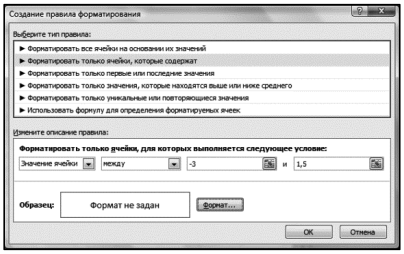
Теперь представляем левую часть уравнения как функцию y

$$y = 7 * \cos(3 * x) - x^2 + 3.$$

Осуществляем табуляцию функции y в заданном диапазоне значений x с шагом, который обеспечивал достаточно точное графическое изображение функции (A2:B22). Значения коэффициентов (7,3,-1 и 3) при стандартных функциях целесообразно оформить в виде констант. Это позволит при необходимости провести дополнительный анализ функции при изменении констант.

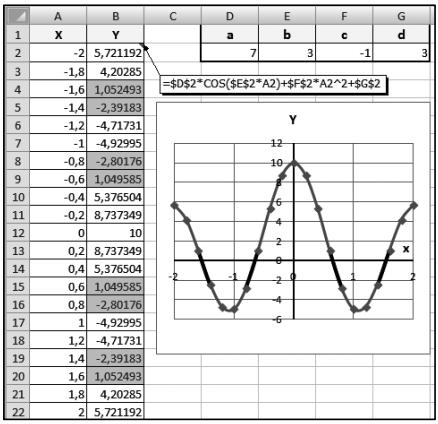
После табуляции и построения графика функции определяем интервалы значений x, где график функции пересекается с осью ординат.

При построении графика функции выбираем диаграмму *Точечная с гладкими кривыми и маркерами*.



Для наглядности проводим условное форматирование значений функции y, которые находятся рядом с точками пересечения с осью ординат. Выделяем столбец ячеек значений функции. На линейке *Главная*, в

разделе *Стили* щелкаем на списке *Условное форматирование* и выбираем пункт *Между* в *Правилах выделения ячеек*. В окне *Между* заносим значения -3 и 1,5, которые выбираем исходя из значений табулированной функции в районе пересечения с осью x. Наблюдаем при этом, какие ячейки значений функции y форматируются. При необходимости изменяем значения границ диапазона условного форматирования для более четкого выделения на графике областей пересечения функции y с осью x.

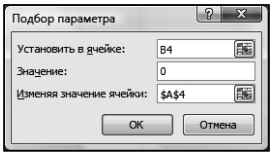


	A	B	C	D	E	F	G
1	X	Y		a	b	c	d
2	-2	5,721192		7	3	-1	3
3	-1,8	4,20285					
4	-1,6	1,052493					
5	-1,4	-2,39183					
6	-1,2	-4,71731					
7	-1	-4,92995					
8	-0,8	-2,80176					
9	-0,6	1,049585					
10	-0,4	5,376504					
11	-0,2	8,737349					
12	0	10					
13	0,2	8,737349					
14	0,4	5,376504					
15	0,6	1,049585					
16	0,8	-2,80176					
17	1	-4,92995					
18	1,2	-4,71731					
19	1,4	-2,39183					
20	1,6	1,052493					
21	1,8	4,20285					
22	2	5,721192					

В областях пересечения с осью ординат можно изменить цвет отрезков функции y при форматировании соответствующих маркеров точек, принадлежащих диапазонам условного форматирования. Наводим курсор на нужную точку отрезка функции, дважды щелкаем на ней и через контекстное меню выходим на пункт *Формат точки данных*. Далее устанавливаем необходимые параметры ряда.

Целесообразно также провести форматирование осей графика для их более четкого выделения.

После форматирования элементов графика табулированной функции, предварительной оценки количества корней и их приближенных значений применяем процедуру *Подбор параметра* для точного нахождения значений корней уравнения. Корни уравнения соответствуют точкам пересечения графика функции y с осью x. В качестве предварительных значений корней целесообразно использовать значения x в выделенных интервалах. В качестве значения первого корня выберем ячейку A4. В ячейке B4 значение функции y достаточно близко к нулю. Далее активизируем ячейку функции B4 и обращаемся к процедуре *Подбор параметра,* которое находится на вкладке *Данные* в разделе *Работа с данными* в списке *Анализ "что-если"*. В окне *Подбор параметра* устанавливаем адрес B4, значение нуль и адрес изменяемой ячейки A4. Щелкаем на кнопке *ОК* и сохраняем результат. Первый корень равняется 1,54098, а значение функции равно -4E-06, которое очень близко к нулю. Чтобы значение функции в таблице равнялось нулю целесообразно произвести форматирование ячеек функции и установить числовой формат 2 или 3 знака после запятой.



	A	B
1	X	Y
2	-2	5,721192
3	-1,8	4,20285
4	-1,54098	-4E-06
5	-1,4	-2,39183

Затем проводим аналогичное применение процедуры *Подбор параметра* для вычисления других корней. Не забывая корректировать адреса ячеек в окнах. По мере нахождения точных значений корней на графике

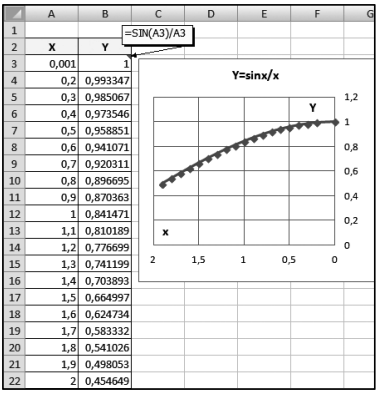
наблюдаем смещения маркеров точек корней функции y на ось ординат x. Дополнительно можно провести проверку корней путем подстановки их значений в исходное выражение уравнения.

Предложенная методика должна обеспечить лучшее понимание использования процедуры *Подбор параметра* и нахождение всех корней уравнений в заданном диапазоне значений x. При изменении значений констант функции можно осуществлять моделирование изменения корней уравнения.

2.1.3. Определение пределов

Создадим графические модели для определения замечательных пределов.

Графическая модель будет основываться на таблицах табулированных функций, где аргументы будут формироваться с учетом особенностей замечательных примеров.

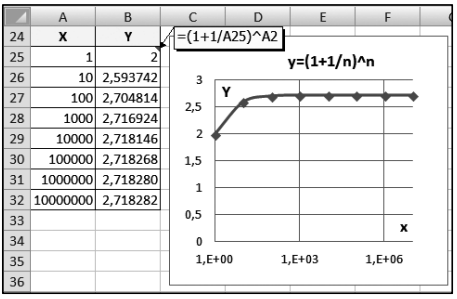


X	Y
0,001	1
0,2	0,993347
0,3	0,985067
0,4	0,973546
0,5	0,958851
0,6	0,941071
0,7	0,920311
0,8	0,896695
0,9	0,870363
1	0,841471
1,1	0,810189
1,2	0,776699
1,3	0,741199
1,4	0,703893
1,5	0,664997
1,6	0,624734
1,7	0,583332
1,8	0,541026
1,9	0,498053
2	0,454649

Создадим сначала модель для замечательного предела

$$\lim_{x\to 0} \sin x/x = 1.$$

Деление на ноль осуществлять нельзя. Поэтому создадим последовательность аргументов с 0,1 до 2. Далее создадим столбец с $y = \sin x/x$. Выделим ячейки табулированной функции (A3:B22) и обратимся к диаграмме *Точечная с гладкими кривыми и маркерами*. Щелкнем по оси аргументов и через контекстное меню выйдем на пункт *Формат оси*. Далее в окне в Параметры оси установим *обратный порядок значений*. На графике наблюдаем стремление значений функции к 1. Изменим первое значение x с 0,1 на 0,01 и значение функции измениться с 0,999983 на 1. Это еще раз подтверждает стремление предела к 1.



X	Y
1	2
10	2,593742
100	2,704814
1000	2,716924
10000	2,718146
100000	2,718268
1000000	2,718280
10000000	2,718282

Создадим модель для предела

$$\lim_{n\to\infty} \left(1+\frac{1}{n}\right)^n = e.$$

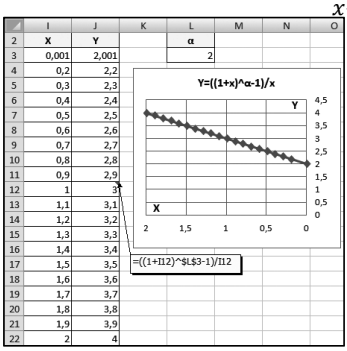
Здесь изменение аргумента обеспечим с 1 до 10^7 . Расчет значений функции производим по формуле $y = \left(1+\frac{1}{n}\right)^n$. Далее строим график табулированной функции, где значения оси аргументов форматируем как экспоненциальные с числом десятичных

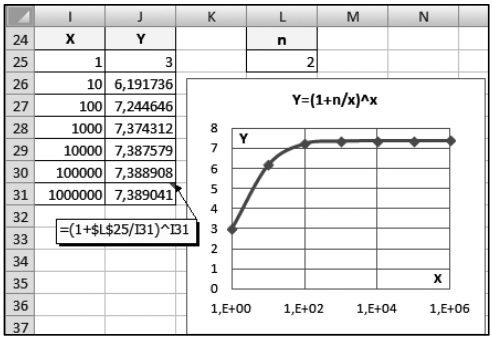
знаков равных нулю. На графике видим, что предел стремиться к замечательному числу e =2,7182. Приведем графические модели для пределов

$$\lim_{x\to\infty}\left(1+\frac{n}{x}\right)^{x}=e^{n}$$

и

$$\lim_{x\to 0}((1+x)^{\alpha}-1)/x=\alpha.$$





После просмотра графических моделей определения значений пределов можно более осмысленно приступить к вычислением пределов с использованием аналитических методов.

2.1.4. Встреча путников

Из пункта А и В одновременно вышли путники. Скорости движения у них разные. Расстояние между пунктами (L_{AB}) задано и равно 21км. Через сколько времени они встретятся?

Построим графики траекторий движения путников из пунктов отправления для нахождения времени встречи. Траектории представим в виде отрезков прямых, длина которых соответствует скорости движения и текущего времени счетчика. Момент встречи фиксируется при соприкосновении траекторий. Изображение траекторий будем начинать для каждого путника из своего пункта отправления.

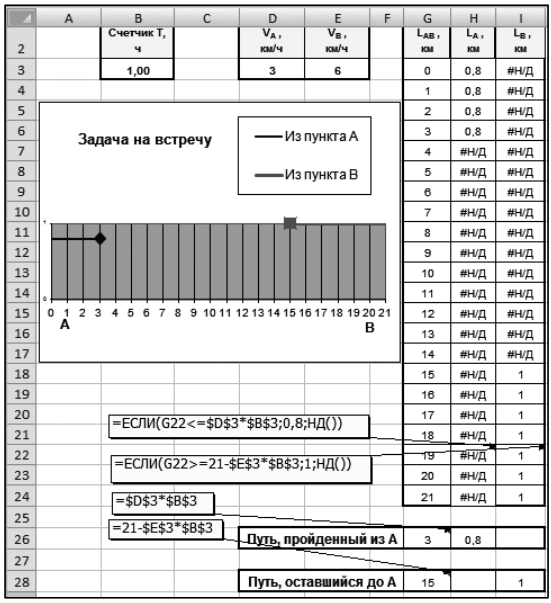
Для моделирования движения путников создаем таблицу, где основными параметрами будут расстояние между пунктами и интервал расстояния. Величина интервала определяет точность моделирования движения. Возьмем интервал равный 1км. Начало координат установим в пункте А.

Тогда для отображения пройденного пути от пункта А (L_A) используем выражение $L_A = V_A * T$, а для отображения пройденного пути от пункта Б (L_B) – $L_B = L_{AB} - V_B * T$.

В выражениях V_A и V_B скорости передвижения путников из пункта А и Б соответственно, а Т – текущие время с момента начала движения путников.

Создадим столбец расстояний (G3:G24) от пункта А до Б с интервалом 1км.

Для изображения на диаграмме траектории пройденного пути каждым путником на соответствующих интервалах дальности задаем определенные значения ординат. В этом случае исключим слияние траекторий движения. Для путника, вышедшего из пункта А установим 0,8 условных единиц, а для другого путника – 1. Этим исключается неоднозначность в определении момента встречи, которая может быть при пересечении траекторий при одинаковых значениях ординат.



Отображать траектории при моделировании необходимо в любое заданное время. Для этого в столбцах L_A и L_B (H3:I24) проверяется условие прохождения каждого участка пути на текущее время (см. примечание) и при выполнении условия формируется соответствующие значение ординаты или в противном случае в ячейку записывается функция *НД()* для исключения возможности индикации участков траектории пути, которые еще не прошли. Для формирования маркеров, которые отображали бы место нахождения путников на их траекториях движения, в дополнительных ячейках первого столбца L_{AB} таблицы рассчитываем текущие положения путников по вышеприведенным выражениям с использованием значения счетчика времени. Ячейки (*Путь, пройденный из А* и *Путь, оставшийся до А*) отделяются от строк таблицы для удобства задания нужных параметров форматирования. Значения ординат для маркеров каждого путника задаем заранее и они соответствуют значениям 0,8 и 1.

Конечно, обязательно создаем счетчик текущего времени (*Счетчик Т*) и ячейки для задания скоростей движения путников (V_A и V_B), в которые записываем скорости движения путников.

Теперь выделив все столбцы таблицы (G3:I28), обращаемся к *Мастеру диаграмм*. Далее выбираем в диалоговом окне тип *Точечная*, а *вид – Точечная диаграмма, на которой значения соединены отрезками.* Производим необходимое текстовое оформление диаграммы. Для формирования маркеров путников наводим указатель мыши на нужную точку графика,

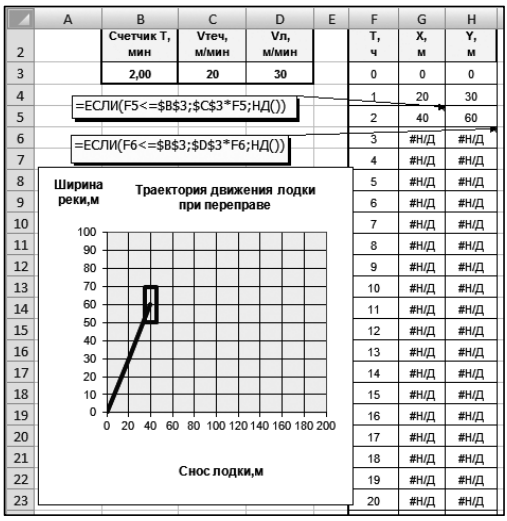
щелкаем на ней и после появления крестообразного указателя со стрелками вызываем контекстное меню точки. Переходим на пункт *Формат точки данных* и в диалоговом окне устанавливаем нужные параметры маркера. Аналогичным образом можно задать параметры линии траектории движения. Только после наведения указателя мыши на нужную траекторию выходим на контекстное меню ряда и пункт *Формат рядов данных*.

Остается только в ячейку *Счетчика Т* задавать время для моделирования движения путников и зафиксировать момент встречи. По мере приближения путников к точке встречи значение времени нужно заносить с дробной частью. Для повышения точности определения момента встречи нужно уменьшить величину интервала дальности.

2.1.5. Переправа через речку

Через речку необходимо переправиться на лодке. Задана скорость течения реки ($V_{\text{теч}}$) и собственная скорость лодки ($V_{\text{л}}$), вектор которой перпендикулярен берегу. Определить точку причаливания и время переправы при заданной ширине реки.

С учетом направленности векторов скоростей можно вычислить координаты лодки по формулам $= V_{\text{теч}} * T$ и $Y = V_{\text{л}} * T$, где Т - время с момента начала переправы. Координата X направлена вдоль берега по вектору течения, координата Y – по направлению вектора собственной скорости лодки. Вектор собственной скорости лодки перпендикулярен линии берега. Создаем ячейки для заданных скоростей и счетчика текущего времени. Далее формируем таблицу для расчета координат лодки. В первый столбец заносим значения времени Т (F3:F24) с заданным интервалом. Максимальное значение времени зависит от возможного времени переправы при различных начальных условиях задачи.



Как и в предыдущей задаче, расчетные координаты траектории лодки записываются в ячейки после проверки условия. Здесь проверяется не превышение времени, которое зафиксировано в соответствующей ячейке таблицы, текущего времени счетчика. На этом можно остановиться и, выделив столбцы с координатами X и Y (G3:H24), обратиться к *Мастеру диаграмм* для моделирования траектории лодки.

Самое главное зафиксировать максимальные значения координат по оси *X* и *Y*. Максимальное значение по оси *Y* должно равняться ширине реки.

При моделировании переправы интересно показать положение лодки. Представим лодку в виде прямоугольника. Пусть половина ширины лодки равняется *dx* , половина длины – *dy* (D29:D30).

	C	D	E	F	G	H
24				21	#Н/Д	#Н/Д
25						
26	Центр лодки			2,00	40	60
27						
28	Размеры лодки					
29	dx	5		Контур лодки	35	70
30	dy	10			45	70
31					45	50
32					35	50
33					35	70

В дополнительных ячейках определим положение центра лодки на текущий момент времени по вышеприведенным формулам (F26:H26). Далее последовательно двигаясь в выбранном направлении, вычисляем координаты крайних точек сторон прямоугольника (G29:H33). Для замыкания контура лодки в последних ячейках (G33, H33) должны быть повторены координаты первой выбранной точки (F29, H29). Если при обращении к *Мастеру диаграмм* в исходный диапазон ячеек будут включены ячейки с координатами контура лодки, то совместно с траекторией пути получим контур лодки. При использовании изложенного принципа можно изображать и другие фигуры на диаграммах. Об этом написано ранее.

Изменяя значения текущего времени можно наглядно продемонстрировать модель переправы через реку на лодке, траектория которой определяется сложением векторов скоростей течения реки и собственной скорости лодки.

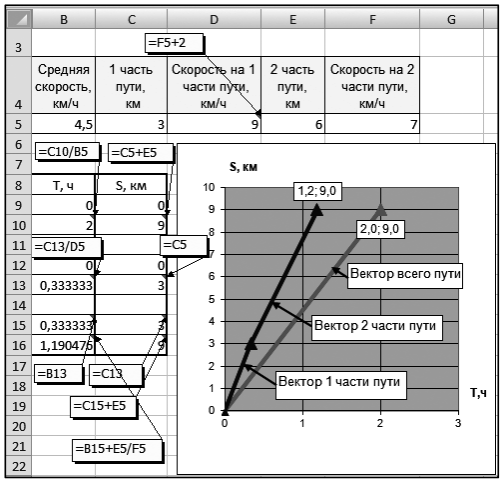
2.1.6. Турист

Турист прошел 3 км по шоссе и 6 км – по проселочной дороге со средней скоростью 4,5 км/ч. По шоссе он шел со скоростью на 2 км/ч больше, чем по проселку. С какой скоростью шел турист по проселочной дороге?

По условию задачи известно, сколько всего прошел турист и его средняя скорость. Следовательно, можем построить вектор в координатах «дальность-время», где начало вектора привязываем к началу координат. Конец вектора определит конечную точку пути туриста в заданных координатах. Далее для первого участка пути строим также вектор, который начинается от начала координат. Аналогично строим вектор для второго участка, но начало его совмещаем с окончанием вектора первого участка. Фактически производим сложение двух векторов. Плохо, что не знаем скорость движения по проселочной дороге. Но ее можно задать. Если эта скорость будет удовлетворять условиям задачи, то конец вектора по второму участку пути совместиться с концом вектора, построенному с учетом средней скорости движения.

В соответствии с условиями задачи формируем ячейки параметров модели. Записываем заголовки параметров и заносим их числовые значения (B5, C5, E5). Если значение отсутствует в задаче или значение нужно найти в ходе решения, то при отсутствии взаимосвязи с другими параметрами записываем любое разумное ее значение (F5). В нашем случае это скорость по проселочной дороге (2 часть пути). Далее формируем зависимые параметры по условиям задачи (D5). Скорость движения на первом участке пути по условию задачи должна быть больше на 2 км, чем на втором участке пути (см. примечание).

После записи параметров и определения их зависимостей приступаем к формированию координат начала и конца векторов движения объектов задачи. Зададим координаты вектора всего пути. Начало вектора помещаем в начало системы координат (0,0). Координата конца вектора по дальности равна сумме дальностей двух участков пути (C10),а координата по времени – частному от деления общей дальности на среднюю скорость (B10). Отступив одну строку, задаем координаты вектора 1-й части пути (B12:C13). Начало вектора помещаем в начало координат (0,0). Координата конца вектора по времени равна частному от деления длины 1-й части пути (C5) на скорость движения (D5), а по дальности – длине 1-й части пути (C5). Начало вектора 2-й части пути совпадает с концом вектора 1-й части пути (B13,C13). Конец вектора зависит от пройденного пути и затраченного времени, которое определяется скоростью (F5).



Теперь можно построить диаграмму Т*очечная диаграмма, на которой значения соединены отрезками*, выделив ячейки с координатами векторов (B9:C16). Если правильно задали размерности параметров, определили их зависимости и задали координаты векторов, то получим верную трактовку задачи в координатах дальность При этом наклон векторов должен определяться скоростью движения туриста и легко проверяется по диаграмме. Целесообразно зафиксировать максимальные и минимальные значения осей координат диаграммы через пункт *Формат оси* контекстного меню соответствующей оси для исключения изменения размеров диаграммы при моделировании.

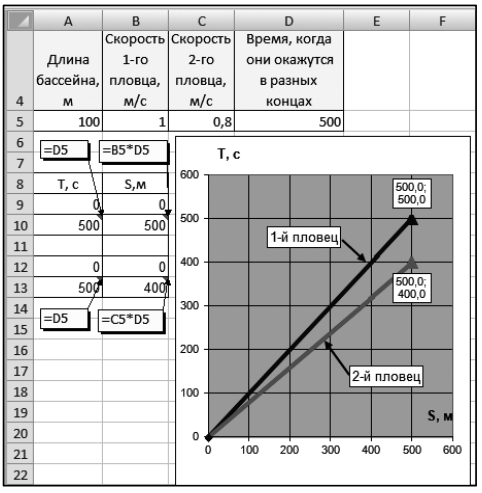
Для наглядности выделим заливкой ячейку названия параметра, который нужно найти по условиям задачи (F4), и ячейку параметра, который

будем использовать при моделировании решения (F5). Изменяя значение скорости на 2-м участке пути, добиваемся совмещения конца вектора всего пути с концом вектора 2-го участка пути. Концы векторов совпадут при скорости 4 км/ч. Для определения момента точного совпадения целесообразно вывести значения координат концов векторов, которые в нашем случае должны стать равными. Для наглядности можно установить маркеры в виде «стрелок» на концах векторов. Для форматирования «стрелок» необходимо навести указатель мыши на точку конца вектора щелкнуть на ней и после появления крестообразного указателя со стрелками вызвать контекстное меню, а затем перейти на пункт *Формат точки данных*. При моделировании следует обратить внимание, что наклон векторов зависит от скорости движения, проекция вектора на ось дальности соответствует пройденному пути, а положение концов относительно оси времени определяют временя движения.

2.1.7. Два пловца

Два пловца стартовали одновременно на длинную дистанцию в стометровом бассейне. Через какое время они первый раз окажутся в разных концах бассейна, если их скорости равны 1 м/с и 0,8м/с.

Составление уравнений для решения этой задачи вызывает определенные трудности. Однако с помощью модели решение находится довольно просто. Нужно сформировать вектора движения для каждого пловца в координатах «время-дальность» и по диаграмме определить момент, когда они будут находиться на противоположных концах бассейна. При этом расстояние между ними должно быть 100м, а проплыть каждый пловец должен расстояние кратное 100м. Формируем начальные параметры модели (A5:D5) по вышеприведенной методике. В ячейку (D5), которую используем при моделировании, будем заносить значение времени.



Далее задаем координаты векторов пловцов. Начало векторов движения обоих пловцов помещаем в начало координат, а координаты концов определяются значением времени (D5) и расстоянием, которое они проплыли, то есть произведением скорости на время (B10, B13).

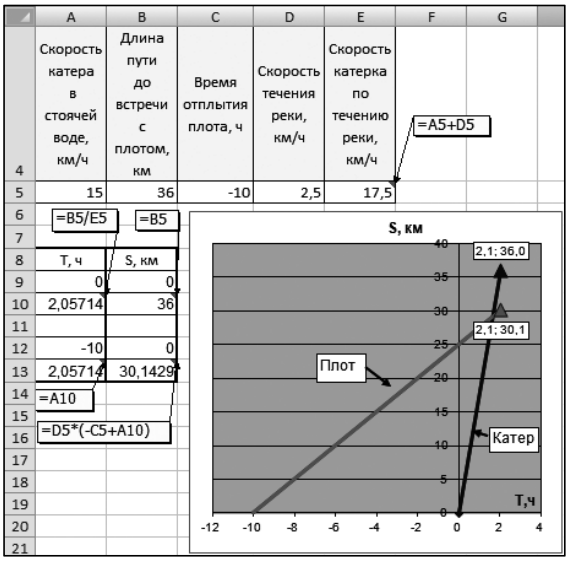
Строим диаграмму и после ее форматирования моделируем движение пловцов, изменяя значение времени в ячейке D5 от 0 до 500с.

Решение находим на 500с после старта пловцов, что хорошо видно на диаграмме. При дальнейшем увеличении значения времени (D5) можно смоделировать нахождение пловцов на различных концах бассейна в другие моменты времени.

2.1.8. Катер и плот

Катер, скорость которого в стоячей воде 15 км/ч, отправился от причала вниз по течению реки и, пройдя 36 км, догнал плот, отправленный от того же причала за 10 часов до отхода катера. Найти скорость течения реки.

По условию задачи катер и плот должны встретиться в конкретный момент времени, то есть концы их векторов движения должны совместиться в результате моделирования. По условиям задачи (A5:E5) формируем координаты векторов по уже известной методике в координатах «время-дальность». Так как катер отправился вниз по течению, то заносим положительное значение скорости течения реки. При этом координата начала вектора плота по времени равна минус 10 часам в связи с отправлением плота на 10 часов раньше катера.



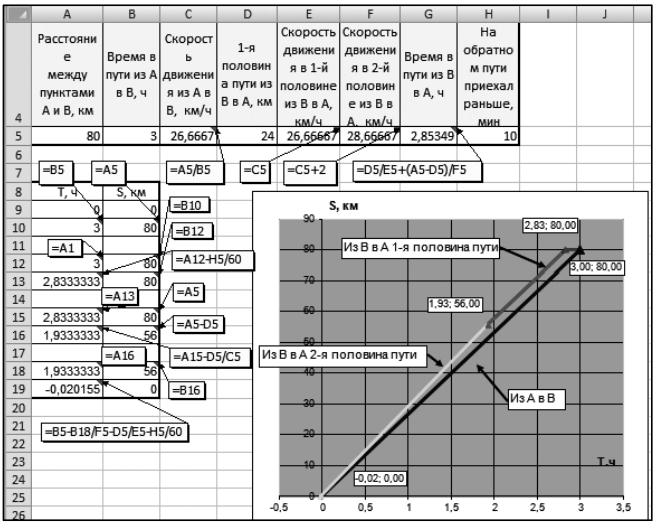
При моделировании изменяем значение скорости течения реки (D5) до совпадения концов векторов объектов.

В момент встречи концы векторов движения катера и плота совпадут при скорости течения реки равной 3км/ч.

2.1.9. Велосипедист

Расстояние между пунктами А и В велосипедист проехал за три часа. Возвращаясь, он первые 24 км ехал с прежней скоростью, а затем увеличил скорость на 2 км/ч и прибыл в пункт А, затратив на обратный путь на 10 минут меньше, чем на путь из А в В. Найти расстояние между пунктами А и В.

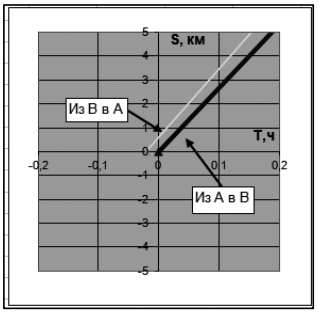
Особенности данной задачи заключаются в наличии большого количества условий и двух решений. Записываем параметры с учетом всех условий задачи (A5:H5). Формируем вектор движения из пункта А в В (A9:B10). Затем задаем вектор, который связан с сокращением времени на обратную дорогу. Этот вектор (A12:B13) параллелен оси времени, так как обеспечивает смещение координаты времени конца вектора, который связан с пунктом В. Далее формируем вектор первых 24 км (A15:B16), а затем вектор прохождения остатка пути с увеличенной скоростью (A18:B19).



Следует заметить , что вектор первого участка на обратном пути параллелен вектору при движении велосипедиста из пункта А в В за счет равенства скоростей движения.

В результате моделирования конец последнего вектора должен совместиться с началом координат, то есть с пунктом А.

На рисунке представлена диаграмма векторов в районе начала координат для более четкой фиксации совпадения векторов. Совмещение концов векторов можно наблюдать по значению ячейки А19. В момент совмещения содержимое ячейки должно равняться нулю.



Нахождение решения осуществляем путем изменения значения ячейки А5, в которой формируется расстояние между пунктами. Это происходит, когда расстояние между пунктами равняется 48 км или 54 км.

При рассмотрении предлагаемого метода моделирования условия задач не изменялись для корректной проверки возможностей метода. Однако для большой наглядности целесообразно модифицировать начальные значения в задаче для более «яркого» графического представления модели решения с помощью векторов движения.

2.1.10. Моделирования касательной в заданной точке функции

В математике много задач связано с вычислением производных функций и использованием геометрической интерпретации первой производной функции. Рассмотрим методику построения касательной к заданной точке функции и определения параметров ее уравнения.

Если задана функция f(x), то первая производная функции в заданной точке с координатами x_0 и y_0 является угловым коэффициентом касательной в этой точке. Уравнение касательной, проходящей через точку с координатами x_0 и y_0, имеет вид:

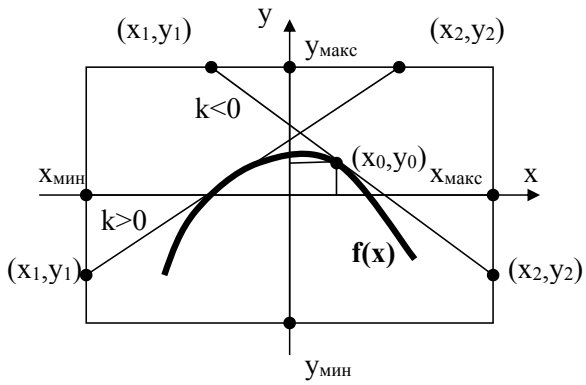
$$y - y_0 = k * (x - x_0),$$

где $k = f'(x_0)$ – угловой коэффициент;

x_0 и y_0 - координаты точки.

Требуется построить график функции $f(x)$ и касательную в прямоугольной области диаграммы, ограниченной минимальными и максимальными значениями абсцисс ($x_{мин}$ и $x_{макс}$) и ординат ($y_{мин}$ и $y_{макс}$).

Так как касательная является прямой линией, проходящей через заданную точку, то ее можно построить по координатам двух точек: (x_1, y_1) и (x_2, y_2), которые располагаются на границах прямоугольника.



Рассмотрим расчет координат точек касательной с учетом значения углового коэффициента $k = f'(x_0)$. Значения углового коэффициента могут быть положительными, отрицательными, нулевыми или иметь бесконечно большие значения.

Пусть $k > 0$. Остановимся на всех возможных вариантах расположения точек. При $x_1 = x_{мин}$ с учетом формулы $y_1 = k * (x_{мин} - x_0) + y_0$.

Если выполняется неравенство $y_{мин} < y_1 < y_{макс}$, то координаты первой точки определены.

В противном случае рассматриваем другие варианты расположение точки. При $y_1 < y_{мин}$ задаем $y_1 = y_{мин}$ и получаем $x_1 = (y_{мин} - y_0 + k * x_0)/k$. При $y_1 > y_{макс}$ задаем $y_1 = y_{макс}$ и вычисляем $x_1 = (y_{макс} - y_0 + k * x_0)/k$.

Координаты второй точки касательной рассчитываются аналогичным образом. При этом задаем $x_2 = x_{\text{макс}}$ и $y_2 = k * (x_{\text{макс}} - x_0) + y_0$.

Если выполняется неравенство $y_{\text{мин}} < y_2 < y_{\text{макс}}$, то координаты второй точки определены.

В противном случае рассматриваем следующие варианты расположения точки.

При $y_2 < y_{\text{мин}}$ задаем $y_2 = y_{\text{мин}}$ и $x_2 = (y_{\text{мин}} - y_0 + k * x_0)/k$.

При выполнении неравенства $y_2 < y_{\text{макс}}$ задаем $y_2 = y_{\text{макс}}$ и подсчитываем $x_2 = (y_{\text{макс}} - y_0 + k * x_0)/k$.

Если $k < 0$, то получаем аналогичные выражения для определения координат точек касательной. При этом всегда в первую очередь рассчитываем правую точку касательной (см. рисунок).

При $k = 0$ касательная является горизонтальной линией относительно оси аргументов. Следовательно, $x_1 = x_{\text{мин}}$, $y_1 = y_0$ и $x_2 = x_{\text{макс}}$, $y_2 = y_0$.

При $k \to \pm\infty$ касательная становиться вертикальной линией. Таким образом, $x_1 = x_0$, $y_1 = y_{\text{мин}}$ и $x_2 = x_0$, $y_2 = y_{\text{макс}}$.

Теперь можно остановиться на методике построения графика функции и касательной в заданной точке функции.

Определяем граничные значения диапазонов для x и y ($x_{\text{мин}}$, $x_{\text{макс}}$, $y_{\text{мин}}$, $y_{\text{макс}}$) при предварительном построении графика заданной функции на основе табулированной функции (B5:C25)

$$y = 2 * \sin x.$$

Формируем таблицы начальных условий задачи.

Задаем ячейки:

координат заданной точки функции(x_0, y_0) (F2 и G2);

углового коэффициента $k = f'(x_0)$ (H2);

граничных значений диапазонов ($x_{\text{мин}}$, $x_{\text{макс}}$, $y_{\text{мин}}$, $y_{\text{макс}}$) (A2:D5).

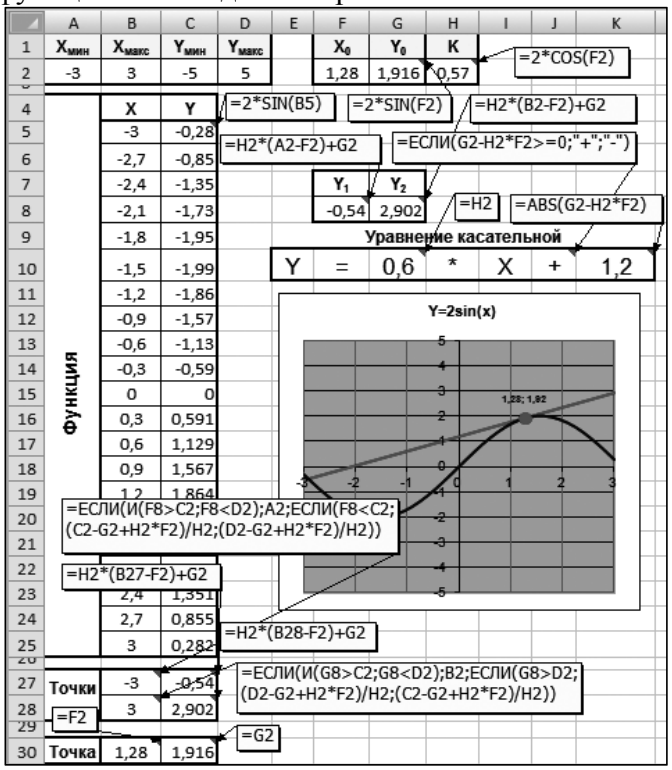
Отступив одну ячейку от таблицы функции (B5:C25), рассчитываем координаты точек касательной (B27:C28) с учетом приведенных выше рассуждений (на рисунке в примечаниях отображены формулы расчетов). Предварительно для удобства вычисляем во вспомогательных ячейках значения y_1 при $x_1 = x_{\text{мин}}$ и y_2 при $x_2 = x_{\text{макс}}$ (F8 и G8). Для отображения точки на графике функции добавляем ячейки с ее заданными координатами (B30 и C30).

Выделив диапазон ячеек (B5:C30), в который включаются ячейки таблицы функции и координат точек касательной, обращаемся к мастеру диаграмм для построения графика функции и касательной. Задаем тип диаграммы - *Точечная*, а вид - *Точечная диаграмма со значениями, соединенными сглаживающими линиями без маркеров*. Фиксируем минималь-

ные и максимальные значения шкал осей x и y в диалоговом окне *Формат оси*.

Для заданной точки, к которой строится касательная, можно установить отличительный маркер и вывести ее координат. Чтобы установить цвет и толщину линии касательной нужно аккуратно щелкнуть по линии касательной. При появлении маркеров точек касательной вызвать контекстное меню и щелкнуть по пункту *Формат точки данных.* В диалоговом окне установить тип, цвет и толщину линии касательной.

Для наглядности одновременно вычисляем параметры уравнения касательной $y = a * x + b$, где из уравнения $a = k$, $b = y_0 - k * x_0$. Эти параметры рассчитываем в соответствующих ячейках, которые соседствуют с ячейками, в которых занесены символы Y,=,* для представления уравнения касательной. Так как значение параметра b могут быть как отрицательными, так и положительными, то в соответствующей ячейке используем функцию ЕСЛИ для отображения символов + и -.



Изменяя значения координаты x_0, осуществляем исследование поведения касательной в заданной точке функции и изменения параметров уравнения касательной.

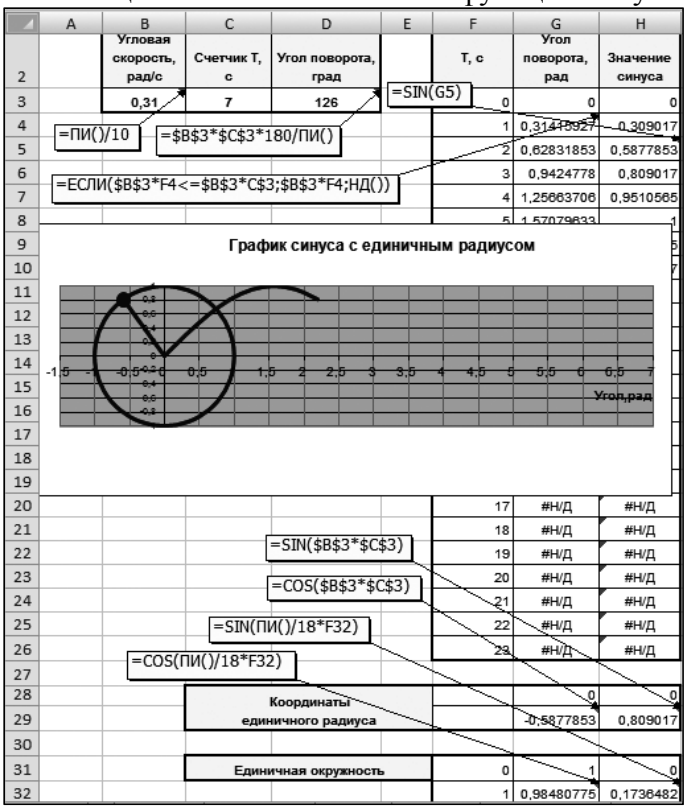
При необходимости можно дополнительно построить график производной заданной функции.

2.1.11. Единичная окружность и синус угла

Для выяснения закономерности изменения основных тригонометрических функций используют единичную окружность и ее радиус-вектор. При этом радиус-вектор в зависимости от угла занимает определенное положение.

Создадим динамическую модель движения единичного радиуса-вектора и построения графика функции синуса. Зададим угловую скорость движения радиуса (B3). Угол поворота радиуса будет определяться произведением заданной угловой скорости на значение счетчика текущего времени (D3). Значение функции синуса равняется проекции радиуса-вектора на ось *Y*.

Исходя из условий, создаем таблицу, где в первом столбце с заданным интервалом заносим время (F3:F26). Во втором столбце (G3:G26) рассчитываем угол поворота при соблюдении условия: угол поворота не превышает угол поворота, соответствующий текущему времени счетчика. При невыполнении условия в ячейки угла записывается функция НД(). В третьем столбце вычисляется значение функции синуса.



Для моделирования движения вектора в дополнительных ячейках исходной таблицы задаем координаты начала радиуса и вычисляем координаты конца радиуса (G28:H29). Не забываем пропустить одну строку

для исключения слияния изображения радиуса с графиком функции синуса.

Отображение единичной окружности производим с помощью дополнительной таблицы, где в первом столбце (F31:F67) задаем количество точек, которое необходимо для нормального изображения окружности. В остальных столбцах вычисляем координаты точек окружности (G31:H67).

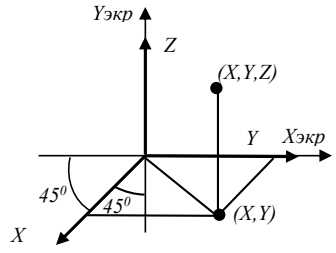
Выделив ячейки столбцов *Угол поворота* и *Значение синуса* таблицы, обращаемся к *Мастеру диаграмм* и после задания параметров переходим к моделированию, изменяя значения счетчика (С3). Ясно, что при построении диаграммы выбираем вид *Точечная диаграмма со значениями, соединенными сглаживающими линиями*. Если единичная окружность на диаграмме получилась деформированной, то нужно растянуть диаграмму по соответствующей стороне.

2.1.12. Построение простейших объемных фигур

Построим каркасные модели элементарных пространственных фигур, которые являются основными элементами трехмерного моделирования.

Задачи стереометрии вызывают определенные трудности. При их решении нужно обладать пространственным воображением.

Возьмем куб, четырехугольную пирамиду, полусферу и цилиндр для объемного изображения во фронтальной диметрической и косоугольной горизонтальной изометрической проекциях. Эти проекции наиболее часто используются. В первую очередь необходимо обеспечить пересчет координат из соответствующей проекции *XYZ* на плоскую поверхность, которую назовем экранной системой координат ХэкрYэкр. Масштабные коэффициенты по осям проекций учитывать не будем.



Пересчет координат из фронтальной диметрической проекции в экранную систему координат осуществляем по формулам (см. рисунок)

$$X_{\text{экр}} = Y - X * \sin 45^0,\ Y_{\text{экр}} = Z - X * \cos 45^0 .$$

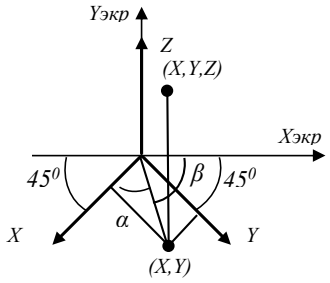
Пересчет из косоугольной горизонтальной изометрической проекции в экранную систему производим по формулам (см. рисунок)

$$X_{\text{экр}} = R * \cos(\beta), Y_{\text{экр}} = Z - R * \sin(\beta),$$

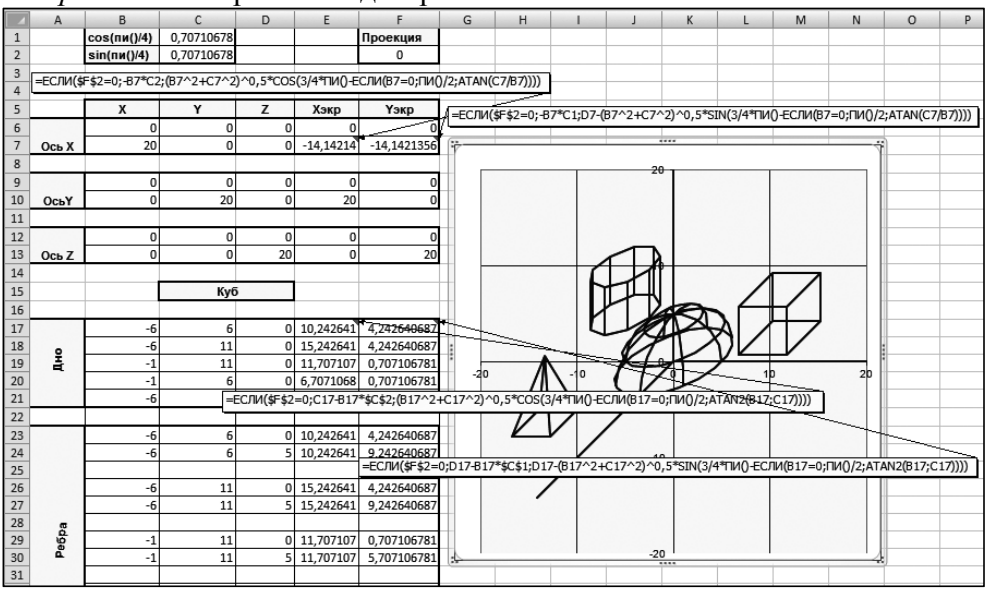
где $R = (X^2 + Y^2)^{0,5}$,

$\beta = 135^0 - arctg(Y/X)$.

Можно легко перейти к другим проекциям задав соответствующие углы ориентации осей проекций, но при этом формулы расчета усложнятся.



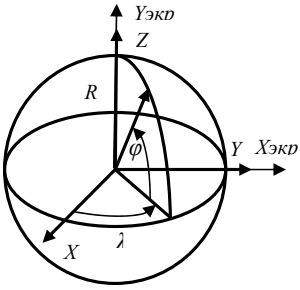
Переход от диметрической к изометрической проекции производится по значению признака «Проекция» в ячейке F2. При нуле фигуры изображаются в диметрической проекции, а при равенстве единице – в изометрической. В ячейках C1 и C2 заранее вычислим значения косинуса и синуса 45^0, которые используются при пересчете координат. Фигуры будем изображать в виде каркасов, задавая координаты концов отрезков или элементарных фигур в трехмерной системе координат *XYZ*. Затем координаты пересчитываются в экранную систему координат по соответствующим формулам в зависимости от значения признака «Проекция». Для отслеживания процесса построения изображения на диаграмме фигур, выделяем ячейки столбцов E и F с достаточным количеством строк, начиная с 6-й строки. Обращаемся к М*астеру диаграмм* и выбираем тип диаграммы *Точечный*.



Изобразим оси проекций через координаты двух точек. Первая точка принадлежит началу координат (B6:D6, B9:D9, B12:D12), а вторая имеет одну соответствующую координату не равную нулю (B7:D7, B10:D10, B13:D13). Значение координаты второй точки оси определяется величиной области построения фигур. Координаты отрезков осей разделяем строками для обеспечения их раздельного изображения. В соответствующих ячейках столбцов E и F вычисляем экранные координаты с проверкой

значения признака «Проекция» и исключением деления на ноль при вычислении функции тангенса (см. примечание). После появления осей на диаграмме приступаем к изображению каркаса куба. Отметим, что основания всех фигур расположены в плоскости XY для простоты задания координат элементов каркаса.

Сначала задаем координаты дна куба (B17:D21), далее координаты концов ребер (B23:D33) и верха куба (B35:D39). Так как целесообразно с помощью автозаполнения заранее обеспечить пересчет координат в экранную систему, то по мере задания координат и формирования пустых сток между элементами куба будем наблюдать процесс появления изображения куба. При расчете угла β используем функцию ATAN2(X;Y), которая определяет угол от $-\pi$ до π.



Аналогично легко получаем изображение четырехугольной призмы. После задания координат дна, определяем координаты высоты призмы и задаем координаты ребер.

При построении каркаса полусферы расположим центр фигуры в начале координат. Тогда для расчета каркаса можно воспользоваться формулами сферы

$$X = R_{\text{сф}} * \cos(\varphi) * \cos(\lambda),\ Y = R_{\text{сф}} * \cos(\varphi) * \sin(\lambda),\ Z = R_{\text{сф}} * \sin(\varphi),$$

где $R_{\text{сф}}$ – радиус сферы,

φ – вспомогательный угол, значения которого изменяются от -90^0 до 90^0,

λ – вспомогательный угол, значения которого изменяются от 0^0 до 360^0.

Радиус полусферы зададим в ячейке H61. Для построения каркаса полусферы воспользуемся «горизонтальными» окружностями при заданных значениях угла $\varphi = 0^0, 30^0$ и 60^0. Расчет координат 19 точек каждой окружности при фиксированном значении φ производим с изменением λ от 0^0 до 360^0 с шагом $\Delta\lambda = 20^0$ (H65). Расчет координат 10 точек «вертикальных» полуокружностей делаем при фик-

	A	B	C	D	E	F	G	H
58		7	-9	0	-13,94975	-4,94974747		
59		9	-7	7	-13,36396	0,636038969		
60								
61			Полусфера			=ПИ()/9	Rсф	5
62							Φ, рад	0
63	0	5	0	0	-3,535534	-3,53553391	Δφ, рад	0,34907
64	1	4,6984631	1,71010072	0	-1,612214	-3,32231512	Λ, рад	0
65	2	3,83022222	3,21393805	0	0,5055619	-2,7083761	ΔΛ, рад	0,34907
66	3	2,5	4,33012702	0	2,5623601	-1,7677669		
67	4	-0,8682409	4,92403877	0	4,3100997	-0,6139390		
68	5	-0,8682409	4,92403877	0	5,5379778	0,61393902		
69	6	-2,5	4,330127					
70	7	-3,8302222	3,21393805	0	5,9223142	2,708376102		
71	8	-4,6984631	1,71010072	0	5,0324158	3,322315122		
72	9	-5	6,1257E-16	0	3,5355339	3,535533906		
73	10	-4,6984631	-1,71010072	0	1,6122144	3,322315122		
74	11	-3,8302222	-3,21393805	0	-0,505562	2,708376102		
75	12	-2,5	-4,33012702	0	-2,56236	1,767766953		
76	13	-0,8682409	-4,92403877	0	-4,3101	0,61393902		
77	14	0,86824089	-4,92403877	0	-5,537978	-0,61393902		
78	15	2,5	-4,33012702	0	-6,097894	-1,76776695		
79	16	3,83022222	-3,21393805	0	-5,922314	-2,7083761	=ПИ()/6	
80	17	4,6984631	-1,71010072	0	-5,032416	-3,32231512		
81	18	5	-1,2251E-15	0	-3,535534	-3,53553391		
82							Φ, рад	0,5236
83	0	4,33012702	0	2,5	-3,061862	-0,56186218	Δφ, рад	0,5236
84	1	4,06898841	1,48099066	2,5	-1,396219	-0,37720929	Λ, рад	0
85	2	3,31706974	2,783352	2,5	0,4378295	0,154477493	ΔΛ, рад	0,34907
86	3	2,16506351	3,75	2,5	2,2190689	0,969068911		
87	4	0,75191867	4,26434266	2,5	3,7326559	1,968313212		
88	5	-0,7519187	4,26434266	2,5	4,7960294	3,031686788		

=$H$61*COS($H$62)*COS(A63*$H$65)
=$H$61*COS($H$62)*SIN(A63*$H$65)
=$H$61*SIN($H$62)

сированных значениях λ =45^0, 90^0, 135^0 и 180^0, но с изменением значений угла φ от от 0^0 до 180^0 с шагом $\Delta\varphi$ =20^0. Не забываем о разделении ячеек с координатами отдельных элементов каркаса полусферы пустыми строками.

Теперь остается изобразить каркас цилиндра. Центр основания цилиндра имеет координаты X_0 и Y_0 (H168:H169). Радиус цилиндра $R_ц$задан в ячейке H167. Каркас цилиндра составим из окружностей основания (B169:D178) и верха (B180:D189), а также образующих 9 ребер. Окружности изображаем 10 точками с использованием формул для сферы, но с учетом координат центра (H168:H169). При задании координат окружности верха цилиндра координаты *X* и *Y* копируем, а координату *Z* приравняем высоте цилиндра (H171). Координаты концов ребер формируем копированием соответствующих координат окружностей дна и верха цилиндра.

	A	B	C	D	E	F	G	H
166								
167			Цилиндр				Rц	3
168							X_0	-7
169	0	-4	-10	0	-7,171573	2,828427125	Y_0	-10
170	1	-4,7018667	-8,07163717	0	-4,746915	3,324721807	Δφ, рад	0,69813
171	2	-6,4790555	-7,04557674	0	-2,464193	4,581384056	Hц	5
172	3	-8,5	-7,40192379	0	-1,391516	6,01040764		
173	4	-9,8190779	-8,97393957	0	-2,030803	6,943136541		
174	5	-9,8190779	-11,0260604	0	-4,082924	6,943136541		
175	6	-8,5	-12,5980762	0	-6,587669	6,01040764		
176	7	-6,4790555	-12,9544			6		
177	8	-4,7018667	-11,9283628	0	-8,603641	3,324721807		
178	9	-4	-10	0	-7,171573	2,828427125		
179								
180		-4	-10	5	-7,171573	7,828427125		
181		-4,7018667	-8,07163717	5	-4,746915	8,324721807		
182		-6,4	-7,045576	5	-2,464193	9,581384056		
183		-8,5	-7,40192379	5	-1,391516	11,01040764		
184		-9,8190779	-8,97393957	5	-2,030803	11,94313654		
185	Верх	-9,8190779	-11,0260604	5	-4,082924	11,94313654		
186		-8,5	-12,5980762	5	-6,587669	11,01040764		
187		-6,4790555	-12,9544233	5	-8,373039	9,581384056		
188		-4,7018667	-11,9283628	5	-8,603641	8,324721807		
189		-4	-10	5	-7,171573	7,828427125		
190								
191		-4	-10	0	-7,171573	2,828427125		
192	Ребра			5	-7,171573	7,828427125		
193								
194		-4,7018667	-8,07163717	0	-4,746915	3,324721807		
195		-4,7018667	-8,07163717	5	-4,746915	8,324721807		
196								
197		-6,4790555	-7,04557674	0	-2,464193	4,581384056		

=$H$168+$H$167*COS(A169*$H$170)

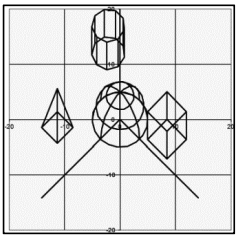
=$H$169+$H$167*SIN(A169*$H$170)

=2*ПИ()/9

=B169 =C169 =$H$171

=B169 =C169 =D169

При равенстве единице признака «Проекция» получаем изображение фигур в изометрии. Это позволит определиться с наглядностью изображения фигур в каждой проекции и принять решение об их использовании при трехмерном моделировании. При задании координат каркаса фигур целесообразно их привязать к одной характерной точке фигуры для облегчения моделирования перемещения. Для расширения возможностей по моделированию



объемного изображения фигур можно ввести масштабные коэффициенты по координатным осям.

2.1.13. Решение графоаналитическим методом задач линейного программирования

Создадим графическую модель решения оптимальной задачи.

Задача линейного программирования состоит в нахождении наибольшего или наименьшего значения линейной целевой функции

$$F = C_1 * X_1 + \cdots + C_n * X_n$$

в области n-мерного пространства, заданной посредством системы неравенств:

$$a_{i1} * X_1 + a_{i2} * X_2 + \cdots + a_{in} * X_n \le b_i;\ i=1,2,\ldots,m;$$
$$X_1 \ge 0;\ X_2 \ge 0;\ \ldots;\ X_n \ge 0.$$

Рассмотрим модель решения задачи графоаналитическим способом при n=2. Сущность графоаналитического метода заключается в нахождении совместных точек области допустимых решений с линией уровня целевой функции. Для этого необходимо построить область решений и обеспечить перемещение на плоскости линии уровня целевой функции до получения совместных точек.

Пусть область допустимых решений имеет форму треугольника (m=3). Это значительно облегчит ее графическое изображение по координатам вершин треугольника с помощью *Мастера диаграмм*.

Итак, целевая функция

$$F = 2 * X_1 + X_2\ ,$$

а область допустимых решений задана неравенствами:

$$3 * X_1 + 4 * X_2 \ge 15;\ -5 * X_1 + 6 * X_2 \ge -20;\ X_1 + 8 * X_2 \le 30.$$

Обычно из условий большинства задач вводят следующие дополнительные ограничения:

$$X_1 \ge 0;\ X_2 \ge 0.$$

Требуется найти минимальное значение целевой функции F при заданных ограничениях.

*Вариант 1.*По условию задачи область решений полностью находится в первом квадранте системы координат. Заменяя в неравенствах знаки больше и меньше на знак равенства, получаем уравнения прямых границ области допустимых решений:

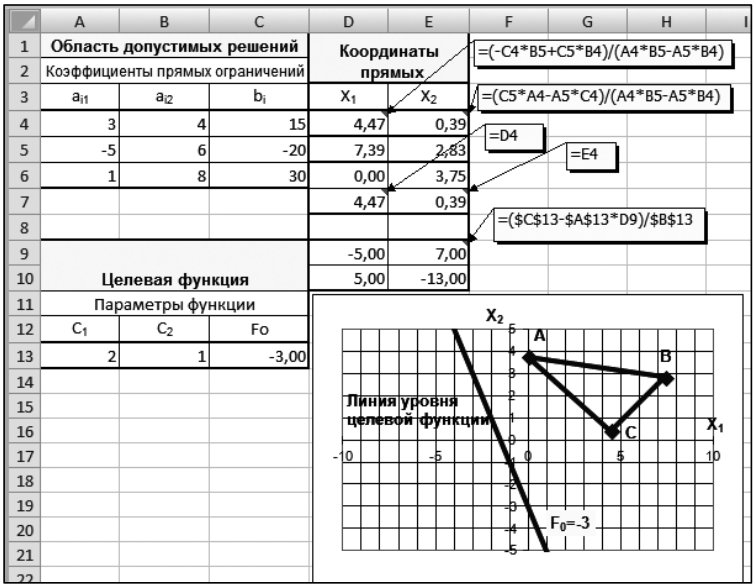
$$3 * X_1 + 4 * X_2 = 15;\ -5 * X_1 + 6 * X_2 = -20;\ X_1 + 8 * X_2 = 30.$$

Для построения границ области достаточно определить координаты вершин треугольной области допустимых решений. Группируем попарно уравнения и решаем системы с помощью формул Крамера для нахождения координат пересечения прямых.

Например, для системы уравнений в общем виде $a_{11} * X_1 + a_{12} * X_2 = b_1$ и $a_{21} * X_1 + a_{22} * X_2 = b_2$ решения получаем по формулам $X_1 = (b_1 * a_{22} - b_2 * a_{12})/(a_{11} * a_{22} - a_{21} * a_{12)})$ и $X_2 = (b_2 * a_{11} - b_1 * a_{21})/(a_{11} * a_{22} - a_{21} * a_{12)})$.

В таблицу (A4:C6) заносим коэффициенты прямых. В таблице (D4:E6) вычисляем координаты точек пересечения первой и второй прямых, второй и третьей, третьей и первой. Для замыкания контура треугольника на графике в конце перезаписываем координаты первой расчетной точки пересечения прямых прямой (D7,E7).

Перед обращением к *Мастеру диаграмм* выделяем диапазон ячеек D4:E10 с учетом дальнейшего построения линии уровня целевой функции. Выбираем тип диаграммы *Точечная*, а вид, где значения соединены отрезками. Чтобы графики области и линии целевой функции были отображены раздельно ячейки с их координатами отделяем строкой. Фиксируем максимальные и минимальные значения шкал через диалоговое окно *Формат оси* для исключения изменений параметров области построения диаграммы при моделировании.



Для построения линии уровня целевой функции задаем начальное значение функции F_0 (C13). Записав значения коэффициентов C_1 и C_2 целевой функции и ее начальное значение, можно по заданному значению X_1 вычислить соответствующие координаты X_2 с помощью формулы $X_2 = (F_0 - C_1 * X_1)/C_2$ для двух концов линии уровня.

Определив координаты концов линии уровня (D9:E10), получаем изображение линии на графике. Выборное значение X_1 можно откорректировать для обеспечения более наглядного расположения линии уровня относительно области допустимых решений. Теперь остается изменять значение F_0 для моделирования перемещения линии уровня целевой функции в сторону области решения. При этом линия уровня целевой

функции будет смещаться параллельно начальному положению. Смещение производится до первого соприкосновения с областью допустимых решений. В нашем случае минимальное значение целевой функции равно 3,75 при $X_1 = 0$ и X_2=3,75 (точка А). Если нужно найти максимальное значение целевой функции, то оно равняется 17,6 при X_1=7,39 и X_2=2,82 (точка В). Такой результат получаем при дальнейшем увеличении значения целевой функции (С13), когда ее линия соприкасается с последней точкой области решений при выходе из нее. При моделировании прямая может выйти из заданного диапазона графика. В этом случае корректируем значения координат концов линии по оси X_1до появления прямой на графике.

Если в первом уравнении прямой коэффициент a_{11}приравнять к 8, то моделируется случай, когда линия целевой функции соприкасается не с одной точкой, а со стороной АС области решений. При этом решение задачи определяется не точкой, а диапазоном значений $0 \le X_1 \le 2{,}5$ и $-1{,}25 \le X_2 \le 3{,}75$ при значении целевой функции 3,75. При желании можно автоматизировать изменение значения F_0 (С13).

Вариант 2. Следует отметить, что если область допустимых решений имеет больше трех сторон, то при рассмотренном подходе нужен дополнительный алгоритм распределения точек вершин многоугольника в соответствии с их последовательным расположением, получаемым при определенном направлении обхода вершин многоугольника.

Так как алгоритм построения многоугольника области допустимых решений по координатам его вершин в общем случае сложен, то целесообразно область решений определять графически после построения ее линий сторон на графике.

Первоначально необходимо определить диапазон координат области ограничений. После задания координат прямых по X_1 можно вычислить значения координат по X_2 по формуле $X_2 = (b_i - a_{i1} * X_1)/a_{i2}$ для двух концов каждой линии границ.

Отобразив на графике прямые с учетом знаков ограничений, окантовы-

ваем область допустимых решений с помощью инструмента *Полилиния* с панели *Рисования* и определяем цвет заливки.

Далее можно заняться отображением линии уровня целевой функции. В первом варианте решения задачи координаты X_1 концов линий оставались при моделировании фиксированными. Хотя иногда необходимо проводить корректировку значений координат концов линии для исключения ее выхода из области построения графика. Это может привести к искажению геометрического смысла при моделировании поиска решения графоаналитическим методом. При изменении значения уровня целевой функции ее линия перемещается по вектору $C = (C_1, C_2)$. Для обеспечения параллельного смещения ограниченной по длине линии целевой функции относительно начального положения и фиксации ее длины нужно вычислить изменения координат ее концов с учетом координат вектора *C*.

Фиксируем начальное положение линии целевой функции для наблюдения динамики движения линии к области решения при заданном F_0. Координаты концов движущейся линии уровня вычисляем по формулам $X_1 = X_{01} + K * C_1$ и $X_2 = X_{02} + K * C_2$. Целевую функцию в этом случае можно записать как $F_1 = C_1 * (X_{01} + K * C_1) + C_2 * (X_{02} + K * C_2)$, где F_1 есть новое значение уровня целевой функции. Значения X_{01} и X_{02} определяют начальное положение линии при F_0.

Из последнего выражение находим коэффициент $K = (F_0 - F_1)/(C_1^2 + C_2^2)$ и вычисляем координаты концов линии целевой функции. При соприкосновении линии с областью находим решение задачи в точке А.

То есть, получаем значение целевой функции и значения X_1 и X_2.

2.2. Физика

2.2.1. Броуновское движение частиц

В природе часто можно наблюдать «хаотические» движения различных тел. Создадим модель броуновского движения частиц на плоскости. Пусть все частицы в начальный момент находятся в одной центральной точке квадратной области на плоскости.

Наибольшей неопределенностью характеризуется равномерный закон распределения. Будем считать, что движение частиц осуществляются по такому закону с заданным максимальным шагом по любой из координат. Тогда изменения координат частицы на i-м шаге можно рассчитать по формулам $dX_i = P_i * dL$ и $dY_i = P_i * dL$, а координаты по выражениям $X_i = X_{i-1} + dX_i$ и $Y_i = Y_{i-1} + dY_i$, где P_i - случайное число в интервале от -0,5 до +0,5. Значение шага определяется текущим значением счетчика времени.

Сделаем модель для 3-хчастиц. Количество шагов частиц должно быть не меньше 10. Для расчета координат частиц создадим 3-и таблицы, а для формирования маркера текущего положения частицы выделим отдельные ячейки для координат их текущего положения.

Для формирования случайных чисел используем функцию СЛЧИС(). Для получения чисел как положительных, так и отрицательных произведем центрирование случайных чисел относительно значения 0,5. Диапазон изменения чисел зависит от значения шага dL. Датчик должен выдавать числа от $-dL/2$ до $+dL/2$.

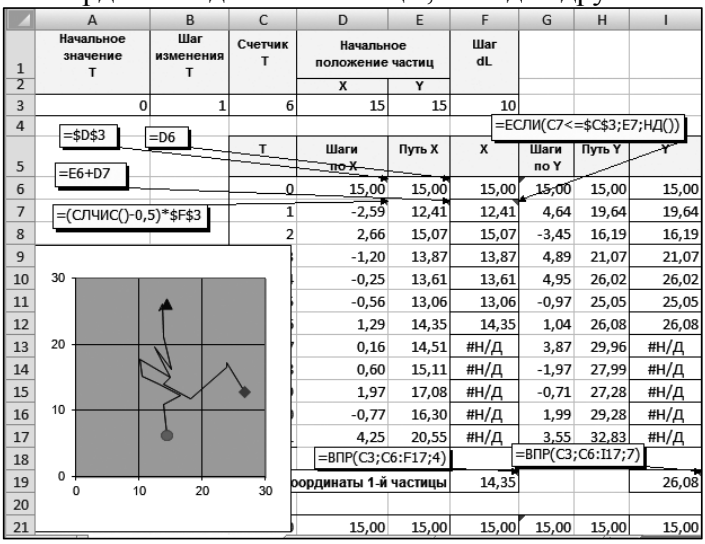
Создадим ячейки для параметров модели (A3:F3). Пусть все параметры будут безразмерными. Для каждой частицы создаем свои таблицы для построения их траекторий движения.

В первый столбец таблицы для 1-й частицы (C6:I17) занесем значение времени с шагом изменения времени от 0 до 11.

В следующем столбце *Шаги по X* (D6:D17) производим расчет шагов по координате *X* с помощью функции СЛЧИС().Далее формируем столбец *Путь X* (E6:E17), где рассчитываем координаты частиц на заданный интервал времени путем прибавления к предыдущей координате величины шага на текущей момент. Теперь остается сформировать данные по траекториям на заданный момент времени, определяемый *Счетчиком T* (C3). Если значение времени в таблице больше текущего, то формируется функция НД(). Это исключает изображение траекторий частицы, которые соответствуют не наступившим моментам времени.

Для отображения положения частицы в дополнительной ячейке *Координаты 1-й частицы* (F19 I19) выбираем ее значение на текущий момент времени с помощью функции *ВПР()* (см. примечание). По этим ко-

ординатам формируем маркеры частиц. Все это повторяется не только для расчета координат *Y* для 1-й частицы, но и для других частиц.



	A	B	C	D	E	F	G	H	I
1	Начальное значение T	Шаг изменения T	Счетчик T	Начальное положение частиц		Шаг dL			
2				X	Y				
3	0	1	6	15	15	10			
4									
5			T	Шаги по X	Путь X	X	Шаги по Y	Путь Y	Y
6			0	15,00	15,00	15,00	15,00	15,00	15,00
7			1	-2,59	12,41	12,41	4,64	19,64	19,64
8			2	2,66	15,07	15,07	-3,45	16,19	16,19
9				-1,20	13,87	13,87	4,89	21,07	21,07
10				-0,25	13,61	13,61	4,95	26,02	26,02
11				-0,56	13,06	13,06	-0,97	25,05	25,05
12				1,29	14,35	14,35	1,04	26,08	26,08
13				0,16	14,51	#Н/Д	3,87	29,96	#Н/Д
14				0,60	15,11	#Н/Д	-1,97	27,99	#Н/Д
15				1,97	17,08	#Н/Д	-0,71	27,28	#Н/Д
16				-0,77	16,30	#Н/Д	1,99	29,28	#Н/Д
17				4,25	20,55	#Н/Д	3,55	32,83	#Н/Д
18									
19				оординаты 1-й частицы		14,35			26,08
20									
21				15,00	15,00	15,00	15,00	15,00	15,00

Для построения траекторий выделяем столбцы *X* и *Y* в таблицах (F6:F49 и I6:I49) для всех 3-х частиц с использованием групповой клавиши *Ctrl*. Выбираем диаграмму *Точечная с прямыми отрезками*. Координаты текущего положения частиц форматируем различными маркерами.

Для автоматизации процесса моделирования создаем кнопки для установки счетчика в начальное положение и начисления значения счетчика на определенную величину.

Создадим кнопку *Начальное значение* с функцией Chn_Click с программой:

```
Private Sub Chn_Click()
Cells(3,3) = Cells(3,1)
End Sub.
```

Аналогично создается кнопка *Счетчик* со следующей программой:

```
Private Sub Chet_Click()
x1 = Cells(3,3)
x2 = Cells(3,2)
Cells(3,3) = x1 + x2
End Sub.
```

При необходимости можно увеличить размеры области, количество частиц и шагов. Однако нужно учитывать размер шага движения частиц.

2.2.2. Траектория полета брошенного тела

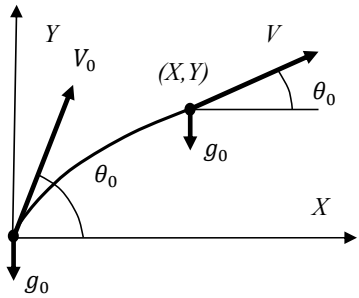
Необходимо рассчитать начальные условия бросания тела при попадании в мишень. Начальное значение скорости V_0 задано. Мишень находится на расстоянии $X_ц$ и имеет высоту $Y_ц$.

Пусть V_0= 300м, $X_ц$= 300м и $Y_ц$=1,5м.

При расчетах движения тела для малых дальностей ускорение силы тяжести может быть принято постоянной по величине и направлению (см. рисунок). Тогда без учета сопротивления воздуха уравнение движения центра масс тела в однородном поле силы тяжести рассчитываются по формулами

$$X = (V_0 * \cos(\theta_0)) * T,$$

$$Y = (V_0 * \sin(\theta_0)) * T - g_0 * T^2/2,$$



где X - дальность полета тела;

Y - высота тела;

V_0- начальная скорость;

θ_0- угол бросания;

g_0=9,81 м/с^2 ускорение свободного падения,

T - время.

Факт попадания в мишень можно определить математически из выражения $0 < Y < Y_ц$ при $X = X_ц$, где X вычисляется по выше представленной формуле.

Однако только в случае, когда $Y = 0$, получаем простое выражение для определения угла бросания θ_0 при заданной начальной скорости V_0

$$\theta_0 = 0{,}5 * \arcsin(g_0 * X_ц / V_0^2).$$

Такая ситуация характерна для поражения мишени на нисходящем участке траектории при попадании тела в основание мишени. Вычисленное значение первоначального угла бросания θ_0 при $Y = 0$ используем как первичное для дальнейшего поиска начальных условий бросания для попадания в центр мишени при последовательном увеличении значения угла θ_0, которое заносим в ячейку под названием *Угол бросания*. Факт попадания определяем по пересечению траектории тела с изображением мишени. Но прежде чем начать моделировать бросание тела в мишень, нужно сначала оценить возможность ее поражения по условию $X_ц < X_{макс}$.

Начинаем с создания таблиц начальных условий бросания (B2, C2), параметров мишени (B22:C25), а также координат траектории тела (B4:C14). Количество ячеек в таблице координат траектории тела определяется числом расчетных точек, нужных для корректного изображения траектории.

После оценки возможности поражения мишени необходимо выбрать интервал изменения координаты дальности X для изображения траектории тела. Для графического построения гладких кривых достаточно десяти точек. В нашей задаче присутствует мишень, которую желательно также изобразить для визуального определения попадания. Но размеры мишени

гораздо меньше дальности бросания. Это несоответствие порождает определенные трудности в визуализации модели.

Величину интервала изменения дальности бросания возьмем 50м для обеспечения отображения траектории тела. Расчет координат траектории производим по вышеприведенным формулам с учетом данных в таблицах начальных условий.

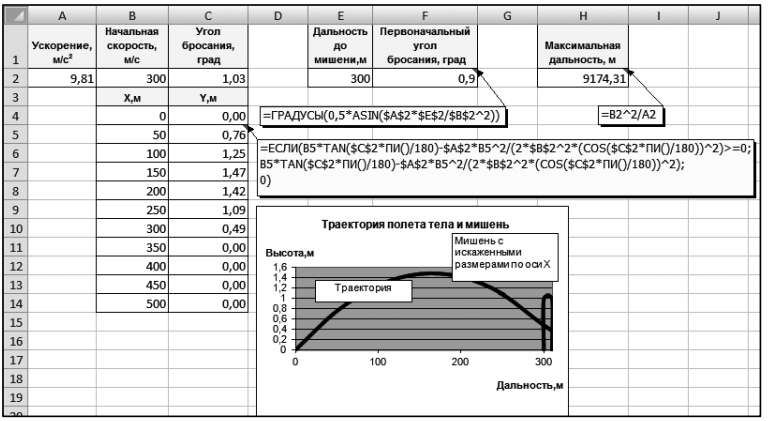
Для исключения отображения на графике участков траекторий с отрицательными высотами, при их расчете желательно использовать стандартную функцию ЕСЛИ(), где в случае отрицательной высоты ее значение приравниваем к нулю: ЕСЛИ(Y>=0;Y;0).

Для построения графика выбираем тип диаграммы *Точечная с гладкими кривыми.* Отобразить мишень на графике можно двумя способами. Наиболее простой способ заключается в изображении цели средств рисования в группе *Иллюстрации*. Однако затруднения могут возникнуть с определением точного местонахождения мишени и правильным выбором масштаба ее изображения на графике. Последние трудности можно обойти, задавая параметры мишени в виде таблицы координат характерных точек ее контура в плоскости полета. Так, для изображения вертикальной мишени нужны только две координаты: $X = X_{\text{ц}}$, $Y = 0$ и $X = X_{\text{ц}}$, $Y = H_{\text{ц}}$, где - $H_{\text{ц}}$ высота мишени. Для изображения мишени в виде контура прямоугольника задаем пять координат вершин, последовательно обходя контур мишени:

$X = X_{\text{ц}}, Y = 0; X = X_{\text{ц}}, Y = H_{\text{ц}}; \quad X = X_{\text{ц}} + L, Y = H_{\text{ц}}; X = X_{\text{ц}} + L, Y = 0; X = X_{\text{ц}}$,Y=0,

где L - длина мишени.

При этом для графического совместного изображения мишени и траектории тела подходит только тип графика *Точечный.* Таблицы координат траектории и параметров мишени должны быть разделены пустой строкой.



Интересно заблаговременно выделить диапазон ячеек (B4:C25), где должны быть координаты траектории и контура мишени. При этом можно наблюдать построение изображений контура мишени и траектории по мере заполнения таблиц данными. На рисунке изображен контур мишени в виде прямоугольника.

Используя таблицы параметров для изображения мишени, можно без затруднений изменять положение мишени на графике.

Для удобства отображения мишени в неискаженном виде из-за разных масштабов осей координат целесообразно построить отдельный график в области мишени с жестким определением максимальных и минимальных значений. Параметры шкал должны соответствовать координатам расположения мишени и ее размерам.

Из графиков видно, что в мишень на дальности 300м попадает тело на высоте 0,5м при угле бросания 1,03 град.

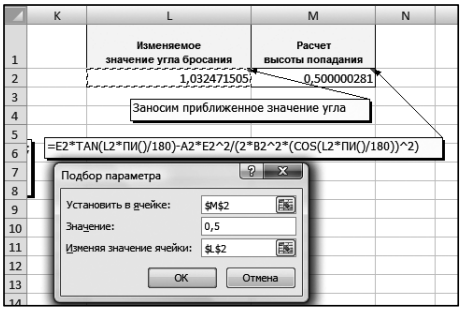
Угол бросания тела можно не подбирать вручную для обеспечения попадания в мишень, а вычислить с помощью процедуры *Подбора параметра*. Процедура находится в списке *Анали «что-если»* группы *Работа с данными* на вкладке *Данные*.

Высота точки попадания $H_п$ в мишень вычисляется по выражению

$$H_п = X_ц * tg(\theta_0) - g_0 * X_ц^2/(2 * V_0^2 * cos^2(\theta_0)) .$$

Используя процедуру *Подбор параметра* при заданных значениях $X_ц$, $H_п$ и V_0 можно найти значение θ_0, которое используется как значение для угла бросания тела. Вычислим θ_0 для $H_п = 0{,}5$ м при заданных условиях задачи.

При применении процедуры *Подбор параметра* в качестве начального значения θ_0 можно взять значение угла бросания, рассчитанное при $H_ц = 0$ (ячейка *Первоначальный угол бросания*). Это значение заносим в ячейку *Изменяемое значение угла бросания*.



Во вторую ячейку *Расчет высоты попадания* заносим правую часть выражения расчета высоты точки попадания со ссылками на заданные начальные значения и переводом значений углов из градусов в радианы. Вызываем процедуру *Подбор пара-*

метра и в соответствующие поля заносим ссылки на ячейки L2 и M2. В окно *Значение* заносим необходимое значение высоты точки попадания в мишень – 0,5м.

После нажатия на кнопку *ОК* получаем значение угла θ_0, которое равняется 1,03 град. Полученный результат совпадает с решением задачи при ручном подборе угла бросания, значение которого находится в ячейке C2. Это значение угла можно занести в ячейку *Угол бросания* и убедится в поражении мишени на заданной высоте.

2.2.3. Оценка эффективности бросаний тела по мишени

Определить число попаданий тела в мишень с учетом случайных ошибок прицеливания при бросании.

Для выбора количества бросаний тела будем исходить из того, что десяти бросаний будет достаточно для первоначальных оценок. Рассеивание бросаний тела будем определять предельными ошибками угла бросания ($\Delta\theta_0$) и углом отклонения от плоскости бросания ($\Delta\propto$), которые в нашем случае подчиняются равномерному закону. Траектория полета тела в плоскости стрельбы рассчитывается по вышеприведенным выражениям с использованием начальных параметров стрельбы (V_0, θ_0). Угол бросания для каждого бросания определяется с учетом случайной ошибки. Отклонение от плоскости бросания ($Z_{\text{п}}$) на дальности цели ($X_{\text{ц}}$) при малом угле α можно рассчитать по выражению

$$Z_{\text{п}} = X_{\text{ц}} * \text{tg}(\propto).$$

Генератор равномерного распределения чисел выдает их в диапазоне то 0 до 1. Однако, определение ошибок нужно производить с учетом как положительных, так и отрицательных ошибок.

Математическое отклонение для α равняется 0, а для угла бросания - θ_0. Таким образом, расчет значения углов с ошибками производится по следующим формулам

$$\theta = \theta_0 + (\text{СЛЧИС}() - 0{,}5) * \Delta\theta_0 \text{ и } \propto = (\text{СЛЧИС}() - 0{,}5) * \Delta\propto.$$

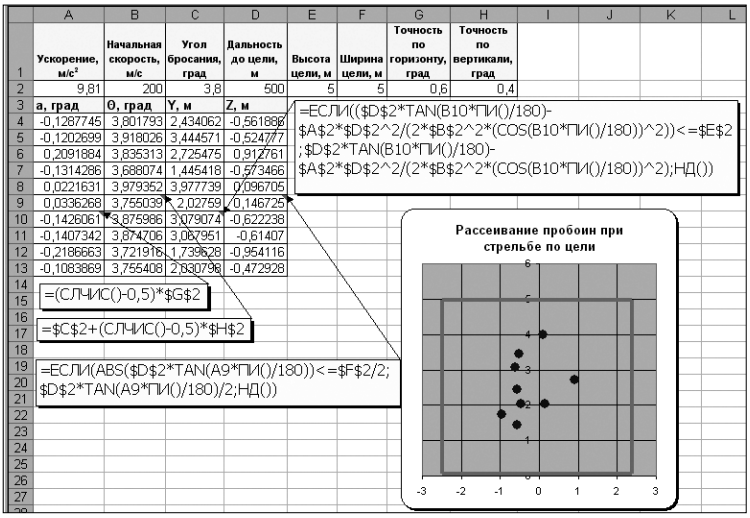
Попадания в мишень проверяются с учетом размеров мишени по высоте ($\text{Н}_{\text{ц}}$) и ширине ($Z_{\text{ц}}$). Для проверки попадания на дальности $X_{\text{ц}}$ по соответствующим координатам используются следующие выражения ЕСЛИ(Y=< Нц;Y; НД()) и ЕСЛИ(ABS(Zп)=< Zц ; Zп; НД()).

В случае промаха в соответствующую ячейку записывается функция НД(). При проверке по координате Z используем функцию ABS(). Так как отклонения от оси симметрии мишени могут быть как положительными, так и отрицательными.

Для построения попаданий в мишень выделяем массив координат попаданий (C4:D10). Для естественного представления попаданий нужно выбрать вид графика, где сравниваются пары значений. Контур цели с

учетом размеров можно изобразить на графике с помощью процедур рисования или задать таблицей координат мишени.

Целесообразно, как и в предыдущей модели, сначала рассчитать начальные условия бросания по условию $Y = 0$ при $X = X_{\text{ц}}$, а затем скорректировать угол бросания путем его увеличения. Корректировку условий бросаний можно закончить при появлении устойчивого количества пробоин в контуре мишени. При каждом пересчете данных на листе приложения Excel датчик случайных чисел будет выдавать новую последовательность, а в контуре мишени будут появляться новые отметки о попадании.

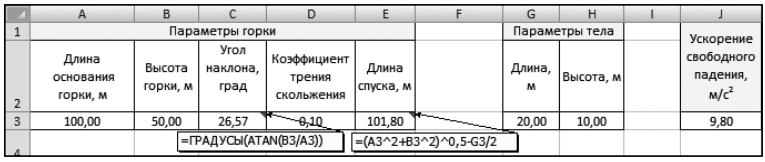


Пересчет можно иногда инициировать простым щелчком на кнопке *Сохранить*. В каждой серии бросаний по количеству попаданий ($N_{\text{п}}$) можно вычислять их частоту $F = N_{\text{п}}/10$. В нашем случае вероятность попаданий в мишень равна 0,9.

2.2.4. Движение тела по наклонной горке

Тело прямоугольной формы в виде бруска соскальзывает с вершины горки. Известны высота и длина основания горки, размеры тела, а также коэффициент трения скольжения.

Создадим модель движения тела по горке. Зададим параметры горки



(A3,B3), тела (G3,H3) и занесем значение ускорения свободного падения (J3). Вычислим значения угла наклона горки (C3) через тангенс как отношение катетов (см. примечание). Длину спуска (E3) рассчитываем по

формуле Пифагора, но вычитаем из нее половину длины тела. Так как будем считать, что движение тела заканчивается при соприкосновении передней стенки с уровнем основания горки.

Известно, что при движении скольжения тела приобретает ускорение, значение которого вычисляется по формуле

$$a = g * (\sin(\propto) - k_c * \cos(\propto)),$$

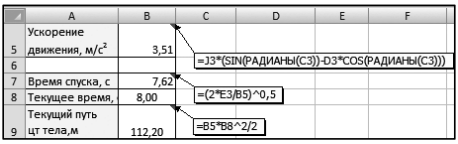
где a – ускорение тела при спуске,

g – ускорение свободного падения,

$\propto$ - угол наклона горки,

k_c- коэффициент трения скольжения.

Следует отметить, что масса или вес тела отсутствует в формуле. Однако при решении задач по определению коэффициента трения при известных параметрах движения вес тела может понадобиться.



При условии, что начальная скорость движения тела равна нулю вычисляем время спуска (B7) по формуле

$$t_{\text{сп}} = \sqrt[2]{2 * L_{\text{сп}}/a}\ ,$$

где $t_{\text{сп}}$ - время спуска,

$L_{\text{сп}}$ - длина спуска,

a – ускорение тела при спуске (B5).

В ячейке B8 задаем текущее время, которое используем для расчета смещения центра тяжести тела (B9) по формуле

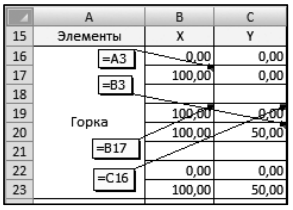
$$S_{\text{цт}} = a * t^2/2\ ,$$

где $S_{\text{цт}}$ - текущий путь центра тяжести тела,

a – ускорение тела при спуске,

t - текущее время.

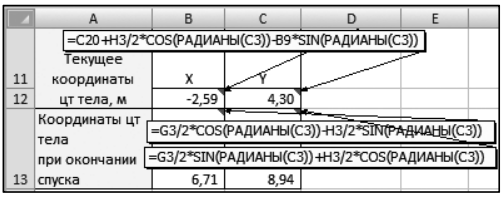
Основные параметры для моделирования движения тела подготовлены. Приступим к созданию изображений горки и тела с помощью диаграммы. Изображение горки определяются координатами трех точек треугольника горки. Конечную точку спуска горки привяжем к началу координат (B16 и C16). Координаты других точек зависят от заданных размеров (B17 и C20). Остальные координаты задаем копированием (B19, C19 и т.п.).



Такой подход позволит автоматически корректировать изображение горки при изменении ее параметров.

Для изображения текущего положения тела в первую очередь необходимо определить координаты его центра тяжести (B12 и C12).

С целью упрощения будем считать, что в начальном положении тело расположено на горке так, что проекция точки центра тяжести на наклонную поверхность горки совпадает с вершиной горки.



То есть, на поверхности горки находиться передняя половина тела. В этом случае координаты центра тела $(x^0_{\text{цт}}, y^0_{\text{цт}})$ в начале движения вычисляются по формулам (см. рисунок)

$$x^0_{\text{цт}} = x_0 - h/2 * \sin(\propto),\ y^0_{\text{цт}} = y_0 + h/2 * \cos(\propto),$$

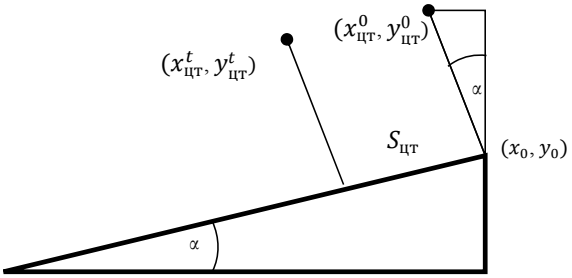
где x_0, y_0 – координаты вершины горки,

h - высота тела.

Для вычисления текущих значений координат центра тяжести $(\mathrm{x}^t_{\text{цт}}, \mathrm{y}^t_{\text{цт}})$ нужно учесть путь тела $\mathrm{S}_{\text{цт}}$. Расчет координат центра тяжести производим по формулам

$\mathrm{x}^t_{\text{цт}} = x^0_{\text{цт}} - \mathrm{S}_{\text{цт}} * \cos(\propto)$,

$\mathrm{y}^t_{\text{цт}} = y^0_{\text{цт}} - \mathrm{S}_{\text{цт}} * \sin(\propto)$.



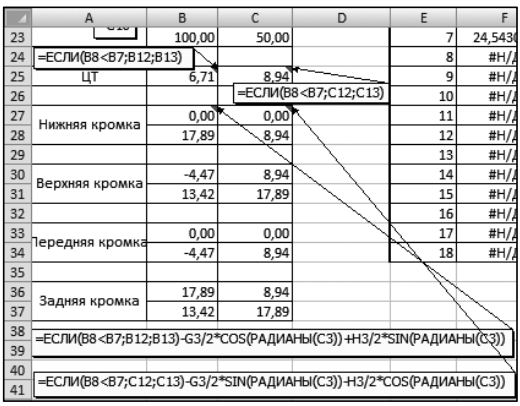
В примечаниях ячеек B12 и C12 можно увидеть реализации представленных формул вычисления текущих координат центра тяжести тела.

Для получения изображения тела в конце спуска определим положение центра тяжести $(\mathrm{x}^{\text{к}}_{\text{цт}}, \mathrm{y}^{\text{к}}_{\text{цт}})$ с учетом того, что оставшийся путь его до основания горки равен половине длины тела. Расчет произведем (B13 и C13) по следующим формулам

$$x^{\text{к}}_{\text{цт}} = l/2 * \cos(\propto) - h/2 * \sin(\propto),\ y^{\text{к}}_{\text{цт}} = l/2 * \sin(\propto) + h/2 * \cos(\propto),$$

где l - длина тела.

Следует заметить, что в начале движения на поверхности горки находится только половина тела. Это должно влиять на величину коэффициента скольжения, которая используется в модели. Однако этот факт опустим.

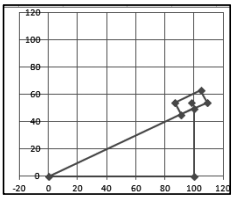


Теперь можно продолжить формирование таблицы координат элементов контура тела. Для изображения точки центра тяже-

сти формируем ячейки текущих координат (B25 и C25) с учетом проверки окончания спуска путем сравнения времени спуска (B7) с текущим временем моделирования (B8). При окончании спуска изображение прямоугольника тела привязываем к конечным координатам центра тяжести (B13 и C13).

Далее по известной методике формируем координаты линий контура прямоугольника тела (см. примечания). Остается путем изменения счетчика текущего времени (B8) в диапазоне от нуля до времени спуска (B7) осуществлять моделирование спуска предварительно сформировав точечную диаграмму по таблице координат элементов (B16:C37).

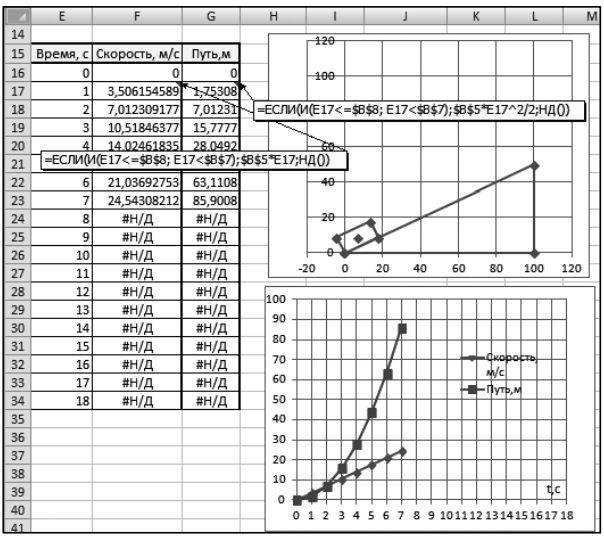
Для исключения искажения при моделировании нужно зафиксировать минимальные и максимальные значения осей. На рисунки показано начальное положение тела. По оси X целесообразно зафиксировать минимальное отрицательное значения для получения изображения тела в конце спуска.



Кроме изображения спуска тела представим графики изменения скорости и пути от времени. Путь вычисляем по уже представленной формуле, а скорость по формуле равноускоренного движения при нулевой начальной скорости

$$V_{\text{цт}} = a * t.$$

Но сначала формируем столбец времени. Для простоты время записываем от нуля до максимально возможного времени спуска. Далее вычисляем скорость и путь тела при условии не превышения записанных значений времени значению счетчика текущего времени. В случае превышения в строке формируется отсутствие данных по скорости и пройденному пути.



Время, с	Скорость, м/с	Путь,м
0	0	0
1	3,506154589	1,75308
2	7,012309177	7,01231
3	10,51846377	15,7777
4	14,02461835	28,0492
5	17,53077294	43,8269
6	21,03692753	63,1108
7	24,54308212	85,9008
8	#Н/Д	#Н/Д
9	#Н/Д	#Н/Д
10	#Н/Д	#Н/Д
11	#Н/Д	#Н/Д
12	#Н/Д	#Н/Д
13	#Н/Д	#Н/Д
14	#Н/Д	#Н/Д
15	#Н/Д	#Н/Д
16	#Н/Д	#Н/Д
17	#Н/Д	#Н/Д
18	#Н/Д	#Н/Д

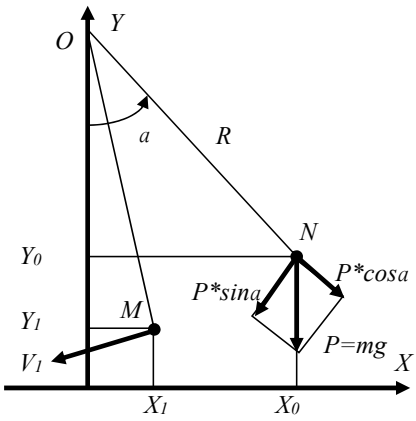
На диаграмме представлены графики скорости и пути, а также положение тела в конце спуска. Однако следует отметить следующие. Время спуска при данном варианте моделирования равно 7,62 с. На графиках последние значения соответствуют 7с. Для получения всех данных нужно уменьшить интервал для формирования значений времени, но

при этом увеличивается число строк. Можно время спуска поделить на конкретное число (10 или 20) и сформировать интервал для записи времени. Значения времени будут при этом не целыми.

2.2.5. Маятник

Маленький шарик весит на нити. Шарик от линии отвеса подняли на определенную высоту и отпустили. Шарик будет совершать колебательные движения.

Создадим модель колебательного движения шарика на нити. Будем считать, что нить нерастяжима, сопротивление воздуха учитывать не будем. Масса шарика равна m, а длина нити R.



Движение маятника описывается уравнением

$$m * \frac{d^2x}{d^2t} = -P * \sin(\propto),$$

где m - масса маятника,

x - отклонение маятника от вертикали,

$\propto$ - угол отклонения нити от вертикали.

Начало движения начинается от вертикальной линии в правую сторону. Поэтому сила взята со знаком минус. Будем рассматривать небольшие угловые отклонения маятника от вертикали. Не более 5 градусов. В этом случае можно записать $\sin \propto = x/R$. После подстановки в первое уравнение получим

$$\frac{d^2x}{d^2t} = -\frac{g}{R} * x.$$

Если ввести обозначение $\omega^2 = g/R$, то уравнение имеет решение

$$x = x_0 * \sin(\omega * t + \varphi),$$

где $\omega = \sqrt{g/R}$ - угловая скорость маятника относительно точки О (точки подвеса),

$x_0 = R * \sin(\alpha_0)$ - амплитуда отклонения маятника от вертикали,

R - длина нити,

α_0 - максимальный угол отклонения маятника от вертикали (имеет положительное и отрицательное значения),

φ - фаза, которая зависит от начального положения маятника (при нахождении маятника справа от вертикали на угле α_0 фаза равна 90^0).

Отклонения маятника от вертикали являются гармоническими колебаниями с периодом $T = 2 * \pi/\omega = 2 * \pi * \sqrt{R/g}$. Эта модель называется математическим маятником.

Теперь можно подготовить выражения для расчета положения маятника от времени по заданным исходным данным.

Координата маятника по оси x вычисляется по формуле

$x_i = x_0 * \sin(\omega * t_i + \varphi)$,

где $t_i = t_0 + i * \Delta t$ – текущий такт времени,

$\Delta t = T/n$ – временной шаг счетчика времени (n=20).

Текущее угловое отклонение вычисляем как $\propto_i = \sin^{-1}(x_i/R)$, а координату по оси y -

$y_i = h_{\text{под}} - R * \sin(\alpha_i)$, где $h_{\text{под}}$ - высота подвески маятника.

Координата точки подвески маятника по оси x равна нулю.

Начнем с задания начальных условий в ячейках A2:C2. Определим амплитуду отклонения маятника по оси x (D2) и зададим начальную фазу положения маятника (E2). В ячейке G3 задаем константу ускорения свободного падения и вычисляем угловую скорость (H2) и период колебания (I3).

	A	B	C	D	E	F	G	H	I
1	Длина нити, м	Начальный угол, град	Высота подвеса, м	Амплитуда маятника, м	Начальная фаза, рад		Ускорение свободного падения, м/с²	Угловая скорость маятника, рад/с	Период, с
2	3	10	3,5	0,520945	1,5707963		9,8	1,80739223	3,47638172
3		=A2*SIN(РАДИАНЫ(B2))			=ПИ()/2		=(G2/A2)^0,5	=2*ПИ()/H2	

Создадим массив для расчета координат траектории маятника (A6:F75). Массив достаточно большой. Это необходимо для моделирования нескольких периодов колебаний маятника. Формируем значения номеров по порядку и вычисляем значения текущих тактов времени (B6:B75). Далее вычисляем координату маятника по оси x, угол отклонения и затем координату по оси y (см. примечания). Количество строк в массиве выбираем с учетом периода колебания маятника.

	A	B	C	D	E	F
5	№ п/п	t_i, с	α_i,рад	x_i,м	y_i,м	v_i,м
6	0	0	0,174532925	0,520945	0,5455767	0,000
7	1	0,173819086	0,165909318	0,495448	0,5411942	0,293
8	2	0,34	0576	0,4		0,557
9	3	0,521457258	0,102245895	0,306204	0,5156677	0,766
10	4	0,695376344	0,03368			0,899
11	5	0,869095451	2,12745			0,945
12	6	1,042914517	-0,05368			0,899
13	7	1,216733603	-0,1022459	-0,3062	0,5156677	0,766
14	8	1,390552689	-0,14095058	-0,42145	0,5297513	0,557
15	9	1,564371775	-0,16590932	-0,49545	0,5411942	0,293
16	10	1,738190861	-0,17453293	-0,52094	0,5455767	0,000

=A6*$L$2 =ASIN(D6/$A$2) =$C$2-$A$2*COS(C6) =(2*$G$2*($O$2-E6))^0,5 =$D$2*SIN($H$2*B6+$E$2) =(2*$G$2*(E6-E7)+F6^2)^0,5

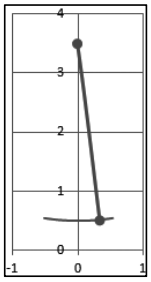
Чтобы наблюдать не только траекторию маятника, но и движение нити с грузом, внизу таблицы сформируем ячейки с координатами концов нити маятника (D77:E78). Но прежде нужно сформировать ячейки счетчика времени (K2) и шага счетчика (L2). Шаг можно увеличить. То есть поделить значение периода на 10. Этого тоже будет достаточно для отображения траектории маятника.

	K	L
1	Счетчик времени, с	Шаг счетчика, с
2	6,43131	0,17382

=I2/20

Координата точки подвеса нити не изменяется при моделировании (D77 и E77). Координаты, где прикреплен груз, зависят от времени. Поэтому для отбора координат груза из массива B6:E75 используем функцию ВПР(), где первым параметром является значение счетчика времени (К2). По этому параметру t_i определяется нужная строка, из которой выбирается координата x_i при значении третьего параметра равного 3 и координата y_i при значении третьего параметра равного 4. Значения третьего параметра в функции ВПР() определяют номера столбцов, из которых выбираются значения координат.

Для отображения траектории движения маятника выделяем столбы D,E таблицы и обращаемся к диаграмме *Точечная с гладкими кривыми и маркерами*. Через контекстное меню ряда убираем маркеры, а через контекстное меню точек линии нити устанавливаем маркеры для имитации точки подвеса и груза. При изменении значения счетчика наблюдаем качание груза и перемещение нити подвеса.



Для автоматизации изменения значения счетчика времени создаем кнопки *Счетчик времени* и *Обнуление счетчика* с программами

```
Private Sub CommandButton1_Click()
Cells(2, 11) = Cells(2, 11) + Cells(2, 12) 'Начисление счетчика
End Sub
Private Sub CommandButton2_Click()
Cells(2, 11) = 0 'Обнуление счетчика
End Sub.
```

Остается для проведения исследований колебаний маятника построить графики отклонения от вертикали и линейной скорости груза маятника в зависимости от времени.

Вычисление линейной скорости груза маятника будем производить с учетом закона сохранения энергии

$$m * V_2^2/2 - m * V_1^2/2 = m * g * (h_2 - h_1),$$

где m – масса тела,

V_1, V_2 – скорости тела в 1-м и 2-м пространственном положении,

h_1, h_2 – высота тела в 1-м и 2-м пространственном положении.

В нашем случае высоты вычисляются по следующим формулам

$$h_{1i} = y_i, h_{2i} = y_{i+1}.$$

Следовательно, скорость определяется по следующему выражению

$$V_{i+1} = (2 * g * (y_{i+1} - y_i) + V_i^2)^{0,5}.$$

В начальной (самой верхней) точке нахождения груза маятника скорость его равна нулю. Однако маятник может находиться в более низкой точке. Поэтому расчет скорости в первой точке будем делать по формуле

$$V_1 = (2 * g * (h_0 - y_1))^{0,5},$$

где h_0 - максимальная высота подъема груза маятника.

Данные по скорости рассчитаны в столбце F таблицы. Построим графики текущего изменения отклонений от вертикали и скорости в зависимости от времени. Копируем из массива A6:F75 столбцы B,D,F и создаем массив O6:Q75 для графиков. При этом проверяем условие: значение времени в столбце O меньше или равно текущему времени. При выполнении условия копируем значения x_i и V_i, а иначе формируем НД(). Для построения графиков обратимся к диаграмме *Точечная с гладкими кривыми и маркерами*. Затем для согласования с моделью маятника через контекстное меню и пункт *Выбрать данные* в окне *Выбор источника данных* активизируем кнопку *Изменить*. Поменяем диапазоны данных для X и Y местами. Далее вызываем контекстное меню оси времени и через пункт *Формат оси* ставим флажок *обратный порядок отсчета.* Также в *Типе линии* увеличиваем ширину и устанавливаем стрелку. Остается для наглядности согласовать размеры графиков.

Также изменяем направление оси времени для графика скорости, форматируем и согласовываем размеры графиков для удобства анализа. Из графиков видно, что скорость максимальна в нижней точке траектории маятника и равна нулю в верхних крайних точках, когда маятник начинает двигаться в противоположную сторону.

С помощью этой модели математического маятника можно наблюдать ряд его замечательных свойств.

2.2.6. Движение бруска при попадании пули

В неподвижный брусок попадает пуля, которая летит горизонтально. После этого брусок начинает двигаться и затем останавливается. Требуется создать модель движения бруска по плоскости.

Известны скорость и масса пули, которая застревает в бруске. Знаем

	A	B	C	D	I	J	K
1	Масса пули, г	Скорость пули, м/с		Масса бруска, г		Коэффициент трения скольжения	Ускорение свободного падения,м/с^2
2	10	120		160		0,5	9,8

массу бруска и коэффициент трения скольжения.

В ячейки занесем значения массы и скорости пули (A2,B2). Зададим массу бруска (D2) и значение коэффициента трения скольжения (J2). В модели понадобиться ускорение свободного падения (K2).

При создании модели в первую очередь нужно определить начальную скорость движения бруска, длину его пути и ускорение.

Из закона сохранения импульса можно записать

$$m_{\text{п}} * V_{\text{п}} = (m_{\text{п}} + m_{\text{б}}) * V_{\text{б}},$$

где $m_{\text{п}}$ - масса пули,

$V_{\text{п}}$ - скорость пули,

$m_{\text{б}}$ - масса бруска,

$V_{\text{б}}$ – начальная скорость бруска с пулей.

Таким образом, начальная скорость бруска вычисляется по формуле

$$V_{\text{б}} = m_{\text{п}} * V_{\text{п}}/(m_{\text{п}} + m_{\text{б}}).$$

Движение бруска происходит под действием сил: тяжести, реакции опоры и трения. Исходя из теоремы о кинетической энергии $\Delta E = -A$.

Кинетическая энергия бруска $\Delta E = (m_{\text{п}} + m_{\text{б}}) * V_{\text{б}}^2/2$.

Работа $A = F_{\text{тр}} * S = k_c * (m_{\text{п}} + m_{\text{б}}) * g * S,$

где $F_{\text{тр}}$- сила трения,

S - путь бруска,

k_c - коэффициент трения скольжения,

$g = 9{,}8$ м/с^2 - ускорение свободного падения.

Следовательно, путь $S = V_{\text{б}}^2/(2 * k_c * g)$.

Так как конечная скорость бруска равна нулю, то время движения бруска вычисляем по выражению $t = 2 * S/V_{\text{б}}$, а ускорение – $a = V_{\text{б}}^2/(2 * S)$.

В ячейках D2:H2 производим все необходимые вычисления по заданным начальным параметрам. Скорость бруска во время движения уменьшается. Поэтому ускорение имеет отрицательный знак (H2).

	C	D	E	F	G	H
1		Масса бруска, г	Начальная скорость бруска, м/с	Длина пути бруска, м	Время движения бруска, с	Ускорение бруска, м/с^2
2		160	7,058823529	5,084386696	1,44057623	-4,9000
3		=A2*B2/(A2+D2)	=E2^2/(2*J2*K2)	=2*F2/E2		=-(E2^2/(2*F2))

Все есть для создания модели движения бруска после попадания в него пули.

	M
1	Временной шаг моделирорвания, с
2	0,072028812
3	
4	=G2/20

Определенный интерес представляют графики пути бруска и изменения скорости от времени движения. Возьмем 20 точек для построения графиков и определим временной шаг.

Скорость будем вычислять по формуле

$$V_i = V_{\text{б}} + a * t_i,$$

где V_i - скорость бруска на момент t_i,

$t_i = (i-1) * \Delta t$ - i-й момент времени i=1,…, 21.

Пройденный путь рассчитываем по формуле $S_i = V_6 * t_i + a * t_i^2/2$.

Все данные для построения графиков находятся в массиве A6:D26. Пройденный путь обозначили через X.

	A	B	C	D
5	№ п/п	Время, с	X,м	V_6, м/с
6	1	0	0	7,058824
7	2	0,07202881	0,495728	6,705882
8	3	0,14405762	0,966033	6,352941
9	4	0,21608643	1,410917	6
10	5	0,28811525	1,830379	5,647059
11	6	0,36014406	2,224419	5,294118
12	7	0,43217287	2,593037	4,941176
13	8	0,50420168	2,936233	4,588235
14	9	0,57623049	3,254007	4,235294
15	10	0,6482593	3,54636	3,882353
16	11	0,72028812	3,81329	3,529412
17	12	0,79231693	4,054798	3,176471
18	13	0,86434574	4,270885	2,823529
19	14	0,93637455	4,461549	2,470588
20	15	1,00840336	4,626792	2,117647
21	16	1,08043217	4,766613	1,764706
22	17	1,15246098	4,881011	1,411765
23	18	1,2244898	4,969988	1,058824
24	19	1,29651861	5,033543	0,705882
25	20	1,36854742	5,071676	0,352941
26	21	1,44057623	5,084387	0

=(A6-1)*$M$2

=$E$2*B6+$H$2*B6^2/2

=$E$2*B6+$H$2*B6^2/2

Зададим параметры бруска (B32,B33) и сформируем его контур в виде прямоугольника (B35:C45). Координаты опорных точек контура связываем с параметрами. Это обеспечит изменение контура при изменении высоты или длины бруска.

Считаем, что пуля застревает в центре бруска (B47,C47). Формируем ячейки текущей скорости (C29) и текущего положения центра тяжести бруска (C30). При этом расчет начинается после момента попадания пули в брусок. Этот момент приравниваем нулю (см. примечания).

Для полноты динамической модели целесообразно отобразить полет пули до ее попадания в брусок. С этой целью задаем количество упрежденных тактов времени до попадания пули (H6).

	H
5	Упреждение
6	3

Для изображения динамики полета пули и перемещения бруска создадим массив F33:G49. К координатам опорных то-

	A	B	C
27	=ЕСЛИ(F6>=0;E2+H2*F6;)		
28			
29	Текущая скорость, м/с		0,000
30	Текущее значение ЦТ, м		5,084
31	=ЕСЛИ(F6>=0;$E$2*F6+$H$2*F6^2/2; 0)		
32	Высота бруска, м	0,1	
33	Длина бруска, м	0,4	=-$B$33/2
34			
35	=-$B$33/2	-0,2	0,1
36	=-$B$35	0,2	0,1
37		=-$B$35	
38	Начальное положение контура бруска	0,2	0,1
39		0,2	0
40			
41		0,2	0
42		-0,2	0
43			
44		-0,2	0
45		-0,2	0,1
46			=$B$32/2
47	Положение пули в бруске	0	0,05

	D	E	F	G
32			X, м	Y, м
33	Минимальное значение		-25,9303721	
34				
35	=-B2*3*M2		4,884386696	0,1
36	=-$B$33/2+C30		5,284	0,1
37				
38	=-$B$35+C30	=$B$32	5,284386696	0,1
39		Брусок	5,284386696	0
40				
41			5,284386696	0
42			4,884386696	0
43				
44			4,884386696	0
45			4,884386696	0,1
46	=ЕСЛИ(F6>=0;E2*F6+H2*F6^2/2;B2*F6)			
47		Пуля	5,084386696	0,05
48				
49	Максимальное значение		6,35548337	
50				
51	=F2+F2/4			

чек контура бруска добавляем текущее значение центра тяжести по оси X. Координату по X полета пули до нулевого момента вычисляем по ее скорости полета, а после попадания – с учетом скорости движения бруска (см. примечание).

Максимальные и минимальные координаты в модели пули и бруска очень сильно меняются при изменении начальных условий. Для автоматизации нбачальной установки в диаграмме максимальных и минимальных значений по оси X сформировали ячейки, где вычисляем минимальное значение координаты пули с учетом упреждения (F33) и максимальное значение положение бруска с запасом (F49).

Теперь для автоматизации формирования значений счетчика времени (F6) создадим кнопки *Обнуление счетчика* и *Начисления счетчика* со следующими программами:

```
Private Sub CommandButton1_Click()
Cells(6, 6) = -Cells(6, 8) * Cells(2, 13) 'с учетом упреждения
End Sub
Private Sub CommandButton2_Click()
If Cells(6, 6) < Cells(2, 7) Then 'проверка превышения времени движения
Cells(6, 6) = Cells(6, 6) + Cells(2, 13)
Else
Cells(6, 6) = Cells(2, 7)
End If
End Sub.
```

Окончание движения бруска должно совпадать с наступление момента времени, который вычислен в ячейке G2. Поэтому при наступлении этого момента счетчик перестает начислят время.

Изображение модели осуществляется с использование двух графиков.

На первом графике наблюдаем полет пули и брусок по данным, расположенным в массиве (F33:G49).

Для наблюдения движения бруска с пулей используем данные, расположенные в массиве (F35:G49). Здесь можно рассмотреть сам брусок и пулю в нем.

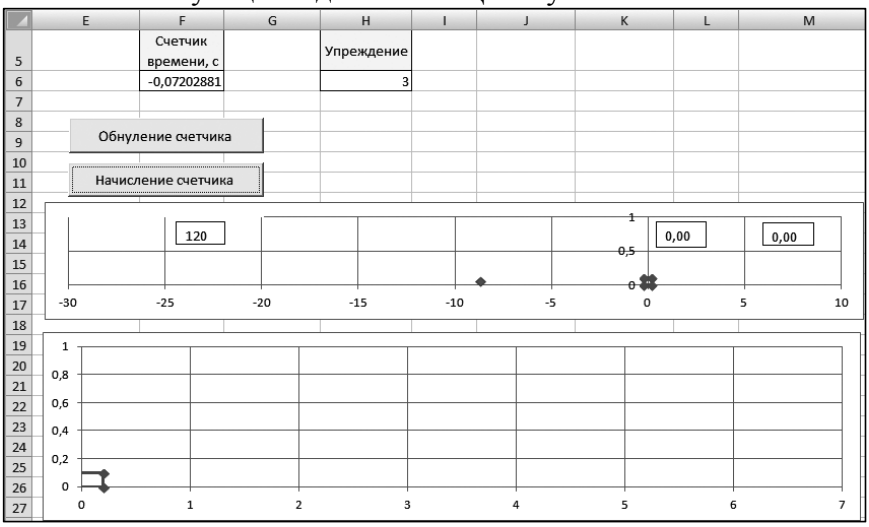
Графики берем точеные с прямыми отрезками и маркерами. С целью наглядности через контекстное меню маркера пули выбираем пункт *Формат точки данных* и устанавливаем красный цвет маркера.

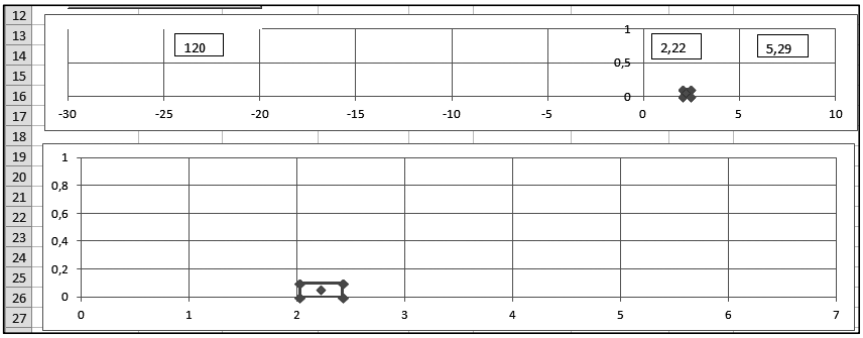
На графиках через контекстное меню оси Y выбираем пункт *Формат оси* и фиксируем максимальные и минимальные значения оси.

Во втором графике нужно зафиксировать минимальное значение на оси X. Заносим значение равное нулю. Поэтому наблюдаем только половину контура бруска до момента попадания в него пули.

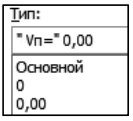
В объектах *Надпись* в первом графике помещены значения скорости пули, текущего положения центра тяжести бруска и его скорости.

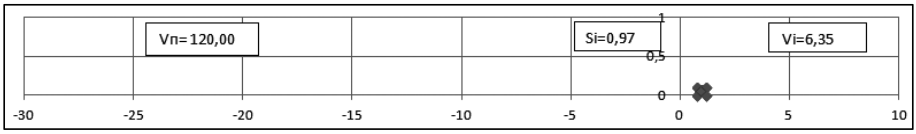
Это позволяет повысить наглядность модели. В этом случае нужно активизировать объект *Надпись*, протяжкой установить размер в нужном месте. Затем в *Строке формул* поставить знак равенства и щелкнуть по ячейке с соответствующими данными. Щелкнуть по *Enter*.



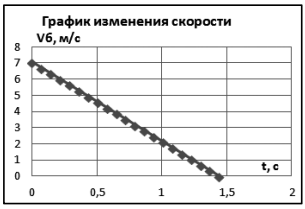


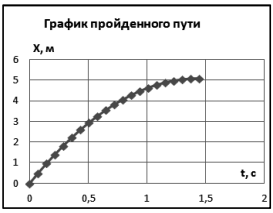
Наблюдать в объектах только числа не очень удобно. Создадим пользовательский формат с включение соответствующих обозначений величин. Чтобы не влиять на вычисления в модели скопируем нужные величины в отдельные ячейки. К этим ячейкам применим соответствующие пользовательские форматы и используем их в объектах *Надпись* для индикации. Выходим на *Формат ячеек*, на закладке *Число* щелкаем на *Все форматы* и выбираем *Тип* 0,00. Далее устанавливаем кавычки и внутри их вносим соответствующие обозначения для величин. Затем форматируем ячейки и вставляем их в объекты *Надпись*.





Построим графики изменения скорости и пути бруска после попадания пули. Видим, что скорость падает до нуля, а величина прохождения пути за один такт времени уменьшается.





2.2.7. Сообщающиеся сосуды

На поверхности стоят две емкости, которые между собой на уровне поверхности соединены каналом с малым сечением. Размеры емкостей одинаковы. Одна емкость полностью заполнена водой. Мгновенно открывается кран, вода начинает заполнять пустую емкость.

Определить время заполнения емкости и уровень воды в емкостях.

При разработке модели будем считать, что объем канала мал, вода не сжимается и процесс перетекания воды стационарный.

При стационарном течении скорость воды в различных сечениях трубы обратно пропорциональна площади поперечного сечения

$$V_{\text{п}}/V_1 = S_{\text{к}}/S_{\text{п}},$$

где $V_{\text{п}}$ - скорость поверхности воды в первой емкости,

V_1 - скорость воды в канале между емкостями,

$S_{\text{к}}$ - площадь сечения канала,

$S_{\text{п}}$ - площадь основания емкости.

Составим уравнение Бернулли для нашего случая

$$\rho * {V_{\text{п}}}^2/2 + P_{\text{атм}} + \rho * g * h_1 = \rho * {V_1}^2/2 + P_{\text{атм}} + \rho * g * h_2,$$

где ρ=1000 кг/м^3 – плотность воды,

g=9,8066 м/с^2 – ускорение свободного падения,

$P_{\text{атм}}$ - атмосферное давление,

h_1 и h_2 - уровни воды в емкостях.

Так как $S_{\text{п}} \gg S_{\text{к}}$, то $V_{\text{п}}$ скорость воды вблизи поверхности в широком сосуде пренебрежимо мала.

Уравнение Бернулли принимает вид

$$\rho * g * h_1 = \rho * {V_1}^2/2 + \rho * g * h_2.$$

Из уравнения получаем формулу Торричелли, которая определяет скорость истечения воды из первой емкости

$$V_1 = (2 * g * (h_1 - h_2))^{0,5}.$$

Зная скорость, можно определить объем воды $\Delta V_в$ вытекаемый из канала за промежуток времени Δt

$$\Delta V_в = S_к * V_1 * \Delta t.$$

Теперь определяем толщину этого объема в емкости $\Delta h = \Delta V_в / S_п$, а затем уменьшение и увеличения высоты воды в соответствующих емкостях на (i+1) такте моделирования по формулам

$$h_{1i+1} = h_{1i} - \Delta h,\ h_{2i+1} = h_{2i} - \Delta h.$$

Все готово для моделирования перетекания воды из одной емкости в другую. Создадим ячейки с параметрами емкостей (A3:C3) и вычислим объем налитой воды. При этом нужно помнить, что 1м^3 равен 1000 литрам. Далее занесем значение плотности воды (F3), ускорения свободного падения (G3) и определим ячейки счетчика времени (I3), шаг изменения счетчика (J3) и точность вычисления скорости истечения воды (K3). Шаг начисления счетчика можно изменять при моделировании. Счетчик изменяется от нуля до момента, когда уровни воды станут равны в сообщающихся сосудах. Так как скорость истечения воды по ходу заполнения второй емкости будет уменьшаться до нуля, то вводим ограничения по точности расчета скорости (K3). Это сэкономит нам количество ячеек.

	A	B	C	D
1	Параметры емкостей			Площадь
2	Высота,м	Площадь дна, м^2	Объем , м^3	сечения канала, м^2
3	3	1	3	0,002
4		=A3*B3		

	F	G	H	I	J	K
1	Плотность воды, кг/м^3	Ускорение свободного падения, м/с^2		Счетчик, с	Шаг счетчика, с	Точность вычисления скорости, м/с
2						
3	1000	9,80665		68	2	0,0005

Создадим массив F7:K107, где в зависимости от времени будем вычислять скорость потока воды из первой емкости во вторую, уровни воды и перепад уровней за заданный промежуток времени. Так как разность уровней может быть отрицательной, то для определения точности вычислений уровней используем функцию ABS(). Если уровни мало отличаются, то скорость потока приравниваем к нулю.

	F	G	H	I	J	K
5	=ЕСЛИ(ABS(I7-J7)<$K$3;0;(2*$G$3*(I7-J7))^0,5)					
6	№ п/п	Время,с	Скорость потока, м/с	h_1,м	h_2,м	Δh,м
7	0		0	3	0	0
8	1	2	7,591858	2,969317	0,030683	0,03068287
9		4	7,[illegible]996	2,93895[illegible]	[illegible]05	0,03036743
10	3	6	7,[illegible]	2,908898	0,091102	0,03005198
11	4	8	7,[illegible]	2,879161	0,120839	0,02973651
12	5	10	7,276381	2,84[illegible]	[illegible]	[illegible]103
13	6	12	7,1975	2,820635	0,179365	0,02910552

=F7*$J$3 =A3 =I7-K8 =J7+K8 =$D$3*H7*$J$3/$B$3

	F	G	H	I	J	K
101	94	188	0,13969	1,500497	1,499503	0,00095466
102	95	190	0	1,499939	1,500061	0,00055876
103	96	192	0	1,499939	1,500061	0
104	97	194	0	1,499939	1,500061	0
105	98	196	0	1,499939	1,500061	0
106	99	198	0	1,499939	1,500061	0
107	100	200	0	1,499939	1,500061	0

Следует отметить, что начальное значение уровня в первой области задается, начальное значение второго уровня и перепад равны нулю. Далее производятся вычисления параметров процесса переливания воды с учетом предыдущих вычислений. В таблице массива по нулевым значениям скорости потока фиксируем время заполнения сообщающихся сосудов. Видим, что на 95 шаге моделирования или на 190 секунде уровни воды в емкостях уравниваются (с учетом заданной точности), а скорость потока стала равна нулю.

	B	C	D
6	Элементы диаграммы	X	Y
7		0	0
8		0	3
9			
10	Первая емкость	0	0
11		1	0
12			
13		1	0,04472136
14		1	3
15			
16		1	0
17		1,2	0
18			
19		1	0,04472136
20	Канал	1,2	0,04472136
21			
22		1,2	0,04472136
23		1,2	3
24			
25	Вторая емкость	1,2	0
26		2,2	0
27			
28		2,2	0
29		2,2	3
30			
31	Первый	0	2,13386384
32	уровень	1	2,13386384
33			
34	Второй	1,2	0,86613616
35	уровень	2,2	0,86613616

=C10+B3^0,5

=D3^0,5

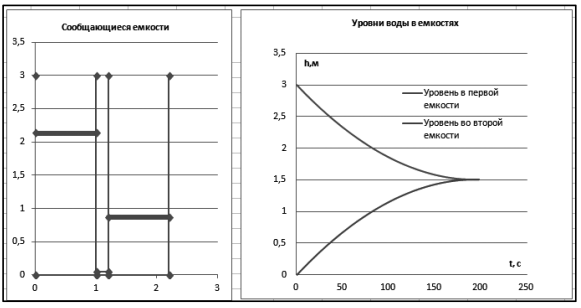
=C16+0,2

=ВПР($I$3;$G$7:$K$107;3)

=ВПР($I$3;$G$7:$K$107;4)

Теперь наглядно в динамике изобразим процесс перетекания воды в сообщающихся сосудах. В массиве B7:D35 определим координаты для изображения контуров емкостей, канала между ними и уровней воды. Для простоты будем считать, что емкости и канал имеют сечения квадратной формы (см. примечания в ячейках C11 и D13). Следовательно, размер стороны дна емкости равен 1м, а сторона канала равна 4,5 см. Для задания высоты уровней воды (C31:B35) воспользуемся функциями ВПР(), которые по значению счетчика определяют строку в массиве параметров и выбирают значение соответствующей высоты уровня. Выделив массив, обращаемся к диаграмме *Точечная с прямыми отрезками и маркерами*. Производим форматирование линий уровня воды через контекстное меню точки координаты уровня и пункт *Формат точки данных.*

Для автоматизации моделирования создаем управляющие кнопки *Обнуления счетчика* и *Начисления счетчика* с соответствующими программами. В начале моделирования щелкаем на кнопке *Обнуления счетчика*, а затем на кнопке *Начисления счетчика* до момента появления нулевой скорости потока или выхода за пределы массива параметров. Если при моделировании не наступает момент равенства уровней воды в емкостях, то нужно увеличить шаг начисления счетчика или количество ячеек в массиве параметров.



На рисунке представлен момент перетекания воды при значении счетчика времени

равного 68 секундам, а рядом график изменения уровней воды в сообщающихся сосудах. На графике видно, что в районе 170-180 секунд процесс перетекания прекращается и уровни воды равны 1,5 метрам.

2.2.8. Подъем аэростата

Аэростат имеет массу 500кг. Форма аэростата шар с радиусом равным 5м.

Необходимо определить максимальную высоту подъема и создать динамическую модель подъема аэростата.

Начальная высота и скорость подъема равна нулю. При моделировании сопротивление воздуха учитывать не будем.

Максимальную высоту подъема определим из равенства силы тяжести аэростата $F_т$ и подъемной силы Архимеда $F_{арх}$

$$F_т = F_{арх}.$$

Плотность воздуха на максимальной высоте подъема должна равна

$$\rho_{макс} = m_а / V_а,$$

где $m_а$ - масса аэростата,

$V_а$ - объем аэростата.

Плотность воздуха вычисляется по уравнению Менделеева-Клапейрона

$$\rho = Р * M / R * T$$

где ρ - плотность воздуха,

Р - абсолютное давление,

$M = 0{,}0289644$ кг/Мол - молярная масса,

R= 8,31447 Дж/Мол*К - универсальная газовая постоянная,

T - абсолютная температура в Кельвинах.

Давление и температура изменяются с высотой. Эти изменения определяются формулами

$$P = P_0 * (1 - L * h / T_0)^{g*M/R*L},$$
$$T = T_0 - L * h,$$

где P_0=101325 Па - стандартное атмосферное давление на уровне моря,

L= 0,0065 К/м - скорость падения температуры с высотой,

h - высота в метрах,

g=9,80665 м/ сек^2 – ускорение свободного падения.

При подстановке в уравнение Менделеева-Клапейрона температуры и давления в зависимости от высоты после преобразований получаем формулу расчета высоты от заданной плотности воздуха

$$h = L^{-1} * (T_0 - (\rho * {T_0}^b / a * P_0)^{1/(b-1)}),$$

где $a = M / R$,

$b = g * M / R * L$.

Если подставить в формулу значение $\rho_{макс}$, то можно вычислить значение максимальной высоты подъема аэростата.

	A	B	C	D
1	Параметры аэростата			
			=4/3*ПИ()*B3^3	
2	Масса, кг	Радиус шара, м	Объем шара, м³	
3	500	5	523,599	

Занесем исходные параметры аэростата (A3:B3) и вычислим его объем (C3).

Далее сформируем ячейки с параметрами стандартной атмосферы

	E	F	G	H	I	J	K
1			Параметры стандартной атмосферы				
2	Ускорение свободного падения, м/с²	Плотность воздуха на уровне моря при 0° С, кг/м³	Давление на уровне моря, Па	Температура на уровне моря, К	Скорость падения температуры с высотой, К/м	Универсальная газовая постоянная, Дж/(Мол*К)	Молярная масса сухого воздух, кг/Мол
3	9,80665	1,292	101325	288,15	0,0065	8,31447	0,0289644

для вычисления плотности воздуха на нужной высоте (E3:K3). Внесем значение ускорение свободного падения и плотность воздуха на уровне моря при 0^0 С, которая может пригодится для анализа правильности расчетов.

Заранее вычислим ряд констант, которые помогут упростить выражения вычислений плотности воздуха (M3:Q3).

	M	N	O	P	Q
1	Вспомогательные параметры				
	=K3/J3	=E3*K3/(J3*I3)	=A3/C3	=1/O3	=M3*G3/H3^N3
2	a=M/R	b=g*M/(R*L)	m/V	V/m	a*P_0/T_0^b
3	0,0034836	5,255781293	0,954929659	1,047197551	4,17377E-11

Определим максимальную высоту подъема аэростата по заданным исходным данным (A6). Высота чуть больше 2,5 км. Рядом выделим ячейки для счетчика времени и шага его изменения (C6, D6). Результаты определения максимальной высоты подъема аэростата можно проверить по таблице изменения плотности воздуха от высоты. Создадим таблицу (N6:S46) изменения плотности воздуха от 0 до 20000 м с шагом 500 м. При этом пересчитаем температуру в градусы по Цельсию и давление в мм ртутного столба по формулам

	A	B	C	D
4	=(H3-(O3*H3^N3/(M3*G3))^(1/(N3-1)))/I3			
5	Максимальная высота подъема, м		Счетчик, сек	Шаг счетчика, сек
6	2519,703949		80	20

$$T^0C = T^0K - 273{,}15,$$

$$P_0\,\text{Па} = 0{,}007506 * P_0\,\text{мм рт. ст.}$$

Плотность аэростата равна 0,9549 (O3). В таблице в ячейке S11 плотность воздуха равна 0,95685. Это соответствует высоте 2,5 км. Следовательно, максимальная высота подъема аэростата определена правильно. По данным таблицы можно построить необходимые графики по стандартной атмосфере.

	N	O	P	Q	R	S
		=O6-273,15		=Q6*0,007506		
5	h, м	T, K	t^0 c	p, Па	p,мм рт. Ст.	p,кг/м3
6	0	288,15	15	101325	760,54545	1,22498
7	500	284,9	11,75	95460,94201	716,5298308	1,16725
8	=$H$3-$I$3*N6	281,65	8,5	89874,76494	674,5999856	1,11162
9	1500	278,4	5,25	84556,28083	634,6794439	1,05805
10	2000	275,15	2	79495,56349	596,6936996	1,00648
11	2500	271,9	-1,25	74682,94472	560,570183	0,95685
12			-4,5			0,90911
13			-7,75			0,86322
14	4000	262,15	-11	61640,78801	462,6757548	0,81912

=$G$3*(1-$I$3*N6/$H$3)^$N$3 =Q6*$K$3/($J$3*O6)

Пусть аэростат находится на высоте h_i , тогда можно записать

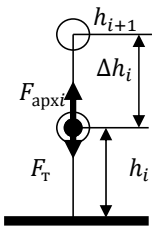
$$F_{\text{арх}i} - F_{\text{т}} = a_i * m_{\text{а}} \text{ и}$$

$$a_i = (F_{\text{арх}i} - F_{\text{т}})/m_{\text{а}} \ = \ (V_{\text{а}} * \rho_i * g - m_{\text{а}} * g)/m_{\text{а}} = g * (V_{\text{а}}/m_{\text{а}} * \rho_i - 1),$$

где a_i - ускорение аэростата за счет действия сил на высоте h_i,

ρ_i - плотность воздуха высоте h_i.

За время Δt аэростат увеличит высоту на $\Delta h_i = a_i * \Delta t^2/2$. Скорость подъема на участке вычисляется как $V_i = a_i * \Delta t$. Следовательно, высота подъема будет равна $h_{i+1} = h_i + \Delta h_i$.



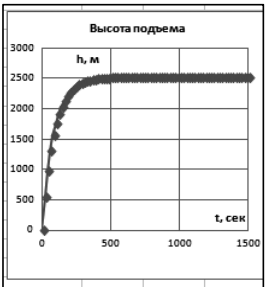
Индекс (номер по порядку в таблице) возрастает от нуля. При нулевом индексе высота равна нулю.

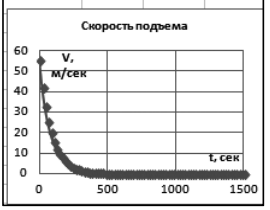
Создадим таблицу расчета высоты подъема в зависимости от временного шага Δt. Возьмем 501 строку в таблице. Количество строк в таблице можно затем откорректировать.

	F	G	H	I	J	K	L
5	№ п/п	Время, сек	Плотность воздуха на высоте подъема	Ускорение, м/сек^2	Скорость подъема, м/сек	Высота участка подъема, м	Высота подъема,м
6	0	0	1,224977056	2,773251712	55,46503421	554,650342	0
7	1	20	1,161067024	2,11692765	42,33855299	423,38853	554,6503421
8	2	40	1,114024172	1,633821089	32,67642179	326,764218	978,035872
9	3	60	1,07872	9444	638889	254,263889	1304,80009
10	4	80	1,051855494	0,995379852	19,90759703	199,0	59,063979
11	5	100	1,031177608	0,783028303	15,66056606	156,605661	1758,139949
12	6	120	1,015130806	0,618235673	12,36471347	123,647135	1914,74561

=K6
=F6*$D$6
=$Q$3*($H$3-$I$3*L6)^($N$3-1)
=$E$3*($P$3*H6-1)
=I6*$D$6
=I6*$D$6^2/2
=L7+K7

По мере подъема ускорение, скорость и высота участков уменьшается. При приближении к максимальной высоте они практически становятся равными нулю. Это можно увидеть на графиках, которые строятся путем выделения нужных столбцов (G,K и J) таблицы с помощью групповой клавиши Ctrl. После 500 сек подъема аэростат достигает максимальной высоты при Δt =20 сек.





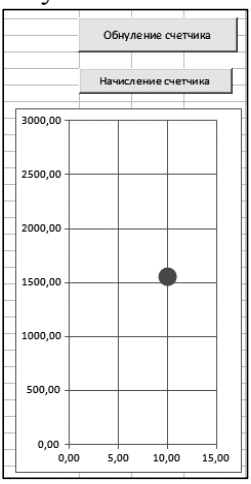
Осталось создать динамическую модель подъема аэростата. Воспользуемся функцией ВПР() для выбора по значению текущего значения счетчика (C6) строки в таблице (G6:L506) и соответственно значения высоты подъема аэростата в столбце L. Для изображения подъема аэростата сформируем ячейку C9 для координаты абсцисс, в которую занесем значение 10. В ячейке D9 формируется значение высоты при проверке условия не превышения текущего значения максимальной высоты подъема. В противном случае записывается значение максимальной высоты (см. примечание).

	A	B	C	D
8	Координаты аэростата		X	H
9		1559,06	10,00	1559,06
10				

=ВПР(C6;G6:L506;6
=ЕСЛИ(B9<A6;B9;A6)

Далее выделив ячейки C9 и D9 обращаемся к диаграмме *Точечная с маркерами*. При этом увидим два маркера. Нужен только один маркер для изображения аэростата. Поэтому через контекстное меню области диаграммы выбираем пункт *Выбрать данные* и через кнопку *Изменить* заносим в окно *Значения X* ячейку C9, а в окно *Значения Y* - D9. Осталось только сделать маркер в виде круга.

Моделирование подъема аэростата осуществляется путем изменения значения счетчика. На рисунке представлен момент подъема на 80 секунде полета.



С целью автоматизации начисления счетчика создадим управляющие кнопки Обнуление и начисление счетчика с соответствующими программами:

```
Private Sub CommandButton1_Click()
Cells(6, 3) = 0 'Обнуление счетчика
End Sub
Private Sub CommandButton2_Click()
Cells(6, 3) = Cells(6, 3) + Cells(6, 4) 'Начисление счетчика
End Sub.
```

Осталось, изменяя исходные параметры и шаг начисления счетчика, проводить исследования по подъему аэростата с различными исходными данными.

2.2.9. Моделирование фигур Лиссажу

Создадим модель фигур Лиссажу.

Пусть заданы два взаимно перпендикулярных колебаний

$$X = A * \sin(\omega_x * t + \varphi_x) \quad \text{и} \quad Y = B * \sin(\omega_y * t + \varphi_y)),$$

где *A, B* - амплитуды колебаний;

ω_x, ω_y - частоты колебаний;

t - время;

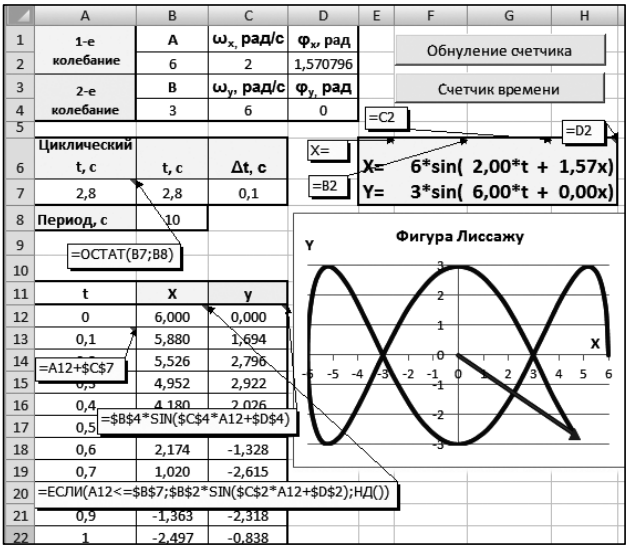
φ_x, φ_y - фазы колебаний.

Наблюдение за фигурами Лиссажу является интересным методом исследования соотношений между частотами (периодами) и фазами гармонических колебаний.

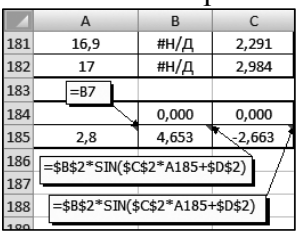
В ходе моделирования фигур Лиссажу нужно обеспечить построение траектории движения точки, получаемой при сложении двух колебаний, и показать движение самой точки в зависимости от времени *t*.

Для построения траектории производим расчеты значений *X* и *Y* (ячейки B12:C182) по заданным начальным условиям (ячейки B2:D2 и B4:D4) в зависимости от *t*. Время (B7) должно изменяться от 0 до такого значения, которое обеспечивало бы полное построение фигуры. Шаг вре-

мени Δt (C7) должен обеспечивать построение неискаженной траектории. При необходимости можно в зависимости от частот колебаний рассчитать время периода и его значение использовать для определения Δt. Но для простоты возьмем $\Delta t=0,1$ с, а максимальное значение времени 17 с. Для построения фигуры Лиссажу выделим диапазоны значений X и Y (B12:C182). Обратимся к диаграмме *Точечная* и зафиксируем максимальные и минимальные значения осей координат через *Формат оси* на закладке *Шкала*. Максимальные и минимальные значения осей координат определяются значениями амплитуд колебаний A (B2) и B (B4). В результате получим изображение фигуру Лиссажу.



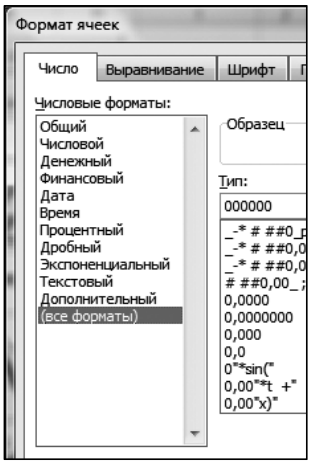
Однако особый интерес вызывает моделирование траектории точки в зависимости от времени, то есть сам процесс формирования фигуры. Для этого предлагается ввести проверку соответствия значения времени в столбце t (ячейки A12:A182) значению счетчика времени (B7). При невыполнении условия в соответствующую ячейку координаты X записывается функция НД(), которая исключает построение графика для соответствующих значений X и Y.



Чтобы обеспечить видимость движения точки отобразим на диаграмме ее радиус-вектор. В ячейках B184 и C184 задаем начало вектора. Координаты конца вектора (B185 и C185) определяем для текущего значения счетчика времени (A185). Ячейки с координатами радиуса-вектора отделяем одной строкой от массива значений координат фигуры для обеспечения отдельного изображения на диаграмме радиуса-вектора. При этом, эти ячейки также должны быть включены в диапазон ячеек для построения фигуры. На конце радиуса-вектора целесообразно сформировать «стрелку». Наводим на конец радиуса-вектора указатель и удерживаем левую клавишу до появления крестообразного указателя. Далее через контекстное меню выходим на *Формат точки данных* и выбираем маркер в виде «стрелки».

Теперь можно приступить к моделированию процесса образования фигуры Лиссажу. Задавая значение счетчика времени (B7) с нулевого значения наблюдаем за формированием фигуры и движением радиуса-вектора точки.

Фигуры Лиссажу используются для сравнения гармонических колебаний. Поэтому важно наблюдать не только за формой фигуры, но и видеть сами уравнения гармонических колебаний с текущими параметрами. Воспользуемся пользовательским форматом, который определяется текстом, заключенным в двойные кавычки. Рассмотрим создание уравнения для одного гармонического колебания. В ячейку E6 заносим «X=». Пусть при моделировании амплитуды будут целочисленными. Активизируем ячейку F6 и обращаемся к *Формат ячейки*. Выбираем закладку *Числовой* и выбираем пункт *Все форматы*. Остановимся на целочисленном формате для значения амплитуды и после *0* в окне *Тип* вставим кавычки и занесем текст «*sin(». Теперь в ячейку F6 заносим ссылку на ячейку, где находиться значение амплитуды гармонического колебания *A* (B2). Переходим к формированию формата для следующей ячейки G6, где будет отображаться гармоническая частота. Аналогично переходим к основному формату, выбираем формат с двумя знаками после запятой. После *0,00* устанавливаем курсор, вставляем кавычку и заносим текст «*t +», закрываем кавычку. *В* ячейке G6 формируем ссылку на ячейку с частотой гармонического колебания (C2). Остается в четвертой ячейке H6 установить формат на основе текста «x)» и занести ссылку на фазу колебания (D2) как и для частоты. Аналогично форматируем ячейки для записи формулы для второго гармонического колебания.



Для автоматизации процесса моделирования создадим кнопки *Обнуление счетчика* и *Счетчик времени* со следующими текстами программ:

```
Private Sub CommandButton1_Click()
Cells(7, 2) = 0
End Sub
Private Sub CommandButton2_Click()
Cells(7, 2) = Cells(7, 2) + Cells(7, 3)
End Sub.
```

После установки начальных параметров гармонических колебаний щелкая на соответствующих кнопках можно начинать моделирование фигур Лиссажу.

Для обеспечения периодичности моделирования фигуры можно задать временной период моделирования и сформировать циклический счетчик времени (A7) при использовании функции ОСТАТ(), где в качестве аргументов используются значения счетчика времени (B7) и период моделирования (B8). Тогда в формулах вместо значений счетчика времени используются значения ячейки A7. В этом случае можно уменьшить массив значений координат фигуры. Если при моделировании формирование фигуры не заканчивается в заданный период (B8), то период нужно увеличить.

2.2.10. Робот футболист с мячом

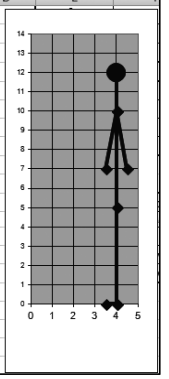
В последнее время большой интерес у всех вызывают роботы. Попробуем смоделировать простейших роботов для приобретения навыков в разработке алгоритмов и основных блоков, обеспечивающих движение роботов. Рассмотрим моделирование робота-футболиста, который жонглирует мячом с помощью ноги. При этом для простоты введем некоторые ограничения. Робот-футболист стоит на месте и при жонглировании ногу не сгибает в коленке. Мяч в момент удара ногой поднимается вверх и под силой тяжести падает вниз до соприкосновения с ногой робота. Скорость движения ноги можно регулировать и задавать скорость мяча в момент удара. Так как нас в первую очередь интересует кинематика, то при моделировании опустим вычисление изменения скорости мяча от масс и скоростей ноги и мяча. Начальное положение мяча определяется координатами ноги робота во время удара.

Моделирование робота будем осуществлять последовательно по следующим этапам:

1. Определение размеров фигуры робота.
2. Разработка модели движения ударной ноги робота.
3. Моделирование вертикального движения мяча.
4. Организация временных циклов моделирования полета мяча, движения ноги и синхронизация их для обеспечения нанесения удара по мячу.

Этап 1. Вначале создадим неподвижную фигуру робота-футболиста из простейших линий. Координаты концов отдельных элементов фигуры задаем в соседних парах ячеек. Чтобы отдельные элементы не соединялись между собой, между парами строк с ячейками, содержащими координаты, пропускаем по одной строке. Голову задаем одной координатой. После задания координат

	A	B	C	D	E
1	Элемент	X,м	Y,м		
2	Нога	4	0		
3		4	5		
4					
5	Ступня	4	0		
6		3,5	0		
7					
8	Туловище	4	5		
9		4	10		
10					
11	Шея	4	10		
12		4	12		
13					
14	Голова	4	12		
15					
16	Рука1	4	10		
17		4,5	7		
18					
19	Рука2	4	10		
20		3,5	7		



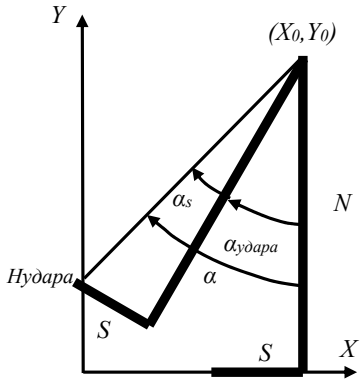
можно обратиться к мастеру построения диаграмм, выбрать *Точечная с прямыми отрезками и маркерами,* а затем, получив изображение откорректировать размеры отдельных элементов фигуры робота.

После исправления размеров элементов фигуры можно через контекстное меню выйти на пункт *Формат рядов данных* и выбрать приемлемую толщину линий и маркер концов элементов фигуры робота.

Для отображения головы робота необходимо навести указатель на маркер головы, щелкнуть левой клавишей мыши для выделения точки (появляется маркер в виде креста со стрелками) и через контекстное меню и пункт *Формат точки данных* задать подобающий размер, цвет и фон точки.

Целесообразно установить максимальные и минимальные значения шкал диаграммы, а также растянуть диаграмму для установки одинаковых размеров цен деления шкал для обеспечения неискаженного изображения робота по горизонтали и вертикали.

Этап 2. Теперь можно перейти к заданию координат ударной ноги робота. Исходное положение ударной ноги совпадает с неподвижной ногой. Пусть нога движется с постоянной заданной угловой скоростью до точки встречи с мячом и обратно в исходное положение (см. рисунок). Ступня всегда перпендикулярна ноге. Нужно вычислить угол, на который поворачивается нога $\alpha_{\text{удара}}$.



$$\alpha_{\text{удара}} = \alpha - \alpha_s,$$

где α - угол, на который поворачивается крайняя точка ступни;

$\alpha_s = arctg(S/N)$ - угловая величина ступни;

S - размер ступни;

N - длина ноги.

Угол α находим из условия равенства координаты X крайней точки ступни нулю в момент удара по мячу

$$0 = X_0 - (N^2 + S^2)^{0,5} * \sin\alpha,$$
$$\alpha = \arcsin(X_0 - (N^2 + S^2)^{0,5})).$$

При заданной угловой скорости ноги ω и $\alpha_{\text{удара}}$ можно найти время движения ноги до удара

$$T_{\text{удара}} = \alpha_{\text{удара}}/\omega.$$

Это время в дальнейшем используем для организации цикла жонглирования мячом.

Для моделирования движения ноги определим выражения для изменения координат отрезков, имитирующих ногу и ступню, в зависимости от угла поворота $\alpha_{\text{тек}}$, который изменяется в диапазоне от нуля до $\alpha_{\text{удара}}$. Одна из координат отрезка ноги *(X₀,Y₀)* остается неизменной, а координаты нижнего конца ноги определяются по формулам

$$X_{N\text{тек}} = X_0 - N * \sin \alpha_{\text{тек}},$$
$$Y_{N\text{тек}} = Y_0 - N * \cos \alpha_{\text{тек}}.$$

Одни координаты пятки соответствуют координатам движущегося конца ноги, а другие координаты ступни, рассчитываются по формулам с учетом перпендикулярности ступни к ноге

$$X_{S\text{тек}} = X_{N\text{тек}} - S * \cos \alpha_{\text{тек}},$$
$$Y_{S\text{тек}} = Y_{N\text{тек}} + S * \cos \alpha_{\text{тек}}.$$

Этап 3. Остается определить выражения для расчета вертикального движения мяча, который в момент удара начинает движение с заданной начальной скоростью V_0. Вектор скорости в момент удара направлен вертикально вверх.

Высота мяча $Y_{\text{мяча}}$ вычисляется по формуле

$$Y_{\text{мяча}} = H_{\text{удара}} + V_0 * T - g * T^2/2,$$

где $H_{\text{удара}}$ - начальная высота полета;

$g = 9{,}8\ \text{м/с}^2$ - ускорение свободного падения;

T - текущее время.

Текущая скорость $V_{\text{тек}}$ определяется выражением

$$V_{\text{тек}} = V_0 - g * T.$$

Эта скорость понадобиться для получения текущих характеристик полета мяча.

Для определения в диаграмме максимального значения шкалы по *Y* целесообразно иметь значения максимальной высоты подъема мяча $Y_{макс}$, которая вычисляется по формуле с учетом высоты удара по мячу $H_{удара}$

$$Y_{\text{макс}} = H_{\text{удара}} + V^2/(2 * g).$$

Этап 4. При моделировании нужно знать полное время полета мяча $T_{\text{пол}}$, которое рассчитывается по формуле

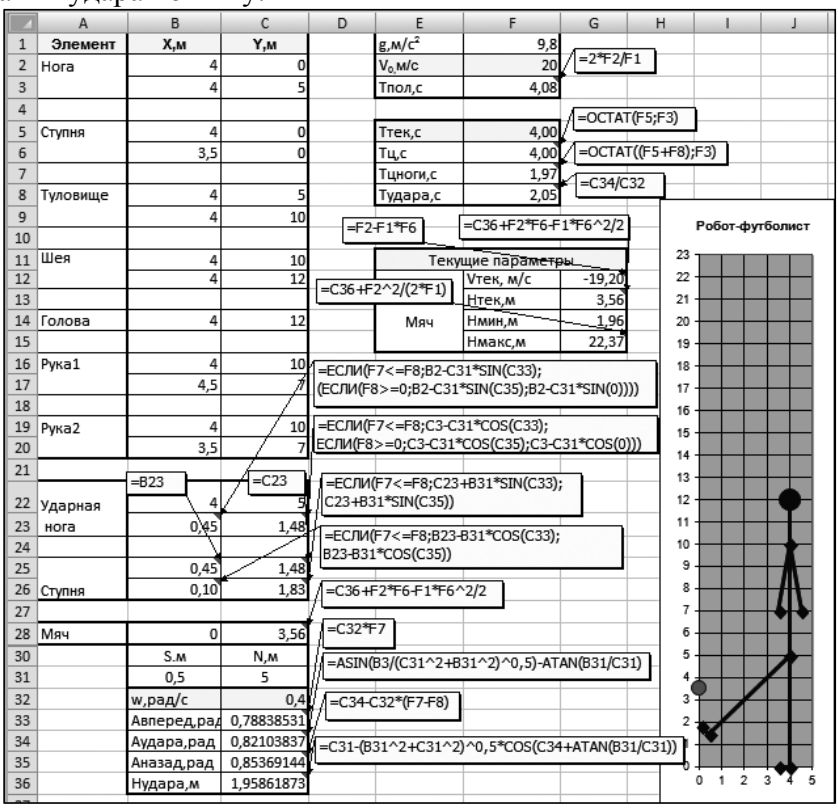
$$T_{\text{пол}} = 2 * V_0/g.$$

При известном $T_{\text{пол}}$ мяча уже можно организовать циклы жонглирования мячом.

Заносим в таблицу выражения для вычисления координат центра тяжести мяча (B28 и C28). Координата мяча по оси X равна нулю. Сформировав счетчик текущего времени T (F5) можем проверить правильность моделирования полета мяча, задавая различные значения счетчика текущего времени. Через контекстное меню точки мяча и пункт *Формат точки данных* можно задать подобающий размер, цвет и фон мяча.

После проверки моделирования полета мяча приступаем к моделированию движения ноги робота футболиста с применением формул (B22:C26).

Однако полет мяча и движение ноги не будут согласованы для моделирования удара по мячу.



	A	B	C	D	E	F	G	H
1	Элемент	X,м	Y,м		g,м/с²	9,8		
2	Нога	4	0		V_0,м/с	20		
3		4	5		Тпол,с	4,08		
4								
5	Ступня	4	0		Ттек,с	4,00		
6		3,5	0		Тц,с	4,00		
7					Тцноги,с	1,97		
8	Туловище	4	5		Тудара,с	2,05		
9		4	10					
10								
11	Шея	4	10		Текущие параметры			
12		4	12		Vтек, м/с	-19,20		
13					Мяч	Нтек,м	3,56	
14	Голова	4	12			Нмин,м	1,96	
15						Нмакс,м	22,37	
16	Рука1	4	10					
17		4,5	7					
18								
19	Рука2	4	10					
20		3,5	7					
21								
22	Ударная	4	5					
23	нога	0,45	1,48					
24								
25		0,45	1,48					
26	Ступня	0,10	1,83					
27								
28	Мяч	0	3,56					
30		S,м	N,м					
31		0,5	5					
32		w,рад/с	0,4					
33		Авперед,рад	0,78838531					
34		Аудара,рад	0,82103837					
35		Аназад,рад	0,85369144					
36		Нудара,м	1,95861873					

Остается согласовать движение ноги для нанесения удара по мячу в каждом цикле. Время цикла зависит от времени полета $T_{\text{пол}}$ мяча от точки удара $H_{\text{удара}}$. Во время цикла наноситься один удар по мячу. Таким образом, для формирования счетчика времени внутри цикла ($T_{\text{ц}}$) используем функцию ОСТАТ(T; $T_{\text{пол}}$). Вопрос точности вычисления координат мяча и положения ноги опускаем. С этого момента для моделирования полета мяча используется $T_{\text{ц}}$. Убеждаемся в правильности моделирования цикла полета мяча, который движется от точки $H_{\text{удара}}$ до $Y_{\text{макс}}$ и обратно. Теперь в момент достижения мяча точки удара нужно обеспечить совпадение положения ноги с мячом. Так как нога движется с определенной скоростью, то ее движение начинается за интервал времени $T_{\text{удара}}$ до момента равного $T_{\text{пол}}$, когда мяч достигает точки удара. Причем после удара нога должна

вернуться в исходное положение. В этом случае угол $\alpha_{\text{тек}}$ изменяется от $\alpha_{\text{удара}}$ до нуля. Сразу нужно отметить что при моделировании необходимо выполнить условие $T_{\text{пол}} > 2 * T_{\text{удара}}$, которое обеспечивает возврат ноги в исходное положение. Если условие не выполняется, то необходимо изменить угловую скорость ноги ω.

Для моделирования движения ноги также организуем счетчик времени цикла

$$T_{\text{цноги}} = \text{ОСТАТ}(T + T_{\text{удара}}; T_{\text{пол}}),$$

который смещен вперед относительно $T_{\text{ц}}$ на $T_{\text{удара}}$.

Занеся, соответствующие выражения и начальные данные можно приступить к моделированию жонглирования мячом путем изменения счетчика текущего времени *T*. На рисунке в примечаниях представлены реализованные выражения формул, которые были рассмотрены выше.

Следует отметить, что для упрощения расчетов при движении ноги предварительно вычисляется угол поворота при подъеме ноги

$$\text{А}_{\text{вперед}} = \omega * T_{\text{цноги}}.$$

При условии $T_{\text{цноги}} \leq T_{\text{удара}}$ в формулах расчета координат ноги и ступни $\alpha_{\text{тек}} = \text{А}_{\text{вперед}}$. Одновременно вычисляется угол поворота ноги назад в исходное положение

$$\text{А}_{\text{назад}} = \alpha_{\text{удара}} - \omega * (T_{\text{цноги}} - T_{\text{удара}}).$$

При условии $T_{\text{цноги}} > T_{\text{удара}}$ и $\text{А}_{\text{назад}} \geq 0$ в тех же формулах $\alpha_{\text{тек}} = \text{А}_{\text{назад}}$.

При отрицательном значении $\text{А}_{\text{назад}}$, когда исходное положение уже пройдено $\alpha_{\text{тек}} = 0$.

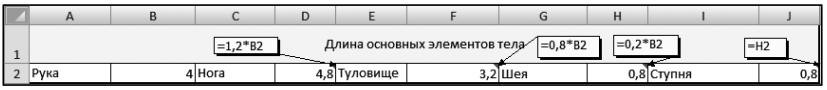
Изменения счетчика времени можно автоматизировать.

При разработке других моделей робота-футболиста интересно установить ворота на некотором расстоянии и смоделировать попадание в них мяча. При этом нужно направить вектор скорости мяча под определенным углом к линии горизонта.

2.2.11. Физическая зарядка для робота

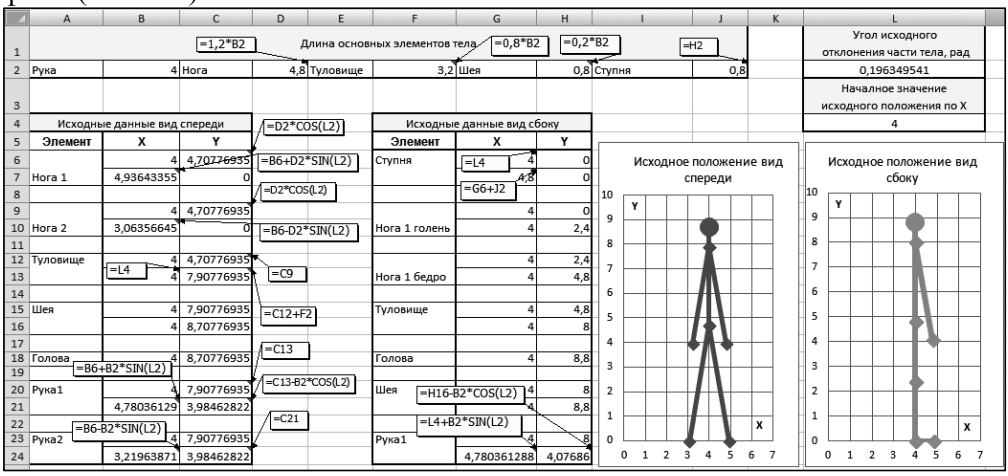
Создадим модель зарядки для робота. Пусть для простоты зарядка будет состоять из 2-х упражнений: поднятие рук в стороны, приседание.

Сначала определимся с параметрами тела робота. Все параметры тела для простоты будут безразмерными. За основу возьмем длину руки. Это упростит расчеты при движении частей тела. Так как в основном движение части тела есть вращение относительно какой-то неподвижной точки.



Пусть полная длина руки будет равна 4 условным единицам. Остальные параметры частей тела определяем через коэффициенты, которые можно увидеть в примечаниях таблицы длин основных элементов тела (B2, D2, F2, H2, J2).

Насколько удачно были определены пропорции частей тела убедимся при создании изображений исходных положений робота для выбранных упражнений. Для этого создадим таблицы координат отрезков необходимых частей тела и построим точечную диаграмму. Для первого упражнения создаем изображение в фас (B6:C24), а для второго – в профиль (G6:H24).

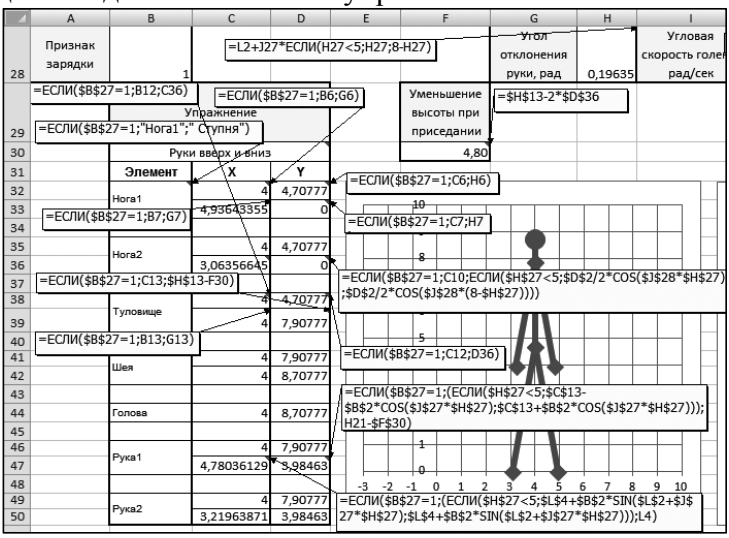


Так как руки или ноги могут находиться не в вертикальном положении, то задаем угол исходного отклонения части тела (L2) и координату начального положения по оси X (L4). Ясно, что робот стоит на земле и начальная координата по Y равна 0. При определении координат концов отрезков частей тела исходим из того, что ноги остаются на земле. Поэтому в первую очередь определяем координаты ноги (ступни). Если нужно было бы создать модель подтягивания, то нужно сначала определять координаты рук, а затем привязывать к ним последовательно координаты туловища, ног и т.д. При определении координат положения рук (ног) считаем угол исходного отклонения положительным от вертикальной оси. Формулы расчета можно увидеть в примечаниях. Для моделирования приседаний длину ноги разделили на равные половины для упрощения расчетов. Если посмотреть на изображения исходных положений робота, то можно сделать вывод о правильном выборе пропорций для частей тела. При необходимости можно их откорректировать. При этом нужно пом-

нить о первоначальном фиксировании значений осей координат и цены основных делений на графике, которые могут изменяться при внесении изменений в размеры робота.

Все готово для создания модели движения при выполнении упражнений. Создадим признак номера упражнений (B27), зададим угловую скорость для рук (J27) и голени ноги (J28). Также создаем счетчик тактов и счетчик по модулю 8-мь. Каждое упражнение разбиваем на равные половины по числу тактов. При этом подъем руки или приседание занимает 4 такта, 4 такта длиться возвращение в исходное положение. Можно задать и другие параметры. Ограничения по угловому положению частей тела не вводим.

Рассмотрим модель 1-го упражнения. Создаем таблицу с элементами частей тела. Для всех неподвижных координат частей тела заносим данные из таблицы исходного положения упражнения.



Движение рук есть вращение вокруг центра, который расположен на плечах. Координату кисти 1-й руки (см. рисунок) определяется формулами

$$X_{\text{рук}} = X_{\text{исх}} + R_{\text{рук}} * \sin(\alpha_{\text{исх}} + \alpha),$$

$$Y_{\text{рук}} = Y_{\text{пл}} - R_{\text{рук}} * \cos(\alpha_{\text{исх}} + \alpha),$$

где $X_{\text{исх}}$ - координата исходного положения,

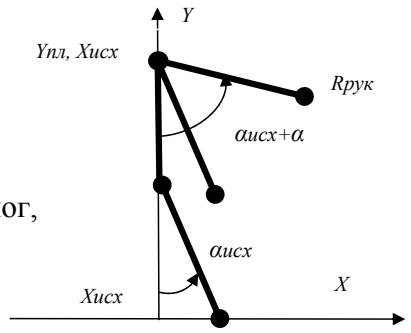
$Y_{\text{пл}}$ – координата плеча робота,

$R_{\text{рук}}$ - длина руки,

$\alpha_{\text{исх}}$ - угол исходного положения рук и ног,

$\alpha = \omega_{\text{рук}} * T$ – угол поворота руки,

$\omega_{\text{рук}}$ - угловая скорость вращения руки,



T - количество тактов по модулю 8-мь.

Для упрощения расчетов предварительно производим вычисления угла отклонения рук от вертикали в ячейке H28 по выражению L2+J27*ЕСЛИ(H27<5;H27;8-H27) с учетом татов подъема и опускания рук. Если число тактов счетчика меньше 5, то руки поднимаются, затем опускаются. Для организации циклического движения рук используем счетчик по модулю 8 (H27).

Для задания координат (X и Y) движения рук используем следующие выражения:

ЕСЛИ($B$27=1;(ЕСЛИ($H$27<5;$L$4+$B$2*SIN(H28);$L$4+$B$2*SIN(H28)));L4) и

ЕСЛИ($B$27=1;(ЕСЛИ($H$27<5;$D$39-$B$2*COS(H28);$D$39-$B$2*COS(H28)));H21-$F$30) (для первой руки, которая справа на изображении);

ЕСЛИ($B$27=1;(ЕСЛИ($H$27<5;$L$4-$B$2*SIN(H28);$L$4-$B$2*SIN(H28)));G24) и

ЕСЛИ($B$27=1;D47;H24-$F$30) (для второй руки).

При этом координата по Y для второй руки не вычисляется, а копируется координата первой руки. Координаты рук по Y равны.

Так как используем один массив для моделирования упражнений, то при смене признака номера упражнения функция ЕСЛИ формирует данные в ячейки для упражнения приседания. Также изменяются название упражнения и частей тела (см. примечания).

Теперь подробнее рассмотрим модель упражнения приседания. Здесь нужно правильно задать центр движения голени. Ясно, что центр находиться на земле в районе пятки. Длина голени и бедра равны. Будем моделировать не глубокие приседания. В этом случае при движении голени и бедра образуется равнобедренный треугольник. Пятка зафиксирована, а координаты коленки легко вычисляются по формулам

$$X_{\text{г}} = X_{\text{исх}} + R_{\text{г}} * \sin(\alpha),$$

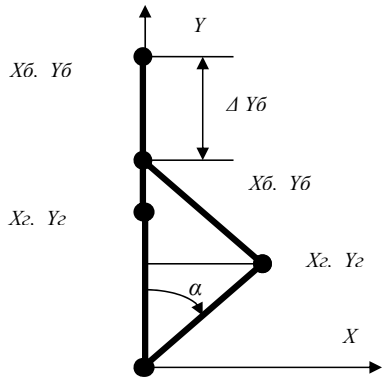
$$Y_{\text{г}} = R_{\text{г}} * \cos(\alpha),$$

где $X_{\text{исх}}$ - исходная координата,

$\alpha = \omega_{\text{г}} * T$ - угол поворота голени,

$\omega_{\text{г}}$- угловая скорость вращения голени,

T - количество тактов по модулю 8-мь.



Будем считать, что смещается туловище только вертикально на величину

$$\Delta Y_{\text{б}} = R_{\text{ног}} - 2 * Y_{\text{г}},$$

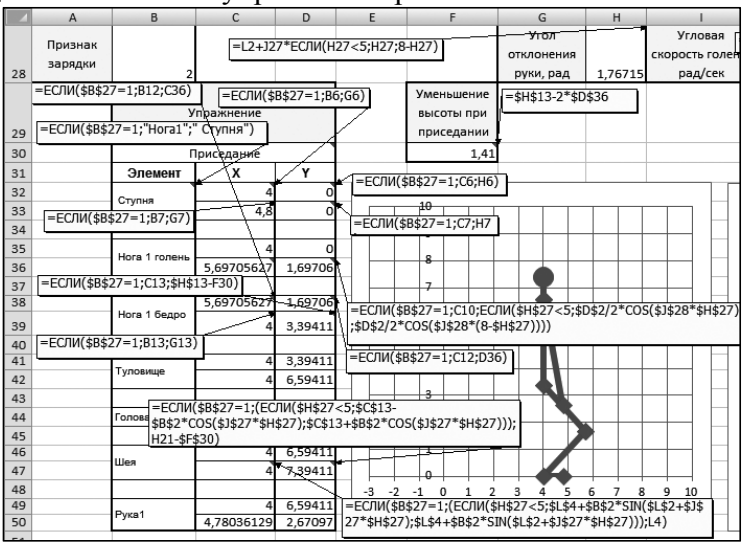
где $R_{\text{ног}}$ - общая длина ноги.

На эту величину смещаются и остальные части тела, которые находятся выше таза.

В примечаниях на изображении видны реализации этих формул с учетом признака номера упражнения (B27) и количества тактов (F27) непосредственно на приседание и возвращение в исходное положение. При возвращении в исходное положение для уменьшения угла α количество тактов вычисляем как (*8-T*).

Конечно, движение частей тела более сложные, но задача состояла в показе возможностей моделирования простейших движений робота.

Теперь можно, задавая номер упражнения и изменяя счетчик данных, наблюдать выполнение упражнений роботом.



В ходе моделирования необходимо собирать данные. Поэтому обеспечим построение траекторий движения в плоскости колена и таза тела в процессе выполнения упражнений. Сформируем массив для записи координат движения заданных частей тела на каждом такте. Массив может быть небольшой, так как каждое упражнение выполняется за 8 тактов. Но возникают определенные трудности в последовательном заполнении массива данными с использованием стандартных функций.

Создадим кнопки для автоматизации процесса моделирования и с программами заполнения массива нужными данными.

Кнопка под названием *Обнуление счетчика тактов и очистка массива параметров* имеет программу

```
Private Sub CommandButton1_Click()
Cells(27, 6) = 0 'обнуление счетчика тактов
Range("C55:D78") = Null 'очистка массива от данных
Cells(28, 2) = 0 'обнуление признака начала упражнения для зарядки
Cells(27, 2) = 1 'запись номера 1-го упражнения
Cells(29, 12) = Null 'очистка ячейки индикации конца зарядки
End Sub.
```

Массив параметров не должен содержать данные, которые используются для наглядного отображения на графике. Одновременно целесообразно записать координаты исходного положения частей тела. Далее аналогичным способом создаем кнопку *Начисление счетчика и запись параметров движения* с программой

```
Private Sub CommandButton2_Click()
If Cells(27, 6) > 23 Then
Cells(29, 12) = "Все!!!"
Else
If Cells(27, 2) = 1 Then 'определение номера упражнения
Cells(55 + Cells(27, 6), 3) = Cells(47, 3): Cells(55 + Cells(27, 6), 4) = Cells(47, 4)
'последовательная запись в массив координат кисти
Else
Cells(55 + Cells(27, 6), 3) = Cells(36, 3): Cells(55 + Cells(27, 6), 4) = Cells(36, 4)
'последовательная запись в массив координат коленки
End If
Cells(27, 6) = Cells(27, 6) + 1 'начисление счетчика тактов
End If
End Sub.
```

Можно создать управляющую кнопку *Зарядка*, которая обеспечит выполнение роботом комплекса упражнений. При этом обеспечим переключение с одного упражнения на другое и индикацию окончания зарядки. В этом случае программа выглядит следующим образом

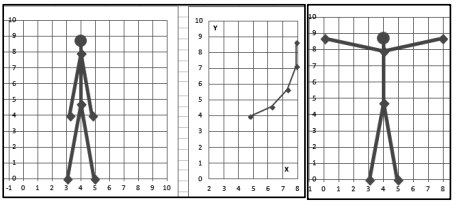
```
Private Sub CommandButton3_Click() 'Зарядка
If Cells(27, 6) > 31 Then
Cells(29, 12) = "Все. Водные процедуры!"
Else
If (Cells(28, 2) = O And Cells(27, 6) <= 16) Then 'определение номера упражнения
Cells(27, 2) = 1 'номер 1 упражнения
If Cells(28, 2) = 0 Then
Cells(28, 2) = 1 'формирование признака исходного положения 1 упражнения
Else
Cells(27, 6) = Cells(27, 6) + 1 'начисление счетчика тактов
End If
Else
If (Cells(28, 2) = 1 And Cells(27, 6) >= 16) Then
Cells(28, 2) = 2 'формирование признака исходного положения 2 упражнения
```

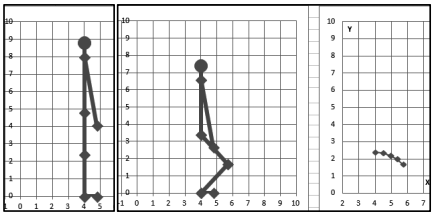
```
Cells(27, 6) = 16
Cells(27, 2) = 2 'номер 2 упражнения
Else
Cells(27, 6) = Cells(27, 6) + 1 'начисление счетчика тактов
End If
End If
End If
End Sub.
```

Данная процедура кнопки обеспечивает последовательное выполнение роботом 2-х упражнений по 2-а раза, а в конце вывод сообщения об окончании «*Все. Водные процедуры!*». Для обеспечения начальных условий выполнения зарядки в процедуру кнопки начального обнуления ввели обнуления ячейки B28, в которой формируется признак зарядки для организации исходных положений упражнений. При моделировании зарядки нужно первоначально щелкнуть на кнопке обнуления, а затем последовательно щелкать на кнопке зарядка до появления в ячейке L29 сообщения об окончании зарядки.

	B	C	D
52	Фиксация параметров		
53	Кисть руки 1		
54	Время (такт)	X	Y
55	0	4,78036129	3,98463
56	1	6,22228093	4,58189
57	2	7,32587845	5,68549
58	3	7,92314112	7,12741
59	4	7,92314112	8,68813
60	5	7,92314112	7,12741
61	6	7,32587845	5,68549
62	7	6,22228093	4,58189
63	8	4,78036129	3,98463
64	9	6,22228093	4,58189
65	10	7,32587845	5,68549
66	11	7,92314112	7,12741
67	12	7,92314112	8,68813
68	13	7,92314112	7,12741
69	14	7,32587845	5,68549
70	15	6,22228093	4,58189
71	16	4,78036129	3,98463
72	17	6,22228093	4,58189
73	18	7,32587845	5,68549
74	19	7,92314112	7,12741
75	20	7,92314112	8,68813
76	21		
77	22		
78	23		

На рисунках представлена траектория кисти руки при выполнении подъема рук. Вся информация о траектории записывается в массив (C55:D78) при щелчках по кнопке *Начисление счетчика и запись параметров движения*. В ячейку B27 *Признака номера упражнения* заносим 1.





	B	C	D
52	Фиксация параметров		
53	Коленка		
54	Время (такт)	X	Y
55	0	4	2,4
56	1	4,46821677	2,35388
57	2	4,91844024	2,21731
58	3	5,33336856	1,99553
59	4	5,69705627	1,69706
60	5	5,33336856	1,99553
61	6	4,91844024	2,21731
62	7	4,46821677	2,35388
63	8	4	2,4
64	9	4,46821677	2,35388

На следующих рисунках представлена траектория коленки при выполнении приседаний и часть массива данных по траектории. Перед записью траектории не забывайте производить обнуление исходных данных. В ячейку B27 *Признака номера упражнения* предварительно заносим 2.

2.2.12. Переход КА с одной круговой орбиты на другую

Задачи баллистики КА довольно «просто» решаются в орбитальной плоскости полета. При этом используется следующая основная формула для расчета радиуса-вектора R объекта в орбитальной плоскости

$$R = p/(1 + e * \cos(u - w)),$$

где p - фокальный параметр,

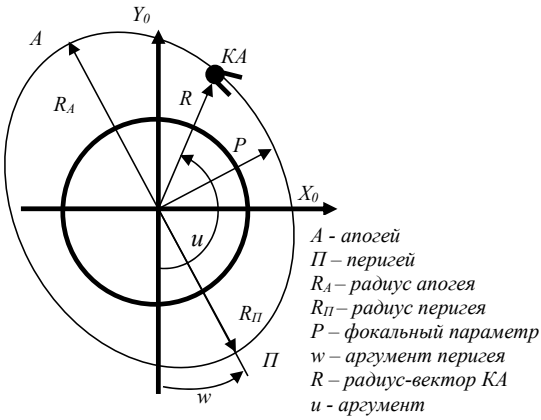
e - эксцентриситет,

u - аргумент;

w - аргумент перигея.

Имея значения R и u можно легко рассчитать координаты в прямоугольной системе координат (X_0,Y_0) в конкретный момент времени.

КА вращается вокруг Земли по круговой орбите на высоте 600 км. При значении аргумента $u_0 = 30^0$ включается двигатель для перехода на другую круговую орбиту с высотой 3000 км. Рассчитать параметры переходной орбиты движения КА и корректирующие импульсы скорости.



Занесем в ячейки C1, H1 известные константы.

В ячейки C2 и D2 записываем высоты круговых орбит. Вычисляем скорости КА (C3 и D2) на круговых орбитах по формуле

$$V_{\text{КРН}} = (\mu/(R_З + H_{\text{КР}}))^{0,5},$$

где $\mu = 398600$ км^3/с^2 - гравитационная постоянная Земли,

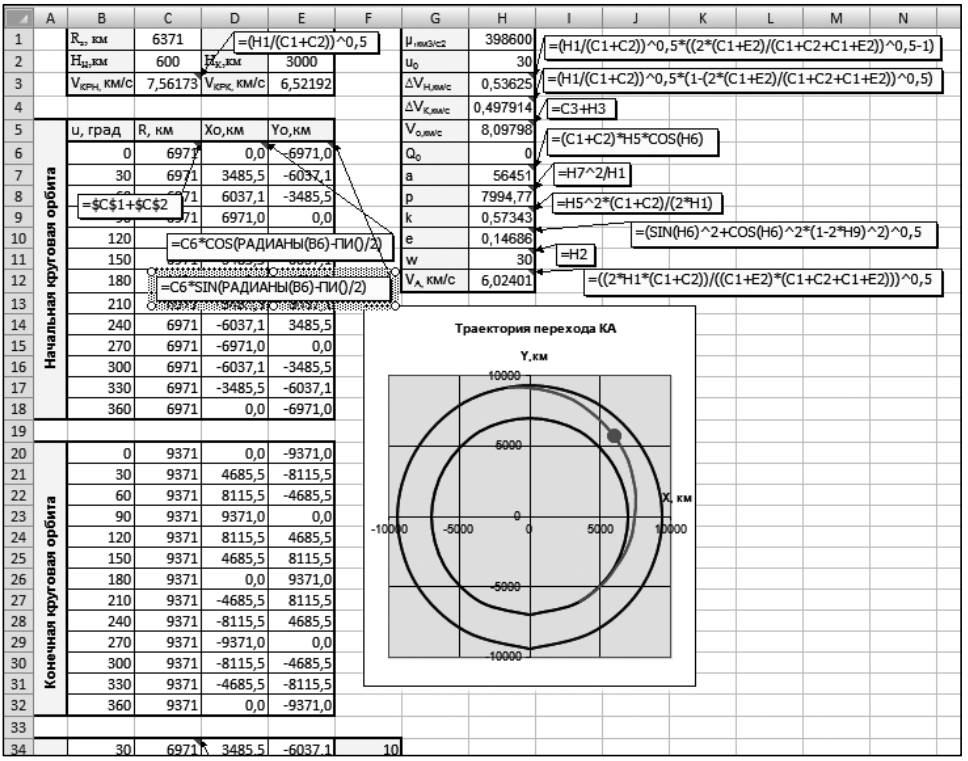
$R_З = 6371$ км – радиус Земли,

$H_{\text{КР}}$ - высота круговой орбиты КА.

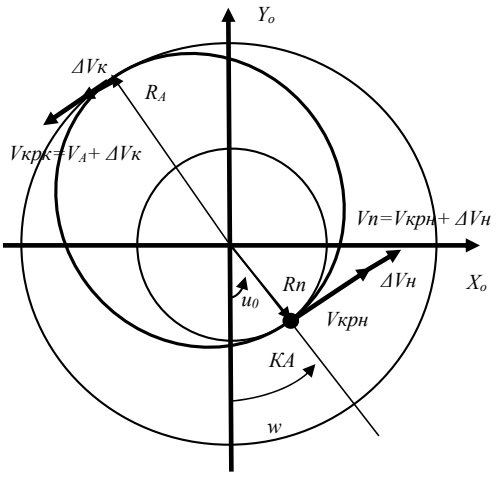
Построим начальную и конечную круговые орбиты по вышеприведенным методикам. Достаточно рассчитать 12 координат точек траектории для ее построения.

Аргумент записываем от 0^0 с шагом 30^0. Вектор-радиус орбит постоянен и определяется только высотой КА. Пересчет координат из орбитальной в прямоугольную систему делаем по формулам

$$X_0 = R * \cos(u - 90^0), Y_0 = R * \sin(u - 90^0).$$



Контур Земли не рассчитываем, чтобы не усложнять рисунок. Выделяем ячейки D6:E52 с координатами для изображения двух круговых траекторий. В выделенный диапазон дополнительно включим 19 ячеек, в которых рассчитаем координаты переходной траектории. По мере производства расчетов будем наблюдать появление переходной траектории на диаграмме.



Рассматриваемый двухимпульсный переход КА называется переходом Гомана.

При этом рассчитываются корректирующие импульсы скорости по следующим формулам

$$\Delta V_{\text{Н}} = (\mu/R_{\text{П}})^{0,5} * ((2 * R_{\text{А}}/(R_{\text{П}} + R_{\text{А}}))^{0,5} - 1),$$
$$\Delta V_{\text{К}} = (\mu/R_{\text{А}})^{0,5} * (1 - (2 * R_{\text{П}}/(R_{\text{П}} + R_{\text{А}}))^{0,5}),$$
(2.19)

где $R_{\text{П}} = R_{\text{З}} + H_H$ - радиус-вектор перигея переходной орбиты,

$R_{\text{А}} = R_{\text{З}} + H_{\text{К}}$ - радиус-вектор апогея переходной орбиты,

H_H и $H_К$ - высоты начальной и конечных орбит соответственно.

В момент аргумента КА равного u_0 (H2), включается управляющий двигатель и круговая скорость $V_{КРН}$ возрастает на $\Delta V_Н$ (H3).

Угол наклона вектора скорости $V_П = V_{КРН} + \Delta V_Н$ (H5) к линии местного горизонта равен нулю (H6). В силу особенности данного перехода аргумент перигея $w = u_0$ (H11).

Далее рассчитываем параметры переходной орбиты (H7:H10). Так как в перигее переходной орбиты скорость КА (H12) увеличивается на $\Delta V_К$ (H4) до значения круговой скорости конечной орбиты, то остается вычислить координаты траектории для значений аргумента от u_0 до $u_0 + 180^0$ с шагом 10^0 (B34:B52). После вычислений увидим на диаграмме переходную траекторию.

Однако особый интерес представляет моделирование движения КА по орбите в зависимости от времени. Для этого нужно в конкретный момент времени определить значение аргумента КА. Воспользуемся для этого уравнением Кеплера

$$E - e * \sin(E) = \mu^{0,5}/a^{1,5} * (T - \tau),$$

где E - эксцентрическая аномалия,

e - эксцентриситет,

a - большая полуось,

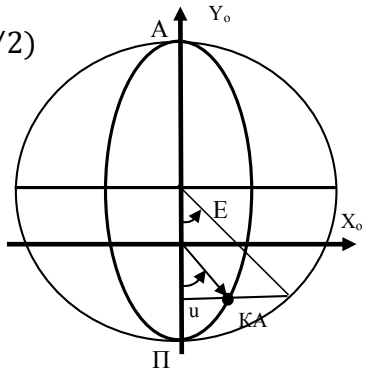
T - текущее время,

τ - время прохождения перигея.

Эксцентрическая аномалия связана с истинной аномалией (аргументом) известной формулой

$$tg(E/2) = ((1 - e)/(1 + e))^{0,5} * tg(u/2)$$

.



В точке перигея (П) $\tau = 0$ и $u = E = 0$.

В точке апогея (А) $u = E = 180^0$ и время полета от перигея равна половине периода обращения КА. В нашем случае возьмем $\tau = 0$. Задаем счетчик времени Т (G51) с шагом 50 с (H51). Для заданного значения счетчика времени нужно вычислить эксцентрическую аномалию. Вычисление производим методом последовательных итераций. Достаточно сделать 6 шагов (H53:H58) для достижения нужной точности. Нулевое значение эксцентрической аномалии рассчитываем по формуле

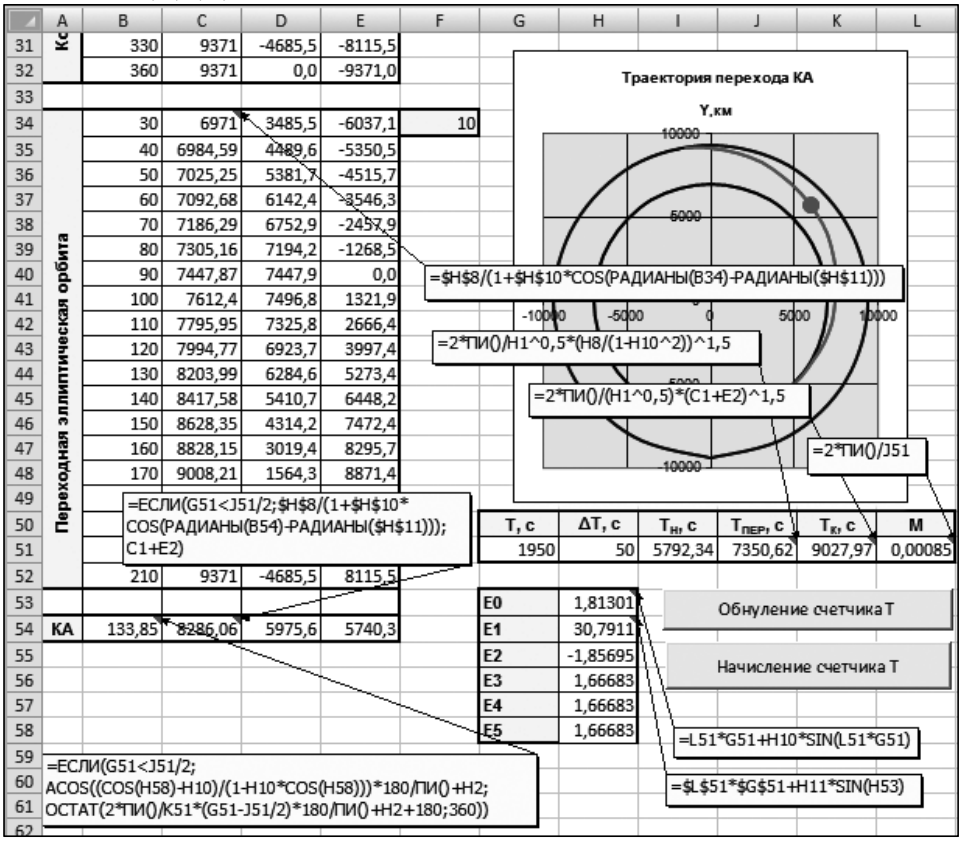
$$E_0 = M + e * \sin(M),$$

где $M = \mu^{0,5}/a^{1,5} * T$ - средняя аномалия.

На следующих шагах используем формулу

$$E_i = M + e * \sin(E_{i-1}),$$

где i=1,2,3,4,5.



	A	B	C	D	E	F
31	Кс	330	9371	-4685,5	-8115,5	
32		360	9371	0,0	-9371,0	
33						
34		30	6971	3485,5	-6037,1	10
35		40	6984,59	4489,6	-5350,5	
36		50	7025,25	5381,7	-4515,7	
37		60	7092,68	6142,4	-3546,3	
38		70	7186,29	6752,9	-2457,9	
39		80	7305,16	7194,2	-1268,5	
40		90	7447,87	7447,9	0,0	
41		100	7612,4	7496,8	1321,9	
42		110	7795,95	7325,8	2666,4	
43		120	7994,77	6923,7	3997,4	
44		130	8203,99	6284,6	5273,4	
45		140	8417,58	5410,7	6448,2	
46		150	8628,35	4314,2	7472,4	
47		160	8828,15	3019,4	8295,7	
48		170	9008,21	1564,3	8871,4	
49	Переходная эллиптическая орбита					
50						
51						
52		210	9371	-4685,5	8115,5	
53						
54	КА	133,85	8286,06	5975,6	5740,3	
55						
56						
57						
58						
59						
60						
61						
62						

T, с	ΔT, с	$T_{Н}$, с	$T_{ПЕР}$, с	$T_{К}$, с	M
1950	50	5792,34	7350,62	9027,97	0,00085

E0	1,81301
E1	30,7911
E2	-1,85695
E3	1,66683
E4	1,66683
E5	1,66683

Из ячейки H58 выбираем значение эксцентрической аномалии для дальнейших расчетов. Остается найти истинную аномалию и затем аргумент. Функция тангенса «коварная» с точки определения углов вне первой четверти. Поэтому не будем делать расчеты по вышеприведенной формуле, а воспользуемся другой формулой

$$\cos(u) = (\cos E - e)/(1 - e * \cos(E)).$$

В ячейке B54 делаем расчет аргумента с учетом аргумента перигея (H2). Чтобы расчет можно было производить на конечной круговой орбите, делаем проверку схода КА с переходной орбиты по превышению счетчика времени значения, равного половине периода обращения на переходной орбите.

При расчете аргумента на круговой орбите учитываем 180^0, который проходит КА от перигея до апогея, и аргумент перигея. Сам аргумент отсчитываем от точки апогея по формуле

$$u = 2 * \pi/T_{\text{к}} * (T - T_{\text{пер}}/2).$$

По расчетному значению аргумента находим радиус-вектор КА (C54) на переходной и круговой орбите. Затем определяем прямоугольные координаты КА (D54,E54), которые используем для изображения КА в виде маркера. При изменении значения текущего времени наблюдаем движение КА по переходной и конечной круговой орбите.

Вносить изменения текущего времени в ячейку G51 вручную не совсем удобно. Автоматизируем этот процесс путем создания кнопок *Обнуление счетчика T* и *Начисление счетчика T*. Для кнопок вносим следующие программы:

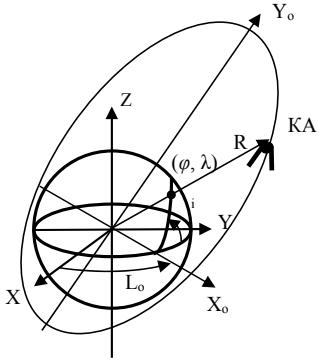
```
Private Sub CommandButton1_Click()
Cells(51, 7) = 0
End Sub
Private Sub CommandButton2_Click()
Cells(51, 7) = Cells(51, 7) + Cells(51, 8)
End Sub.
```

Первая программа обнуляет счетчик, а вторая начисляет значение счетчика с шагом, значение которого записано в ячейке H51.

2.2.13. Трехмерное моделирование полета КА

Создадим трехмерную модель полета КА по круговой орбите вокруг Земли с заданной высотой, наклонением орбиты и долготой восходящего узла.

Для расчета и построения трасс объектов необходимо уметь рассчитывать траекторию полета в земной системе координат, которая привязана к Земле, а так же знать формулы пересчета координат траекторий в географическую систему координат.



i – наклонение орбиты
L_0 – долгота восходящего узла
φ, λ - географические координаты точки трассы

Процесс моделирования целесообразно разбить на несколько этапов:

1. Построение каркасной модели Земли.
2. Моделирование вращения Земли.

3. Моделирование полета спутника вокруг Земли.
4. Автоматизация моделирования полета спутника.

Для моделирования каркаса Земли воспользуемся уравнениями сферы

$$X = R_З * \cos(\varphi) * \cos(\lambda),$$
$$Y = R_З * \cos(\varphi) * \sin(\lambda),$$
$$Z = R_З * \sin(\varphi),$$

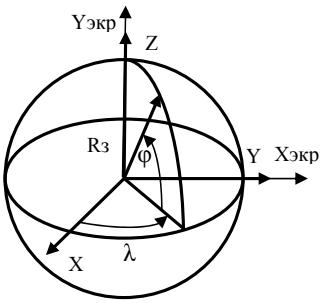
где $R_З = 6371$ км - радиус Земли,

φ - широта, значения которой изменяются от -90^0 до 90^0,

λ - долгота, значения которой изменяются от 0^0 до 360^0.

Координаты сферы Земли рассчитываем в инерциальной системе координат *XYZ*. Центр Земли совпадает с началом системы координат. В начальный момент моделирования координаты земной системы координат совпадают с координатами инерциальной системы.

В качестве каркаса возьмем окружности широт и долгот, для построения которых будем использовать по 19 точек. При построении окружностей широт в качестве параметра берем широту φ, значения которой задаются величинами: 0^0, 30^0, 60^0, -30^0, -60^0, а долготы точек λ задаются с шагом 20^0 от 0^0 до 360^0. Окружности долгот строим при заданных значениях долгот λ: 45^0, 90^0, 135^0 и 180^0. При этом широты φ точек берем с шагом 20^0. Для трехмерного изображения Земли воспользуемся косоугольной горизонтальной изометрической проекцией. В данной проекции ось *Z* направлена вертикально, между осями *X* и *Y* угол равен 90^0, а между осью *Z* и *Y* угол равен 135^0. В этом случае в экранной системе координат Хэкр Yэкр координаты точек определяются по простым формулам



$$X_{ЭКР} = R * \cos(\beta),$$
$$Y_{ЭКР} = Z - R * \sin(\beta),$$

где $R = (X^2 + Y^2)^{0,5}$,

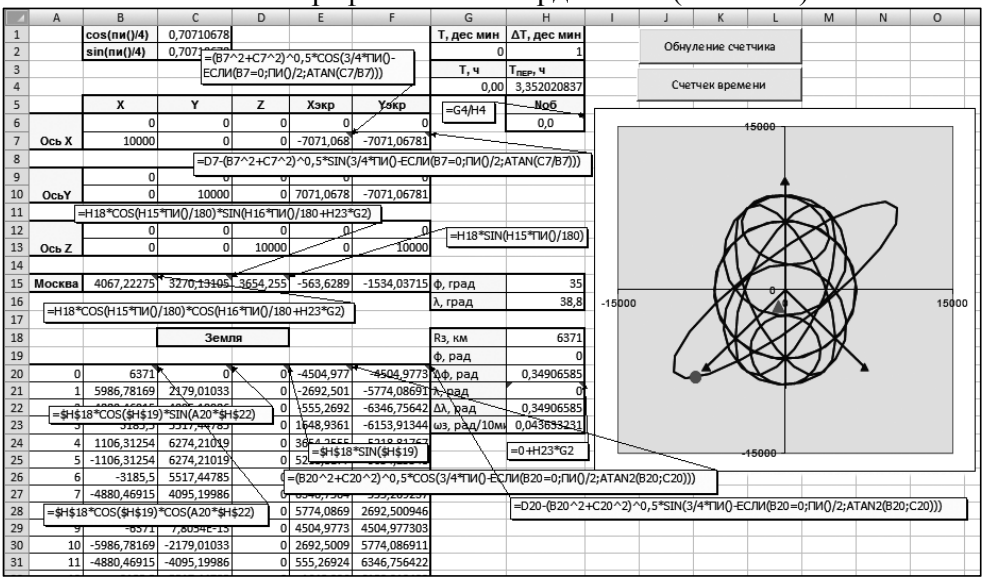
$\beta = 135^0 - arctg(Y/X)$.

Сначала изобразим оси изометрической проекции прямыми линиями по координатам 2-х точек, принадлежащим соответствующим осям. Начальные координаты соответствуют началу координат (B6:D6, B9:D9, B12:D12) и равны нулю. Координата второй точки для осей возьмем равными 10000км по соответствующей координате (B7:D7, B10:D10, B13:D13).

По формулам вычисляем координаты точек осей в экранной системе координат (E6:F13). Для исключения деления на ноль производим провер-

ку при расчете значения arctg (см. примечание). Чтобы оси изображались отдельно строки с координатами разделяем пустыми строками. В дальнейшем создаем маркеры в виде стрелок на концах осей.

Для наглядности сформируем координаты (B15:F15) точки расположения г. Москва по географическим координатам (H15:H16).



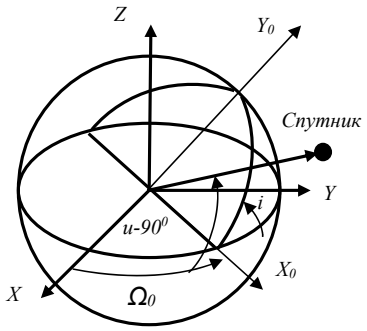
Далее рассчитываем по формулам координаты экваториальной окружности Земли при $\varphi = 0^0$ (B20:F38) по исходным данным, записанным в ячейках H18:H22. В столбце A занесены значения параметра i от 0 до 18. С помощью параметра рассчитывают значения долгот точек окружности $\lambda_i = i * \Delta\lambda$.

При вычислении экранных координат используем функцию ATAN2(X;Y) для расчета углов от –π до π. Затем рассчитываем координаты других окружностей широт и долгот с указанными выше параметрами. Для каждой окружности отводится по 19 строк с пропуском одной строки. Теперь можно выделить столбцы с экранными координатами и через М*астер диаграмм* построить изображение Земли в изометрии. Тип диаграммы выбираем *Точечный*. Производим необходимое форматирование. Точке изображения г. Москва на диаграмме формируем треугольный маркер красного цвета.

Земля вращается вокруг оси *Z* инерциальной системы координат с угловой скоростью $\omega_3 = 7{,}292 * 10^{-5}$рад/с. При этом долготы точек увеличиваются на $\omega_3 * (T - T_0)$ в зависимости от значения текущего времени T и значения начального времени T_0. Возьмем для простоты $T_0 = 0$ и единицу времени приравняем 10 минутам. В ячейке H23 занесем значение угловой скорости, рассчитанной за 10 минут.

В ячейке H21 рассчитываем значение долготы точки с учетом начального значения долготы и ее изменение за счет вращения по выражению

$$\lambda = \lambda_i + \omega_З * T.$$



В ячейке G2 формируем значения счетчика времени в десятках минут, а в G4 – в часах. При изменении счетчика каркас Земли и маркер г. Москва начинают вращение вокруг оси *Z* против часовой стрелки.

Теперь можно перейти к моделированию полета КА. КА вращается по круговой орбите, с заданным наклонением.

В этом случае значительно упрощаются расчеты положения спутника в орбитальной системе координат. Движение КА происходит в инерциальной системе координат. Ориентация плоскости орбиты при этом остается неизменной. Рассчитаем координаты орбиты, а затем координаты положения КА в зависимости от значения текущего времени. Изображение орбиты осуществим по 19 координатным точкам.

Записываем высоту H спутника (I193) и наклонение орбиты *i* в градусах (I194). Для удобства дальнейших вычислений сразу пересчитываем наклонение i из градусов в радианы (J194). Долготу восходящего узла Ω_0 возьмем равную нулю (I195). В ячейку I197 занесем гравитационный параметр Земли μ.

Рассчитаем круговую скорость КА $V_{\text{КР}}$ по формуле $V_{\text{КР}} = (\mu/(R_З + H_{\text{КР}}))^{0,5}$ (I196) и период обращения Тпер по формуле $T_{\text{ПЕР}} = 2 * \pi * (R_З + H)^{1,5}/\mu^{0,5}$ в секундах (I198) и часах (J198).

Приступаем к расчетам координат орбиты в орбитальной системе координат X_0Y_0. В ячейках G203: G221 заносим с шагом 20^0 аргумент u от 0^0 до 360^0 и в ячейки H203:H221 значение величины радиуса-вектора $R = R_З + H$. Далее рассчитываем прямоугольные координаты спутника в орбитальной системе координат по формулам

$$X_0 = R * \cos(u - 90^0),$$
$$Y_0 = R * \sin(u - 90^0).$$

В ячейках B203:D221 вычисляем координаты *XYZ* орбиты в инерциальной системе координат по формулам

$$X = \cos(\Omega_0) * X_0 - \sin(\Omega_0) * \cos(i) * Y_0,$$
$$Y = \sin(\Omega_0) * X_0 + \cos(\Omega_0) * \cos(i) * Y_0,$$
$$Z = \sin(i) * Y_0.$$

Затем координаты из инерциальной системы пересчитываются в экранную систему координат. Увеличив диапазон данных (E6:F223) дополнительно

получим изображение орбиты спутника, а затем изображение самого спутника.

Осталось определить координаты КА в зависимости от текущего времени. Положение КА определяется аргументом u и величиной радиуса-вектора R. Величина радиуса-вектора постоянна, а аргумент определяется угловой скоростью КА и вычисляется по формуле

$$u = 2 * \frac{\pi}{T_{\text{ПЕР}}} * 600 * T + u_0 \text{ (G223).}$$

Начальное значение аргумента u_0 возьмем равным 90^0. Это соответствует начальному положению КА в экваториальной плоскости Земли на оси X. Умножение на число 600 связано с тем, что счетчик времени выдает количество десятков минут. Затем осуществляем пересчет координат КА в экранную систему координат. Далее формируем маркер для координаты КА в виде круга с увеличенным размером. Все готово для моделирования. При изменении счетчика времени *T* наблюдаем вращение Земли, перемещение маркера г. Москва и движение КА по орбите.

	A	B	C	D	E	F	G	H	I	J
190	10	4233,29393	-4233,29393	-2179,01	-5986,782	-2179,01033				
191	11	3451,01283	-3451,01283	-4095,2	-4880,469	-4095,19986				
192	12	2252,48865	-2252,48865	-5517,45	-3185,5	-5517,44785				
193	13	782,281099	-782,281099	-6274,21	1106,3125	-6274,21019				
194	14	-782,281099	782,281099	-6274,21	1106,3125	-6274,21019				
195	15	-2252,48865	2252,48865	-5517,45	3185,5	-5517,44785				
196	16	-3451,01283	3451,01283	-4095,2	4880,469	-4095,19986				
197	17	-4233,29393	4233,29393	-2179,01	5986,782	-2179,01033				
198	18	-4504,9773	4504,9773	-1,6E-12	6371	-1,5611E-12				
199										
200			КА							
201										
202							u, град	R, км	Xo,км	Yo,км
203		6,9656E-13	-8710,69136	-7309,14	-6159,389	-1149,74898	0	11371	0,0	-11371,0
204		3889,11105	-8185,3724	-6868,34	-8537,949	-3830,42743	20	11371	3889,1	-10685,2
205		7309,13791	-6672,77671	-5599,12	-9886,707	-6049,0998	40	11371	7309,1	-8710,7
206		9847,57487	-4355,34568	-3654,57	-10942,98	-7538,16146	60	11371	9847,6	-5685,5
207		11198,249	-1512,59568	-1269,22	-10987,24	-8118,00959	80	11371	11198,2	-1974,6
208		11198,249	1512,59568	1269,218	-6848,791	-7718,70596	100	11371	11198,2	1974,6
209		9847,57487	435				120	11371	9847,6	5685,5
210		7309,13791	6672,77671	5599,124	-449,9753	-4287,58216	140	11371	7309,1	8710,7
211		3889,11105	8185,3724	6868,343	3037,9155	-1669,60617	160	11371	3889,1	10685,2
212		6,9656E-13	8710,69136	7309,138	6159,3889	1149,748978	180	11371	0,0	11371,0
213		-3889,11105	8185,3724	6868,343	8537,9491	3830,427427	200	11371	-3889,1	10685,2
214		-7309,13791	6672,77671	5599,124	9886,7066	6049,099796	220	11371	-7309,1	8710,7
215		-9847,57487	4355,34568	3654,569	10042,981	7538,161455	240	11371	-9847,6	5685,5
216		-11198,249	1512,59568	1269,218	8987,9244	8118,009592	260	11371	-11198,2	1974,6
217		-11198,249	-1512,59568				280	11371	-11198,2	-1974,6
218		-9847,57487	-4355,34568				300	11371	-9847,6	-5685,5
219		-7309,13791	-6672,77671	-5599,12	449,97532	4287,582165	320	11371	-7309,1	-8710,7
220		-3889,11105	-8185,3724	-6868,34	-3037,916	1669,606165	340	11371	-3889,1	-10685,2
221		-2,0897E-12	-8710,69136	-7309,14	-6159,389	-1149,74898	360	11371	0,0	-11371,0
222										
223		11371	0	0	-8040,511	-8040,51121	90	11371	11371,0	0,0

=ПИ()/180*I194

=(I197/(H18+I193))^0,5

=I198/3600

=2*ПИ()/I197^0,5*(H18+I193)^1,5

$H_{,км}$	5000
$i_{,град}$	40
$\Omega_{0,град}$	0
$V_{КР,\ км/с}$	5,92065
$\mu_{,км3/с2}$	398600
$T_{ПЕР,\ с}$	12067,3

=0,69813

=0

=3,35202

=H203*SIN(РАДИАНЫ(G203)-ПИ()/2)

=H203*COS(РАДИАНЫ(G203)-ПИ()/2)

=SIN($J$194)*J20

=$H$18+$I$1

=COS($J$195)*I203-COS($J$194)*SIN($J$195)*J203

=SIN($J$195)*I203+COS($J$194)*COS($J$195)*J203

=(2*ПИ()/I198*600*G2+ПИ()/2)*180/ПИ()

Задав высоту спутника около 36000 км получим модель стационарного КА. При вращении Земли маркер КА будет висеть над определенной точкой поверхности.

Для автоматизации начисления счетчика времени создадим кнопки. Для кнопки *Счетчик времени* вносится текст программы:

```
Private Sub CommandButton1_Click()
Cells(2, 7) = Cells(2, 7) + Cells(2, 8)
End Sub,
```

а для кнопки *Обнуление счетчика* следующий текст:

```
Private Sub CommandButton2_Click()
Cells(2, 7) = 0
End Sub.
```

После снятия режима *Конструктор* используем кнопки при моделировании. В ячейке H9 рассчитываем число оборотов спутника вокруг Земли.

2.3. Оптимизационные модели

2.3.1. Кратчайший путь между пунктом отправления и назначения

Необходимо найти кратчайший путь между пунктами отправления и назначения с учетом дорожной сети, и отобразить путь на географической карте.

Алгоритмы нахождения кратчайшего пути между двумя пунктами известны. Основная трудность заключается в подготовке необходимых данных для нахождения кратчайшего пути и его отображения на географической карте.

Решение задачи можно разбить на несколько следующих этапов:

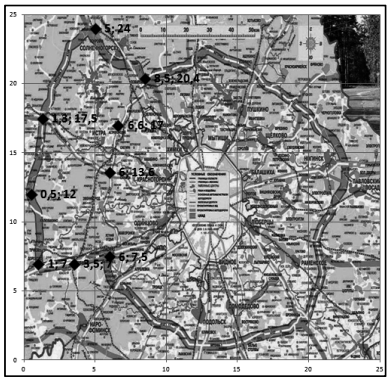
1. Нахождение изображения географической карты с сетью дорог между необходимыми пунктами.
2. Определение опорных пунктов на сети дорог.
3. Вычисление расстояния между опорными пунктами.
4. Установление кратчайшего пути между пунктом отправления и назначения.
5. Графическое изображение на географической карте кратчайшего пути.

Географическую карту нужного участка местности можно без труда найти в интернете. При необходимости с помощью программы Paint вырезаем необходимую часть карты и сохраняем для дальнейшего использования при решении задачи. Главное, чтобы хорошо просматривалась сеть дорог между нужными пунктами.

	B	C	D
2		Координаты	
3	№ пункта	X	Y
4	1	6	7,5
5	2	3,5	7
6	3	6	13,6
7	4	1	7
8	5	0,5	12
9	6	1,3	17,5
10	7	6,6	17
11	8	8,5	20,4
12	9	5	24
13	10		
14	11		

Далее определяем опорные пункты маршрута. Пункты расставляем не только в населенных пунктах, но и на всех изгибах дорог для более детальной прорисовки кратчайшего пути.

Для определения опорных пунктов необходимо выделить диапазон ячеек, в которые должны быть занесены координаты пунктов (C4:D14). Выделив этот диапазон ячеек, обращаемся к вставке диаграммы *Точечная с маркерами*. После появления диаграммы через контекстное меню выходим на *пункт Формат области построения*. В окне через *Заливку* включаем *Рисунок или текстура*. Далее находим файл географической карты

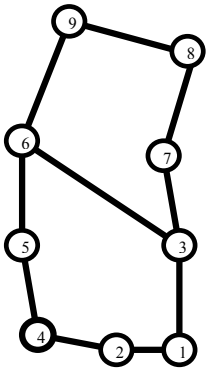


(*Вставить из: Файл*) и делаем его фоном в области построения диаграммы.

Теперь вносим координаты опорных пунктов и при необходимости их корректируем при несовпадении маркера с нужной точкой на карте. Чтобы не изменялся размер области построения нужно установить фиксированные максимальные значения параметров оси в окне *Формат оси* для оси X и Y. Для удобства задания опорных пунктов через контекстное меню осей целесообразно добавить основные линии сетки. Первый опорный пункт должен соответствовать месту отправки, а последний пункт - месту назначения. Промежуточные пункты должны иметь нарастающие номера по мере приближения к пункту назначения. Этим обеспечиваются условия нахождения кратчайшего пути. На карте пункты изображаем в виде маркеров с координатами. На данный момент координаты представлены в виде безразмерных условных единицах.

Прежде чем вычислить расстояние между пунктами необходимо сформировать матрицу смежности, которая обеспечивает взаимосвязь между пунктами в соответствии с дорожной сетью. Матрица смежности является квадратной матрицей (G4:D12), где количество строк и столбцов определяется числом опорных пунктов. В нашем случае всего 9 опорных пунктов. При наличии дороги между пунктами на пересечении соответствующей строки и столбца заносим единицу, а при отсутствии дороги – ноль. То есть, элемент матрицы V_{ij}=1 при наличии дороги и V_{ij}=0 при отсутствии дороги. Матрица смежности соответствует графу, который представлен на рисунке.

	F	G	H	I	J	K	L	M	N	O
1					Матрица смежности					
2										
3	i \ i	1	2	3	4	5	6	7	8	9
4	1		1	1						
5	2	1			1					
6	3	1						1		
7	4		1			1				
8	5				1		1			
9	6			1						1
10	7			1					1	
11	8							1		1
12	9						1		1	



Целесообразно предусмотреть в таблице координат опорных пунктов и матрице смежности увеличение количества пунктов. На этом заканчивается работа по внесению необходимых исходных данных.

Математическая модель решения задачи по поиску кратчайшего пути состоит из задания целевой функции и ограничений. Модель соответствует задаче линейного программирования об оптимальном потоке при перемещении груза из исходного пункта в конечный пункт.

Целевая функция

$$F = \sum_{i=1,j=1}^{i=9,j=9} Rij * Xij \rightarrow min,$$

где Rij- расстояние между опорными пунктами,

Xij – переменные, которые определяют взаимосвязь опорных пунктов кратчайшего пути,

i=1,…,9 – индекс исходящего опорного пункта,

j=1,…,9 – индекс входящего опорного пункта.

При решении задачи должны быть учтены следующие ограничения:

- Переменные Xij должны быть целыми, $Xij \geq 0$ и равны 1.
- Сумма

$$\sum_{i=1,j=1}^{j=2,\ldots,9} Xij = 1$$

для всех строк i=1,…,8.

- Сумма

$$\sum_{i=1,j=2}^{i=1,\ldots,8} Xij = 1$$

для всех столбцов j=2,…,9.

Последние два ограничения обеспечивают перемещение груза из опорного пункта только в один опорный пункт.

На основании таблиц координат пунктов и матрицы смежности произведем расчеты расстояний между связными пунктами, которые представим в виде матрицы транспортных расходов. Расходы определяются расстояниями между пунктами. Если нет дороги между пунктами, то расстояние равно нулю. Также устанавливаем расстояние равное нулю для диагональных элементов матрицы.

Матрица транспортных расходов

=ЕСЛИ(I4=1;(($C4-ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;I4:$O4;0)+T$3+2;$B$4+2;4)))^2+($D4-ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;I4:$O4;0)+T$3+2;$B$4+3;4)))^2)^0,5;0)

i \ j	1	2	3	4	5	6	7	8	9
1	0	2,54951	6,1	0	0	0	0	0	0
2	2,549509757	0	0	2,5	0	0	0	0	0
3	6,1	0	0	0	0	0	3,45254	0	0
4	0	2,5	0	0	5,02494	0	0	0	0
5	0	0	0	5,02494	0	5,55788	0	0	0
6	0	0	10,448	0	0	0	0	0	0
7	0	0	3,45254	0	0	0	0	3,894868419	0
8	0	0	0	0	0	0	3,89487	0	5,02096
9	0	0	0	0	0	0	0	7,762087348	0

Вычисление расстояния между пунктом i и j производим при выполнении условия $V_{ij} = 1$, где V_{ij} является элементом матрицы смежности, по формуле $R_{ij} = ((X_i - X_j)^2 + (Y_i - Y_j)^2)^{0,5}$. Если $V_{ij} = 0$, то присваиваем соответствующей ячейке нуль.

В ячейке R4 для расчета расстояния создано следующие выражение =ЕСЛИ(G4=1;(($C4-ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;G4:$O4;0)+R$3+2; $B$4+2;4)))^2+($D4-ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;G4:$O4;0)+R$3+2; $B$4+3;4)))^2)^0,5;0).

Выражение создано с учетом использования в ячейках всей 1-ой строки. Строка соответствует 1-му опорному пункту. Если в ячейке матрицы смежности G4 записан нуль, то $R_{11} = 0$. Иначе необходимо опреде-

лить связный j пункт с 1-м пунктом и определить его координаты для расчета расстояния R_{1j}.

Для поиска используем функцию ПОИСКПОЗ(1;G4:$O4;0), где 1 является элементом поиска, G4:$O4 – диапазон поиска значения 1. Нуль определяет нахождение первого значения 1. Функция ПОИСКПОЗ выдает относительную позицию элемента массива со значением равным 1. Эта позиция соответствует значению j, которое определяет связный j пункт с 1-м пунктом.

При получении значения j позиции можно далее использовать функцию АДРЕС(ПОИСКПОЗ(1;G4:$O4;0)+R$3+2; $B$4+2;4). Данная функция позволяет определить адрес ячейки на листе, для которой указаны номера строки и столбца. Номер строки определяется аргументом ПОИСКПОЗ(1;G4:$O4;0)+R$3+2. Номер столбца задается аргументом $B$4+2. Аргумент 4 обеспечивает относительную ссылку адреса ячейки.

В нашем случае при записи в ячейку G4 значения 1 получим номер строки равный 4, а столба равным 3 на данном листе. То есть получаем адрес ячейки С4, в которой записана координата X_{11}. Аналогично находятся соответствующие адреса координат Y_{1j}. При наличии координат связных пунктов вычисляется расстояние.

После записи в ячейку R4 рассмотренного выражения с помощью автозаполнения заполняем другие ячейки данной строки. При этом изменяется только начальный адрес диапазона в функции ПОИСКПОЗ. Этим обеспечивается поиск связей 1 пункта со всеми возможными пунктами. Аналогичные выражения формируются для других строк матрицы транспортных расходов. Матрица транспортных расходов является классическим началом для решения транспортных задач, где имеются основные исходные данные задачи.

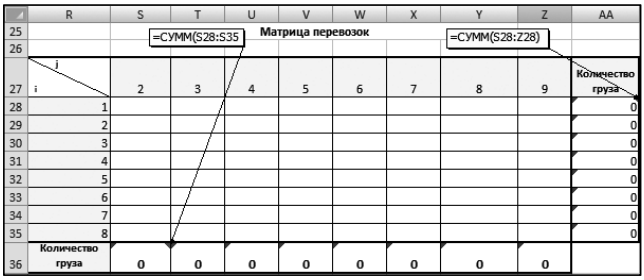
Теперь приступаем к этапу нахождения кратчайшего пути между пунктом отправления под номером 1 и пунктом назначения под номером 9.

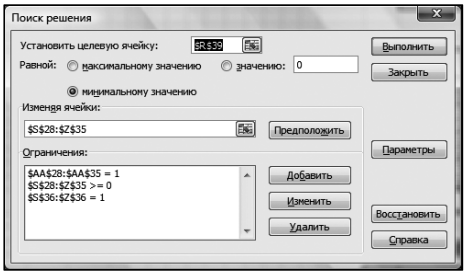
Требуется определить последовательность пунктов, через которые должен перемещаться груз, отправленный из исходного пункта в пункт назначения. Количество отправленного груза примем за 1, количество прибывшего груза в конечный пункт приравняем к 1. Для нахождения кратчайшего пути сформируем матрицу транспортных расходов (S16:Z23) на основе матрицы транспортных расходов. Если между пунктами нет пути, то такое расстояние приравнивается к числу, которое будет больше максимального расстояния из матрицы транспортных расходов. Для фор-

	R	S	T	U	V	W	X	Y	Z
15	i \ j	2	3	4	5	6	7	8	9
16	1	2,5	6,1	777,0	777,0	777,0	777,0	777,0	777,0
17	2	0,0	777,0	2,5	777,0	7	=ЕСЛИ(V7=0;777;V7)	7,0	777,0
18	3	777,0	0,0	777,0	777,0	777,0	3,5	777,0	777,0
19	4	2,5	777,0	0,0	5,0	777,0	777,0	777,0	777,0
20	5	777,0	777,0	5,0	0,0	5,6	777,0	777,0	777,0
21	6	777,0	10,4	777,0	777,0	0,0	777,0	777,0	777,0
22	7	777,0	3,5	777,0	777,0	777,0	0,0	3,9	777,0
23	8	777,0	777,0	777,0	777,0	777,0	3,9	0,0	5,0

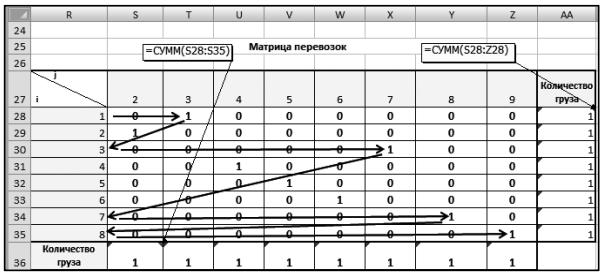
мирования матрицы заносим в ячейку S16 выражение =ЕСЛИ(S4=0;777;S4), которое затем с помощью автозаполнения распространяем на весь диапазон матрицы. Для чисел в матрице количество знаков после запятой устанавливаем равным единице. Максимальное расстояние устанавливаем 777 условным единицам. Обращаем внимание, что i=1,…,8,а j=2,…,9.

Теперь формируем матрицу перевозок (S28:Z35), где будут определяться целочисленные переменные x_{ij}, которые определят кратчайший путь. В диапазоне AA28:AA35 формируем суммы строк матрицы перевозок, а в диапазоне S36:Z36 – суммы столбцов матрицы (см. примечания). В ячейке R39 формируем выражение для целевой функции в виде суммы произведения матрицы транспортных расходов (S16:Z23) и матрицы перевозок (S28:Z35). Все готово для вызова процедуры *Поиск решения* (закладка *Данные*, группа *Анализ*). В окне *Поиск решения* устанавливаем адрес целевой функции, включаем флажок *минимальному значению*. Диапазон ячеек матрицы перевозок устанавливаем в *Изменяя ячейки*. Добавляем ограничения для ячеек матрицы перевозок, которые должны быть больше нуля. В диапазонах количества грузов устанавливаем равенство единице для обеспечения транспортировки груза через опорные пункты. Вызываем окно *Параметры поиска решения* через кнопку *Параметры* и устанавливаем флажок *Линейная модель*. Щелкаем на кнопке *ОК* и *Выполнить* в окне *Поиск решения*. Сохраняем найденное решение, которое наблюдаем в матрице перевозок и ячейке результата вычисления целевой функции. Длину кратчайшего пути получаем равной 18,47 условным единицам.





В матрице перевозок ячейки с единицами указывают на связные опорные пункты кратчайшего пути, который нужно рассматривать с 1-го опорного пункта. Кратчайший путь

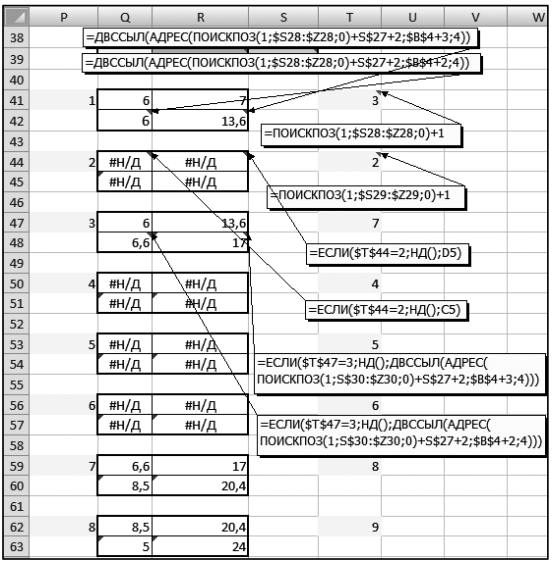


начинается с 1-го опорного пункта, который соединяется с 3-м. Затем продолжается до 7-го, далее до 8-го и заканчивается на конечном 9-м пункте.

Осталось изобразить на географической карте кратчайший путь по результатам решения.

Переходим к графическому изображению на географической карте кратчайшего пути. В качестве исходных данных используем матрицу перевозок и таблицу с координатами опорных пунктов.

Для построения кратчайшего пути создаем массив (Q41:R63) из парных ячеек с координатами связных пунктов. Парные ячейки позволят отобразить в виде линии отдельный участок пути. Количество пар должно соответствовать максимальному количеству участков дорожной сети. В столбце Р формируем номера опорных пунктов. В столбце Т определяем номер позиции или номер связного опорного пункта в строке матрицы перевозок, которая соответствуетствует определенному опроному пункту.



В ячейке Т41 вычисляем с помощью функции ПОИСКПОЗ(1;$S28:$Z28;0)+1 номер связного опорного пункта кратчайшего пути. Причем поиск значения 1 осуществляется во всей строке матрицы перевозок. Так как в ней может быть записана только одно значение равное 1 при нахождении краичайшего пути. Добавляем 1 в связи с тем, что первый столбец соответствует 2-му опроному пункту. Аналогично находим номера опорных пунктов в других строках (см. примечания). Эти номера будем использовать для исключения формирования координат участков, которые не принадлежат кратчайшему пути. Это связано с отсутствием ряда ограничений при применении процедуры *Поиск решения*, которые не позволили бы вносить корректировки в дорожную сеть. Но эти ограничения не влияют на нахождение кратчайшего пути.

В ячейки Q41 и R41 заносим координаты 1-го опорного пункта из таблицы координат (ячейки С4 и D4).

Координаты пары находим с помощью следующих выражений:

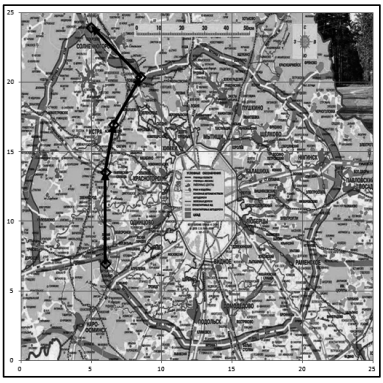
=ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;$S28:$Z28;0)+S$27+2;$B$4+2; 4))

=ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;$S28:$Z28;0)+S$27+2;$B$4+3;4)).

В этих выражениях по номеру позиции связного пункта выбираются его координаты из таблицы координат (см. примечания ячеек Q42 и R42).

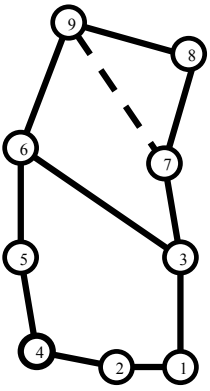
Второй пункт может не принадлежать кратчайшему пути. Поэтому для его исключения из оторбажения проводим проверку с испоьзованием выражений =ЕСЛИ($T$44=2;НД();C5) и =ЕСЛИ($T$44=2;НД();D5). При выполнении условия формируется признак «нет данных» с использованием функции НД(). В противном случае записываются коррдинаты 2-го пункта из таблицы координат. Для второй пары также проверяются условия и в случае выполнения формируем признак «нет данных».

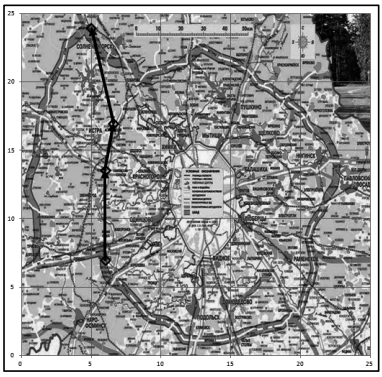
Иначе, как видно из выражений, записанных в ячейках Q45 и R45 =ЕСЛИ($T$44=2;НД();ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;S$29:$Z29;0) +S$27+2;$B$4+2;4))) и =ЕСЛИ($T$44=2;НД();ДВССЫЛ(АДРЕС (ПОИСКПОЗ(1;$S29:$Z29;0)+S$27+2;$B$4 +3;4))), определяются координаты связного опорного пункта, принадлежащего кратчайшему пути. Аналогично формируются координаты других участков кратчайшего пути.



В заключении выделяем диапазон ячеек Q41: R63 и строим диаграмму *Точечная с прямыми отрезками и маркерами*. Производим необходимое форматирование отрезков и маркеров для более четкого изображения кратчайшего пути.

Рассмотрим другой вариант дорожной сети и попробуем найти кратчайший путь. Соеденим 7-й опорный пункт и 9-й конечный пункт. Начальный и конечный пункты остаются те же. Наверняка кратчайший путь будет другим. Для этого добавим 1 в 7-ю строку 9-го столбца матрицы связности. При новом варианте моделирования нужно затем обнулить или удалить данные через клавишу Del из массива перевозок. Далее активизируем ячейку целевой функции и обращаемся к процедуре *Поик решения*. Длительность кратчайшего

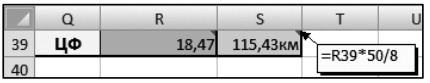




пути стала меньше почти на 2 единицы и равняется 16,73 условным единицам.

Для корректности осталось определить длительность пути в километрах.

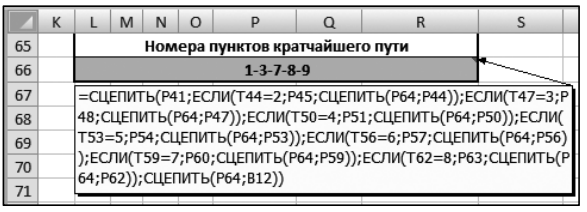
Для этого нужно узнать масштаб условной единицы расстояния. На карте изображена масштабная линейка. 5см на карте соответствует 50км. Обратимся к карте с изображением опорных пунктов и сформируем маркеры в начале и конце отрезка масштабной линейке, которые соответствуют 5 см или 50 км. Получим отрезок длиною 8 условных единиц. Следовательно, масштаб равен 50/8 км/усл.ед. При этом считаем, что нет искажений по координатам X и Y. В ячейке S39 вычисляем длину кратчайшего пути, которая с учетом масштаба равна 115,43 км. При этом для этой ячейки создали пользовательский формат с указанием размерности «км». Для этого вышли на *Формат ячеек*. В окне на закладке *Число* выбрали пункт *Все форматы*. В окне *Тип* выделили формат 0,00 и дописали “км”. Получаем длину пути с двумя знаками после запятой и подписью «км».



Целесообразно дополнительно сформировать ячейку для индикации кратчайшего пути с использованием номеров опорных пунктов. В качестве исходных данных будем использовать информацию массива P41:T63, которая формируется для графического построения кратчайшего пути. В индикационной ячейке сформируем цепочку из номеров опорных пунктов через тире. Начало цепочки начинается с 1, заканчиваться 9. Для формирования цепочки номеров будем испоьзовать функцию СЦЕПИТЬ. В ячейке L66 создаем следующие выражение для формирования номеров пунктов кратчайшего пути

=СЦЕПИТЬ(P41;ЕСЛИ(T44=2;P45;СЦЕПИТЬ(P64;P44));ЕСЛИ(T47=3;P48;СЦЕПИТЬ(P64;P47));ЕСЛИ(T50=4;P51;СЦЕПИТЬ(P64;P50));ЕСЛИ(T53=5;P54;СЦЕПИТЬ(P64;P53));ЕСЛИ(T56=6;P57;СЦЕПИТЬ(P64;P56));ЕСЛИ(T59=7;P60;СЦЕПИТЬ(P64;P59));ЕСЛИ(T62=8;P63;СЦЕПИТЬ(P64;P62));СЦЕПИТЬ(P64;B12)).

Первый аргумент функции соответствует номеру исходного пункта, который занесен в ячейку P41, последний аргумент функции СЦЕПИТЬ(P64;B12), где в ячейке P64 записали тире, а в ячейке B12 – число 9, которым обозначен конечный пунк пути. В последующих аргументах предварительно определяется принадлежность пункта кратчайшему пути. Если номер опорного пути не совпадает с номером связного пути, то в



сцепку включается тире и номер опорного пути. При выполнении условия в сцепку включается «пустая» ячейка. Таких аргументов равно 7-мь. В нашем случае в ячейке сформирована цепочка для первого нахождения кратчайшего пути:1-3-7-8-9.

2.3.2. Задача коммивояжера

Задан перечень населенных пунктов, которые соединены между собой дорожной сетью. Необходимо посетить все населенные пункты и вернуться в исходный пункт.

Нужно найти оптимальный путь между населенными пунктами и изобразить его на географической карте.

Перед нами классическая задача коммивояжера, который объезжает все населенные пункты. При этом каждый пункт посещается только один раз. Коммивояжер выезжает из исходного пункта и возвращается после объезда также в исходный пункт. Такая задача характерна для поездок по историческим местам или организации автобусного движения в городах.

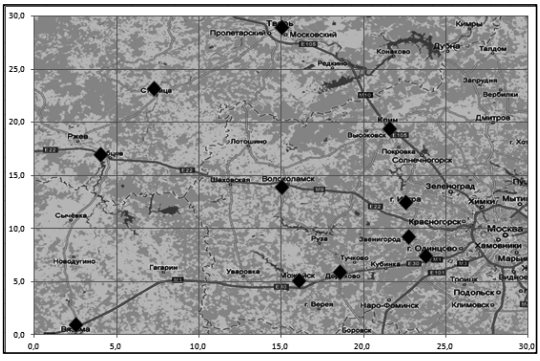
Методика решения задачи совпадает с методикой нахождения кратчайшего пути между населенными пунктами. Поэтому подробные комментарии будем давать при наличии отличий или особенностей решения.

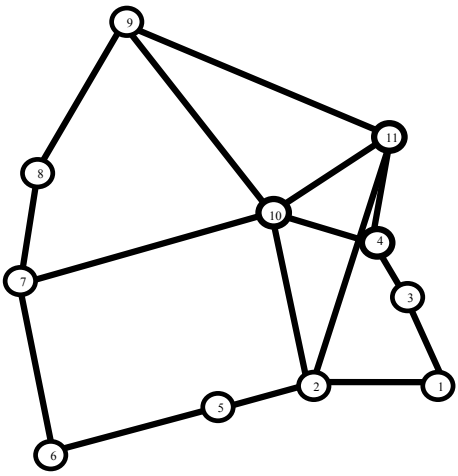
№ населенного пункта	Координаты X	Координаты Y
1	23,7	7,5
2	18,5	6,0
3	22,7	9,3
4	22,5	12,5
5	16,0	5,2
6	2,5	1,0
7	4,0	17,0
8	7,2	23,2
9	15,0	29,0
10	15,0	14,0
11	21,5	19,5

Матрица смежности

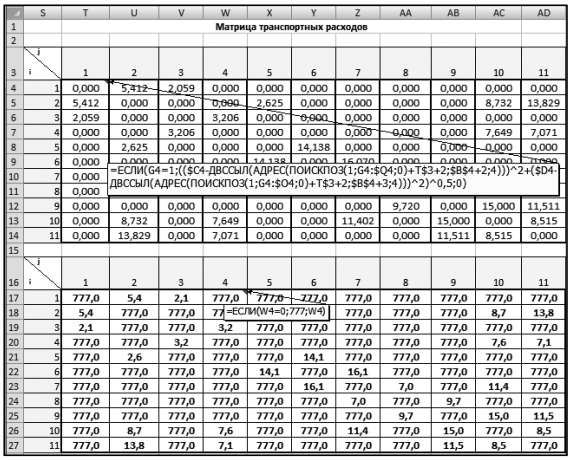
i \ j	1	2	3	4	5	6	7	8	9	10	11
1		1	1								
2	1				1					1	1
3	1			1							
4			1							1	1
5		1				1					
6					1		1				
7						1		1		1	
8							1		1		
9								1		1	1
10		1		1			1		1		1
11		1		1					1	1	

В начале формируем координаты населенных пунктов (C4:D14) и изображаем их на географической карте. Затем создаем матрицу смежности (G4:Q14) в соответствии с имеющейся дорожной сетью. Не будем учитывать опорные пункты, определяющие характерные изменения дорог, соединяющие населенные пункты, для сокращения





размерности матрицы. Матрица смежности соответствуют графу населенных пунктов, который определяет их последовательность и соединение между собой дорожной сетью. Создаем матрицу транспортных расходов (T4:AD14), где рассчитываются расстояния между населенными пунктами в условных единицах. На основании матрицы транспортных расходов формируем вспомогательную матрицу (T17:AD27), в которой при отсутствии дороги между пунктами заносим большое число. Это обеспечивает исключение из предполагаемого маршрута данные участки, соответствующие отсутствующим участкам дорог между населенными пунктами.



Матрица транспортных расходов

i \ j	1	2	3	4	5	6	7	8	9	10	11
1	0,000	5,412	2,059	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
2	5,412	0,000	0,000	0,000	2,625	0,000	0,000	0,000	0,000	8,732	13,829
3	2,059	0,000	0,000	3,206	0,000	0,000	0,000	0,000	0,000	0,000	0,000
4	0,000	0,000	3,206	0,000	0,000	0,000	0,000	0,000	0,000	7,649	7,071
5	0,000	2,625	0,000	0,000	0,000	14,138	0,000	0,000	0,000	0,000	0,000
6	0,000	0,000	0,000	0,000	14,138	0,000	16,070	0,000	0,000	0,000	0,000
7	0,000										
8	0,000										
9	0,000	0,000	0,000	0,000	0,000	0,000	0,000	9,720	0,000	15,000	11,511
10	0,000	8,732	0,000	7,649	0,000	0,000	11,402	0,000	15,000	0,000	8,515
11	0,000	13,829	0,000	7,071	0,000	0,000	0,000	0,000	11,511	8,515	0,000

=ЕСЛИ(G4=1;((($C4-ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;G4:$Q4;0)+T$3+2;$B$4+2;4)))^2+($D4-ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;G4:$O4;0)+T$3+2;$B$4+3;4)))^2)^0,5;0)

i \ j	1	2	3	4	5	6	7	8	9	10	11
1	777,0	5,4	2,1	777,0	777,0	777,0	777,0	777,0	777,0	777,0	777,0
2	5,4	777,0	777,0	77	=ЕСЛИ(W4=0;777;W4)		777,0	777,0	777,0	8,7	13,8
3	2,1	777,0	777,0	3,2	777,0	777,0	777,0	777,0	777,0	777,0	777,0
4	777,0	777,0	3,2	777,0	777,0	777,0	777,0	777,0	777,0	7,6	7,1
5	777,0	2,6	777,0	777,0	777,0	14,1	777,0	777,0	777,0	777,0	777,0
6	777,0	777,0	777,0	777,0	14,1	777,0	16,1	777,0	777,0	777,0	777,0
7	777,0	777,0	777,0	777,0	777,0	16,1	777,0	7,0	777,0	11,4	777,0
8	777,0	777,0	777,0	777,0	777,0	777,0	7,0	777,0	9,7	777,0	777,0
9	777,0	777,0	777,0	777,0	777,0	777,0	777,0	9,7	777,0	15,0	11,5
10	777,0	8,7	777,0	7,6	777,0	777,0	11,4	777,0	15,0	777,0	8,5
11	777,0	13,8	777,0	7,1	777,0	777,0	777,0	777,0	11,5	8,5	777,0

Размерность матриц определяется максимальным количеством рассматриваемых населенных пунктов.

Теперь все готово для нахождения маршрута по известным методикам. Математическая модель задачи выглядит следующим образом.

Целевая функция

$$F = \sum_{i=1,j=1}^{i=11,j=11} Rij * Xij \rightarrow min,$$

где Rij- расстояние между населенными пунктами,

Xij – переменные, которые определяют взаимосвязь населенных пунктов по пути коммивояжера,

i=1,…,9 – индекс исходящего населенного пункта,

j=1,…,9 – индекс входящего опорного пункта.

При решении задачи должны быть учтены следующие ограничения:

- Переменные Xij должны быть целыми, $Xij \geq 0$ и равны 1.
- $Xij = 0$ при $i = j$.
- Суммы $Xij + Xkl = 1$ при $k=j$ и $l=i$ для исключения петель в маршруте коммивояжера между населенными пунктами.
- Сумма

$$\sum_{j=1}^{j=1,\ldots,11} Xij = 1$$

для всех строк i=1,…,11.

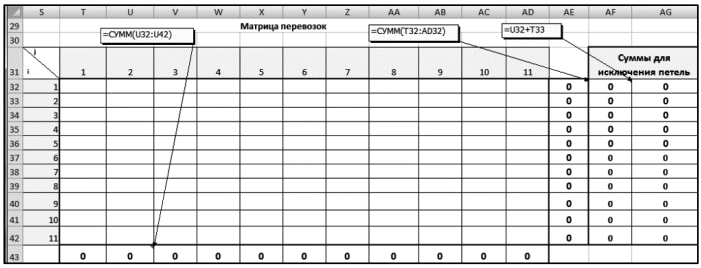
- Сумма

$$\sum_{i=1}^{i=1,\ldots,11} Xij = 1$$

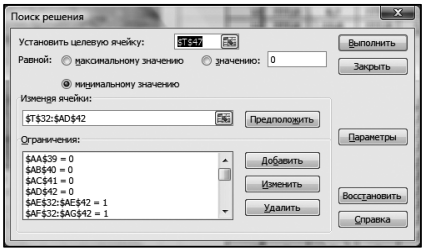
для всех столбцов j=1,…,11.

Последние два ограничения обеспечивают перемещение груза из одного населенного пункта только в другой населенный пункт.

В соответствии с методикой формируем матрицу перевозок (T32:AD42) для переменных Xij. Рядом формируем ячейки для ограничений по строкам и столбцам, а также для исключения петель в маршруте. Остается вызвать процедуру *Поиск решения*, установить адрес целевой ячейки, изменяемые адреса матрицы перевозок, ограничения и указать, что целевая функция должна стремиться к минимальному значению.



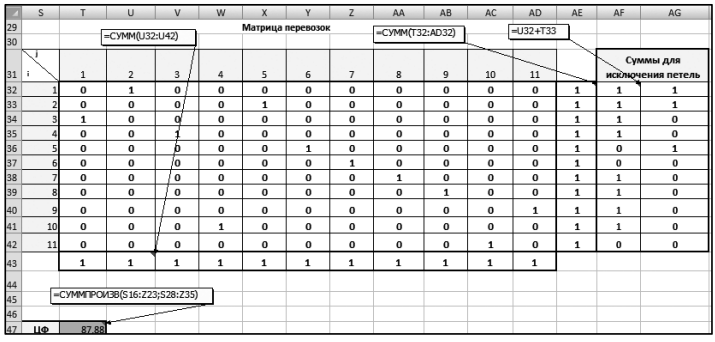
Следует перечислить все ограничения, которые были внесены. Это следующие ограничения:



- T32:AD42=целое, T32:AD42>=0.
- T32=0, U33=0, V34=0, W35=0, X36=0, Y37=0, Z38=0, AA39=0, AB=0, AC41=0, AD=42.
- AF32:AG42=1.
- T43:AD43=1, AE32:AE42=1.

Эти ограничения соответствуют ограничениям сформулированным в математической модели.

Через кнопку *Параметры* установить *Линейную модель*. Далее запустить процедуру *Поиск решения.* Получаем в целевой ячейке длину маршрута, которая равна 87,88 условным единицам.



Остается изобразить путь коммивояжера, который начинается с первого населенного пункта и на нем заканчивается. Так как все населенные пункты должны располагаться на пути, то задача формирования координат отрезков пути упрощается по сравнению с изображением кратчайшего пути.

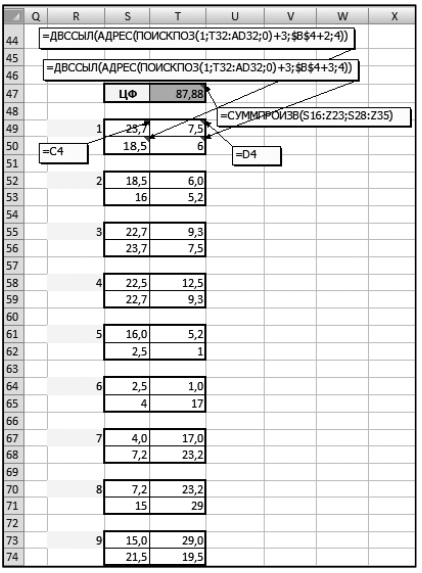
Формируем координаты отрезков пути для всех населенных пунктов. В первую пару ячеек координат просто заносим координаты соответствующих пунктов из таблицы координат (B4:D14). Координаты для второй пары ячеек отрезка формируем на основании поиска позиции с записью 1 в матрице перевозок. Запись 1 в соответствующей строке указывает номер населенного пункта, куда нужно отправляться из исходного пункта. Строки соответствуют очередному исходному населенному пункту. Остановимся на вычислении координат следующего пункта для первого исходного пункта.

При определении координаты X и Y используем следующие выражение:

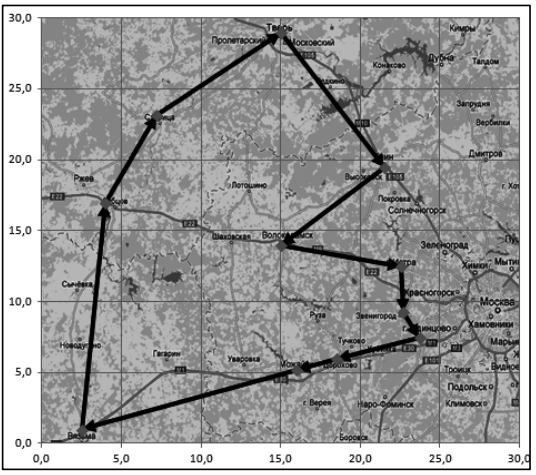
=ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;T32:AD32;0)+3;$B$4+2;4)),

=ДВССЫЛ(АДРЕС(ПОИСКПОЗ(1;T32:AD32;0)+3;$B$4+3;4)).

Аналогично находим координаты для других населенных пунктов.



Для изображения пути коммивояжера выделяем диапазон координат S49:T80 и обращаемся к построению диаграммы *Точечная с прямыми отрезками и маркерами.* Далее остается только провести форматирование графика для большей наглядности.



Сформируем текст маршрута объезда пунктов назначения. Создаим цепочку из названий пунктов и разделительного символа. Сначала сформируем вспомогательную таблицу

=ДВССЫЛ(АДРЕС(50+ПОИСКПОЗ($D51;$A$51:$A$61;0);3;4))

=ПОИСКПОЗ(1;$T32:$AD32;0)

=ДВССЫЛ(АДРЕС(50+$D51;2;4))

=A51

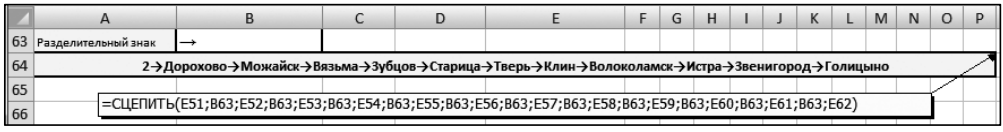
№ населенного пункта	Наименование населенного пункта	Номера связанных пунктов	Номера пунктов маршрута	Наименование пунктов маршрута
1	Голицыно	2	1	2
2	Дорохово	5	2	Дорохово
3	Звенигород	1	5	Можайск
4	Истра	3	6	Вязьма
5	Можайск	6	7	Зубцов
6	Вязьма	7	8	Старица
7	Зубцов	8	9	Тверь
8	Старица	9	11	Клин
9	Тверь	11	10	Волоколамск
10	Волоколамск	4	4	Истра
11	Клин	10	3	Звенигород
			1	Голицыно

номеров и названий населенных (A51:B61). Далее с помощью функции поиска позиции занесем номера населенных пунктов в соответствии с матрицей перевозок для каждой строки вспомогательной таблицы. Строки соответствуют конкретным населенным пунктам маршрута (см. примечание для ячейки C51). Это выражение для первого населенного пункта автозаполнением распространяем на строки других пунктов. Определили куда нужно перемещаться для каждого пункта в соответствии с маршрутом объезда. Осталось сформировать последовательность номеров пунктов согласно маршруту объезда. Запишем в ячейку D1 номер исходного пункта маршрута. Далее с помощью выражения

=ДВССЫЛ(АДРЕС(50+ПОИСКПОЗ($D51;$A$51:$A$61;0);3;4))

определяем номер пункта перемещения согласно маршрута коммивояжера. В функции АДРЕС() число 50 определяет начальный адрес вычисления строки. Число 3 обеспечивает выбор номера пункта, куда надо переместиться. Число 4 - параметр относительной ссылки. Функция ПОИСКПОЗ() осуществляет поиск позиции по содержимому ячейки D51, которая в данном случае соответствует 1 и является номером предыдущего пункта для нахождения номера последующего пункта маршрута коммивояжера. Используя автозаполнение рассмотренное выражение распространяем на нижестоящие строки (D53: D62). Здесь пришлось выйти за границу таблицы за счет того что в первую ячейку столбца занесли номер начального пункта маршрута. Последовательность номеров населенных пунктов в соответствии с маршрутом сформировали. Можно уже сформировать из номеров маршрут коммивояжера, но лучше это сделать с использованием названий населенных пунктов. В ячейку E51 заносим название исходного пункта с помощью выражения =ДВССЫЛ(АДРЕС(50+$D51;2;4)). По содержимому ячейки D51 формируем адрес названия населенного пункта во 2-м столбце вспомогательной таблицы. Аналогично формируем названия для других строк. Все готово для формирования текста маршрута коммивояжера.

Чтобы не использовать код символа разделения названий пунктов в ячейку B63 заносим символ стрелки. Используя функцию СЦЕПИТЬ формируем в ячейке A64 текст маршрута, куда последовательно заносим адреса ячеек с названиями пунктов и адрес ячейки с разделительным



знаком. Предварительно делаем объединение ячеек в строке, чтобы был виден весь текст маршрута.

2.3.3. Перечень покупаемой продукции при условии минимизации суммы оплаты

Имеются прейскуранты продукции двух производителей, в которых указаны цены и количество продукции. Необходимо составить перечень покупаемой продукции при условии минимизации общей суммы оплаты. Количество покупаемой продукции одного типа должна равняться 10.

Для первичного анализа прейскуранта производим условное форматирование ячеек отсутствующей продукции. С помощью групповой клавиши Ctrl выделяем диапазоны D6:D19 и F6:F19, обращаемся к *Условному форматированию* на закладке *Главная*. В правилах выделения ячеек выбираем равенство нулю и устанавливаем нужный формат. Задаем количество продукции, которую нужно приобрести. В нашем случае устанавливаем количество равное 10 (J6:J19).

	B	C	D	E	F	G	H	I	J
2	Прейскурант стоимости продукции							Таблица количества необходимой продукции	
3			Название фирмы						
4	№		Фирма 1		Фирма 2			№	Количество,
5	п/п	Наименование продукции	Количество, шт	Стоимость, руб	Количество, шт	Стоимость, руб		п/п	шт
6	1	Продукция 1	100	100	30	120		1	10
7	2	Продукция 2	90	200	0	190		2	10
8	3	Продукция 3	80	300	70	310		3	10
9	4	Продукция 4	70	10	90	8		4	10
10	5	Продукция 5	60	25	0	30		5	10
11	6	Продукция 6	50	35	130	40		6	10
12	7	Продукция 7	40	75	150	80		7	10
13	8	Продукция 8	30	100	200	100		8	10
14	9	Продукция 9	20	150	0	200		9	10
15	10	Продукция 10	30	300	400	100		10	10
16	11	Продукция 11	0	60	500	70		11	10
17	12	Продукция 12	50	80	600	90		12	10
18	13	Продукция 13	60	100	700	90		13	10
19	14	Продукция 14	70	250	800	250		14	10

Сформулируем математическую модель решения задачи. Целевая функция

$$F = \sum_{i=1,j=1}^{j=14} Cij * Xij + \sum_{i=2,j=1}^{j=14} Cij * Xij \to min,$$

где Cij- стоимость продукции,

Xij – переменные, которые определяют оптимальное количество покупаемой продукции,

i=1,2 – индекс фирмы,

j=1,…,14 – индекс продукции.

При решении задачи должны быть учтены следующие ограничения:

- Переменные Xij должны быть целыми и $Xij \geq 0$.
- Сумма $X1j + X2j = 10$ по каждому виду продукции.

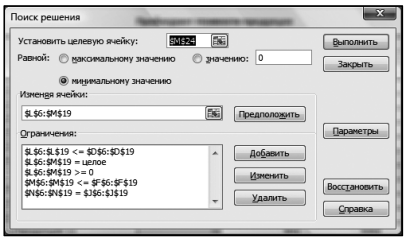
Формируем диапазоны перемен-

	I	J	K	L	M	N
2	Таблица количества необходимой продукции			Таблица переменных X1i и X2i		=L6+M6
3				Название фирмы		
4	№					Сумма
5	п/п	Количество, шт		Фирма 1	Фирма 2	переменных X1i и X2i
6	1	10				0
7	2	10				0
8	3	10				0
9	4	10				0
10	5	10				0
11	6	10				0
12	7	10				0
13	8	10				0
14	9	10				0
15	10	10				0
16	11	10				0
17	12	10				0
18	13	10				0
19	14	10				0
20						
21	=СУММПРОИЗВ(E6:E19;L6:L19)+СУММПРОИЗВ(G6:G19;M6:M19)					
22						
23						
24				ЦФ		0

ных L6:L19 M6:M19, которые определят количество покупаемой продукции в каждой фирме. Так как количество покупаемой продукции не может превышать заданную (см. J6:J19), то создаем диапазон сумм переменных для каждого вида продукции N6:N19, который используем при формировании ограничений.

В ячейке M24 создаем выражение для целевой функции =СУММПРОИЗВ(E6:E19;L6:L19)+СУММПРОИЗВ(G6:G19; M6:M19), которая является суммой произведения цены на количество покупаемой продукции.

Обращаемся к процедуре *Поиск решения* на закладке *Данные*. В окне

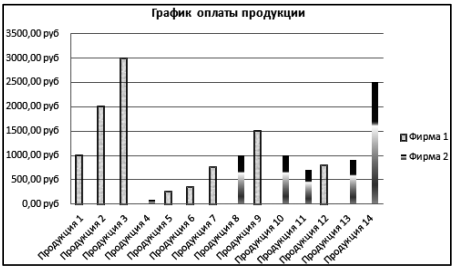


устанавливаем адрес целевой ячейки, включаем *Равной минимальному значению.* Задаем диапазон N6:N19 в окне *Изменяя ячейки.* Добавляем ограничения, в *Параметры* включаем флажок *Линейная модель.* Запускаем процедуру *Поиск решения* и сохраняем результат решения.

Создаем таблицу оплаты, где формируем произведения стоимости на количество купленной продукции для каждой фирмы. В таблице оплаты проводим условное форматирования для выделения ячеек, где отсутствует продукция. Так же в таблице создаем пользовательский формат с указанием размерности оплаты в рублях (см. предыдущие задачи). Для облегчения анализа результатов решения создаем гистограмму *График оплаты продукции*. При выборе вариантов приобретения продукции целесообразно иметь дополнительную информацию о сроках, дальности доставки и т.п.

	L	M	N	O	P	Q
2	Таблица переменных X1i и X2i		=L6+M6	=E6*L6	Таблица оплаты	
3	Название фирмы		Сумма переменных X1i и X2i		Название фирмы	
4						
5	Фирма 1	Фирма 2			Фирма 1	Фирма 2
6	10	0	10		1000,00 руб	0,00 руб
7	10	0	10		2000,00 руб	0,00 руб
8	10	0	10		3000,00 руб	0,00 руб
9	0	10	10		0,00 руб	80,00 руб
10	10	0	10		250,00 руб	0,00 руб
11	10	0	10		350,00 руб	0,00 руб
12	10	0	10		750,00 руб	0,00 руб
13	0	10	10		0,00 руб	1000,00 руб
14	10	0	10		1500,00 руб	0,00 руб
15	0	10	10		0,00 руб	1000,00 руб
16	0	10	10		0,00 руб	700,00 руб
17	10	0	10		800,00 руб	0,00 руб
18	0	10	10		0,00 руб	900,00 руб
19	0	10	10		0,00 руб	2500,00 руб
20						
21		=СУММПРОИЗВ(E6:E19;L6:L19)+СУММПРОИЗВ(G6:G19;M6:M19)				
22						
23						
24	ЦФ	15830				

В таблице приведены данные о стоимости, сроках и дальности доставки, которые в определенных условиях необходимо учитывать, а также сумма оплаты для каждой фирмы.



Ниже приведена таблица, где для первой фирмы параметры приравнены к

100%, а для второй фирмы произведены относительные данные. Такие сравнения иногда легче воспринимаются при выборе альтернатив, чем абсолютные величины. Причем лучше делать с помощью графиков.

	C	D	E	F	G	H
20	Стоимость доставки, руб	1000,00		2000,00		
21	Сроки доставки , час	48,00		24,00		
22	Дальность доставки, км	50,00		70,00		
23	К оплате продукции, руб	9650,00		6180,00		
24				=СУММПРОИЗВ(G6:G19;M6:M19)		
25	Стоимость доставки	100 %		200 %		
26	Сроки доставки	100 %		50 %		
27	Дальность доставки	100 %		140 %		
28	К оплате продукции	100 %		64 %		
29				=F21/D21*D26		
30	**Таблица анализа эффективности оптимального плана закупок**					
31	=СУММПРОИЗВ(E6:E19;J6:J19				=D33-$M$24	
32	Название фирмы	Стоимость покупки только в одной фирме	Экономия	Номера отсутствующей продукции в фирме		
33	Фирма 1	17850,00 руб	2020,00 руб	11		
34	Фирма 2	16780,00 руб	950,00 руб	2 5 9		
35	=СЦЕПИТЬ(ЕСЛИ(D6=0;СЦЕПИТЬ(B6;СИМВОЛ(32)););ЕСЛИ(D7=0;СЦЕПИТЬ(B7;СИМВОЛ(32)););ЕСЛИ(D8					
36	=0;СЦЕПИТЬ(B8;СИМВОЛ(32)););ЕСЛИ(D9=0;СЦЕПИТЬ(B9;СИМВОЛ(32)););ЕСЛИ(D10=0;СЦЕПИТЬ(B10;С					
37	ИМВОЛ(32)););ЕСЛИ(D11=0;СЦЕПИТЬ(B11;СИМВОЛ(32)););ЕСЛИ(D12=0;СЦЕПИТЬ(B12;СИМВОЛ(32)););Е					
38	СЛИ(D13=0;СЦЕПИТЬ(B13;СИМВОЛ(32)););ЕСЛИ(D14=0;СЦЕПИТЬ(B14;СИМВОЛ(32)););ЕСЛИ(D15=0;СЦЕ					
39	ПИТЬ(B15;СИМВОЛ(32)););ЕСЛИ(D16=0;СЦЕПИТЬ(B16;СИМВОЛ(32)););ЕСЛИ(D17=0;СЦЕПИТЬ(B17;СИМВ					
40	ОЛ(32)););ЕСЛИ(D18=0;СЦЕПИТЬ(B18;СИМВОЛ(32)););ЕСЛИ(D19=0;СЦЕПИТЬ(B19);))					

Оценку эффективности можно сделать на основе данных в *Таблице анализа эффективно*сти оптимального *плана закупок* продукции. Стоимость покупок в одной фирме оцениваем с помощью суммы произведений стоимости продукции и необходимого количества продукции. Экономию рассчитываем в сравнении с суммой оптимальной закупки. Нужно это делать с учетом отсутствующей продукции на фирме.

Номера отсутствующей продукции формируем с помощью функции СЦЕПИТЬ. Где сначала проверяем отсутствие продукции по равенству нулю соответствующей ячейки и при выполнении условия сцепляем номер продукции с символом пропуска, который имеет код 32

СЦПИТЬ(ЕСЛИ(D6=0;СЦЕ-
ПИТЬ(B6;СИМВОЛ(32)););.(здесь приведен пример для 1-го продукта 1-й фирмы).

Количество таких записей равно числу продукции в фирме. Применение символа пропуска позволяет исключить момент связанный с определением последнего отсутствую-

	L	M	N	O	P	Q
2	**Таблица переменных X1i и X2i**		=L6+M6	=E6*L6	**Таблица оплаты**	
3	Название фирмы		Сумма переменных X1i и X2i		Название фирмы	
4						
5	Фирма 1	Фирма 2			Фирма 1	Фирма 2
6	5	5	10		500,00 руб	600,00 руб
7	10	0	10		2000,00 руб	0,00 руб
8	10	0	10		3000,00 руб	0,00 руб
9	0	10	10		0,00 руб	80,00 руб
10	10	0	10		250,00 руб	0,00 руб
11	10	0	10		350,00 руб	0,00 руб
12	10	0	10		750,00 руб	0,00 руб
13	10	0	10		1000,00 руб	0,00 руб
14	10	0	10		1500,00 руб	0,00 руб
15	0	10	10		0,00 руб	1000,00 руб
16	0	10	10		0,00 руб	700,00 руб
17	10	0	10		800,00 руб	0,00 руб
18	0	10	10		0,00 руб	900,00 руб
19	0	10	10		0,00 руб	2500,00 руб
20						
21			=СУММПРОИЗВ(E6:E19;L6:L19)+СУММПРОИЗВ(G6:G19;M6:M19)			
22						
23						
24	ЦФ	15930				

щего товара и удалением других разделяющих символов номеров продукции. Символы в функции СИМВОЛ выбираются из таблицы знаков компьютера по числам от 1 до 255.

Из начальных условий видим, что количество необходимой продукции значительно меньше, чем имеется для продажи. Теперь проверим нахождение оптимального плана покупки при наличии продукции меньше, чем требуется. Зададим количество *Продукции1*на фирмах равное 5-ти. В этом случае получим, что необходимо заплатить 15930 рублей и *Продукция1* будет куплена у каждой фирмы по 5 штук. По сравнению с первым решением нужно заплатить на 100 рублей больше. Как видим можно изменить исходные данные и получать оптимальные планы закупок при различных обстоятельствах.

2.4. Защита информации

2.4.1. Зашифрованное письмо

Сейчас много внимания уделяют вопросам безопасности информации. Создадим простейшую модель шифрования текста.

Следует различать методы кодирования и шифрования текста. В первом случае буквы алфавита заменяются произвольными символами (числами). При шифровании обязательно производятся какие-то действия по заданному алгоритму замены исходных букв текста символами.

Произведем кодирование и шифрование слова «кодирование». Для простоты каждая буква слова записаны в отдельную ячейку (C3:M3).

Каждую букву закодируем двузначным числом. Создадим массив букв алфавита в виде таблицы из 5-ти строк и 7-ми столбцов. Пустые ячейки заполним нулями. Можно записать в них нужные символы.

	P	Q	R	S	T	U	V	W	X
12	а	б	в	г	д	е	ё	1	
13	ж	з	и	к	л	м	н	2	Строка
14	о	п	р	с	т	у	ф	3	
15	х	ц	ч	ш	щ	ъ	ы	4	
16	ь	э	ю	я		0	0	5	
17	1	2	3	4	5	6	7		
18	Столбец								

Первая цифра кода буквы соответствует строке массива, а вторая – столбцу массива. Воспользуемся функцией ВПР() для определения первой цифры (см. примечание). В качестве первого параметра используем букву слова. Адрес массива делаем абсолютным для дальнейшего использования в автозаполнении для других букв слова. В 8-м столбце массива формируем числовые значения строк. Третий параметр определяет номер столбца массива, из которого функция выбирает соответствующие значение для заданного значения буквы слова сообщения. Функция ВПР() выполняется без сбоев при формировании массива букв по возрастанию по горизонтали и вертикали.

Используем функцию ГПР() для определения второй цифры буквы. Для бесперебойной работы функции ГПР() организуем «динамический» массив букв алфавита с использованием функции СМЕЩ(). Первая строка «динамического» массива определяется строкой исходного массива, в которой стоит заданная буква. Число строк также зависит от первой цифры кода буквы. Смещение по столбцам отсутствует, а число столбцов постоянно и равно 7-ми (см. примечание). Последний параметр ГПР() вычисляется с учетом строки массива, в которой находится заданная буква.

Все цифры букв сообщения определены. Можно формировать код, который однозначно соответствует конкретной букве. Если у нас двузначный код, то первую цифру берем с весом 10, а вторую – с весом 1 и суммируем (см. примечание). Можно усложнить формирование кода. При этом фактически будет происходить шифрование текста. Например, код первой буквы формируем по вышеизложенному правилу, а код четной буквы формируем аналогично, но веса меняем местами и т.д. Можно также сформировать сообщение из 3-х чисел (C9:C11). Вариантов достаточно много.

Пусть передано зашифрованное сообщение (C4:M4). На приемном стороне расшифруем его.

Первую цифру кода вычисляем, как целую часть числа, поделенное на 10. Вторую цифру кода определяем, как остаток при вычитании от числа кода первой цифры умноженной на 10. Далее используем функцию ИНДЕКС(), где первая цифра определяет строку, а вторая столбец в массиве букв алфавита (P12:V16). Текст сообщения расшифрован.

Целесообразно дополнительно для шифрования использовать последовательность случайных чисел, которые распределены по равномерному закону в диапазоне от 0 до 10. Формировать последовательность можно с помощью функции СЛУЧМЕЖДУ(). Однако в этом случае нужно заранее определить порядок кода (число цифр в коде букв).

2.4.2. Стеганографическое письмо

Стеганография является наукой о скрытой передачи данных. Для передачи сообщений используются «контейнеры». В качестве контейнеров используются графические и звуковые файлы, а также тексты и т.п. В основном для передачи используются последние разряды кодов графических и звуковых файлов.

Используем в качестве контейнера табулированные значения функции. Создадим таблицу (C3:AD4) с табулированными значениями функции $\sin x$. Функцию можно взять любую. Длина таблицы определяется возможным количеством передаваемых символов в сообщении. В нашей модели с помощью контейнера передадим слово «стеганография».

	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	ΔX	0,2		K	100000															
2																				
3	X	0	0,2	0,4	0,6	0,8	1	1,2	1,4	1,6	1,8	2	2,2	2,4	2,6	2,8	3	3,2	3,4	3,6
4	SIN(X)	0,000000	0,198669	0,389418	0,564642	0,717356	0,841471	0,932039	0,985450	0,999574	0,973848	0,909297	0,808496	0,675463	0,515501	0,334988	0,141120	#######	#######	#######

После создания таблицы заносим в каждую ячейку по букве из заданного слова (C9:O9). Для удобства первая буква располагается под первым значением табулированной функции. Затем определяем коды букв с использованием функции КОДСИМВ().

	B	C	D	E	F	G	H	I	J	K	L	M	N	O
6														
7						*Передающая сторона*								
8														
9	**Сообщение**	**С**	**т**	**е**	**г**	**а**	**н**	**о**	**г**	**р**	**а**	**ф**	**и**	**я**
10														
11	**Код символа**	209	242	229	227	224	237	238	227	240	224	244	232	255
12			=КОДСИМВ(C9)											
13	**Нормированный код**	0,002090	0,002420	0,002290	0,002270	0,002240	0,002370	0,002380	0,002270	0,002400	0,002240	0,002440	0,002320	0,002550
14														
15	**Письмо отправили**	0,002090	0,201089	0,391708	0,566912	0,719596	0,843841	0,934419	0,987720	1,001974	0,976088	0,911737	0,810816	0,678013

=C4+C13

=F11/$F$1

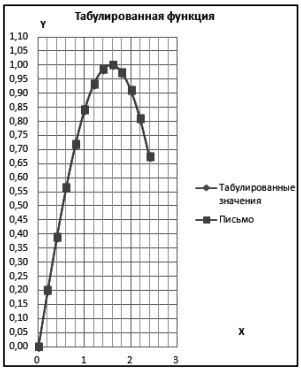
Нормируем коды с помощью коэффициента К, значение которого записано в ячейке F1. Нормировка обеспечит использование самого младшего разряда значений табулированной функции. Далее создаем таблицу для скрытой передачи сообщения путем сложения значений табулированной функции с нормированными кодами букв слова «стеганография» (C15:O15). Эта таблица передается получателю скрытого сообщения.

После приема сообщения нужно его прочитать. На приемном конце должна быть исходная табулированная функция или данные для ее генерации. В нашем случае вид функции, начальное значение аргумента и шаг изменения аргумента.

Прочтение сообщения осуществляется в 3-и этапа. На первом этапе вычисляем значения нормированных значений кодов. Из принятых значений табулированной функции вычитаем ее истинные значения (C22:O22).

	B	C	D	E	F	G	H	I	J	K	L	M	N	O
19														
20		=C15-C4				*Приемная сторона*								
21														
22	**1 этап расшифровки**	0,002090	0,002420	0,002290	0,002270	0,002240	0,002370	0,002380	0,002270	0,002400	0,002240	0,002440	0,002320	0,002550
23			=C22*$F$1				=СИМВОЛ(F24)							
24	**2 этап расшифровки**	209	242	229	227	224	237	238	227	240	224	244	232	255
25														
26	**Письмо прочли**	**С**	**т**	**е**	**г**	**а**	**н**	**о**	**г**	**р**	**а**	**ф**	**и**	**я**

Затем производим обратную нормировку и получаем значения кодов букв сообщения (C24:O24). И, применив функцию СИМВОЛ(), прочитываем принятое сообщение (C26:O26).



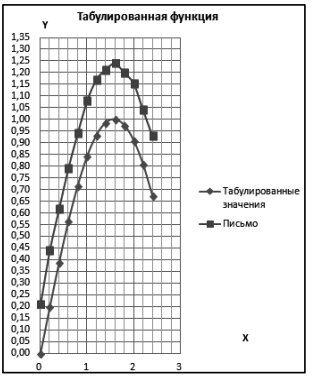
Построим графики истинной и с сообщением табулированной функции. Выделим диапазоны C3:O4 и C15:O15 для построения диаграммы *Точечная с гладкими кривыми и маркерами.* На графике две кривые совпадают. То есть факт передачи сообщения практически нельзя обнаружить.

	B	C	D	E	F	G	H
1	ΔX	0,2		K	1000		
2							
3	X	0	0,2	0,4	0,6	0,8	1
4	SIN(X)	0,000000	0,198669	0,389418	0,564642	0,717356	0,841471
5							
6							
7							*Передающая ст*
8							
9	Сообщение	С	т	е	г	а	н
10							
11	Код символа	209	242	229	227	224	237
12			=КОДСИМВ(C9)				
13	Нормированный код	0,209000	0,242000	0,229000	0,227000	0,224000	0,237000
14			=C4+C13				
15	Письмо отправили	0,209000	0,440669	0,618418	0,791642	0,941356	1,078471

=F11/$F$1

Изменим масштабный коэффициент так, чтобы коды символов совпадали со старшим разрядом после запятой табулированных значений (К=1000).

В этом случае наблюдаем несовпадение графиков табулированной функции и табулированной функции с сообщением. То есть по этим несовпадениям можно обнаружить факт передачи сообщения.



Изменяя значение масштабного коэффициента можно оценить возможности по скрытию сообщения. Кроме значений табулированной функции в качестве контейнера можно использовать векторные изображения, где в координаты опорных точек фигур можно добавить нормированные коды символов коротких сообщений.

2.4.3. Электронная подпись

Создадим модель электронно-цифровой подписи. Два абонента пересылают друг другу свои зашифрованные фамилии.

Для шифрации фамилий используем алгоритм RSA с открытым ключом.

Разработка модели состоит из следующих основных этапов:

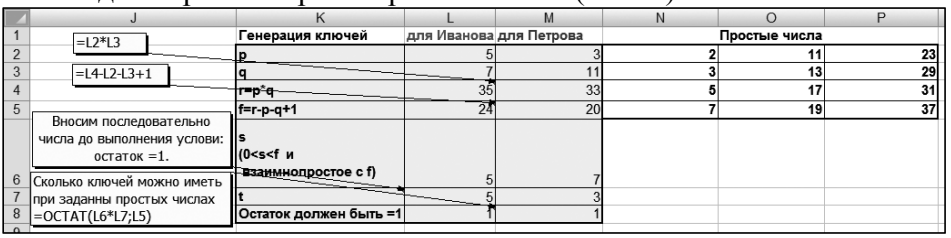
- Генерации ключей абонентам.
- Шифрования и отправки ЭЦП.
- Получения ЭЦП и ее дешифровки.

Алгоритм генерации ключей заключается в выборе 2-х простых чисел p и q, вычислении произведения простых чисел r=p*q, нахождении функции Эйлера f=(p-1)*(q-1). Далее нужно выбрать взаимнопростое с функцией Эйлера число s, которое удовлетворяет условию 0<s<f. В заключении определяется число t. Это число находится из выражения t*s=1*mod(f).

Числа r и s являются открытыми ключами, а число t – закрытым ключом.

Открытые ключи публикуются и используются для шифрования сообщений владельцу ключей. Закрытый ключ хранится у владельца и используется для дешифрования поступивших ему сообщений.

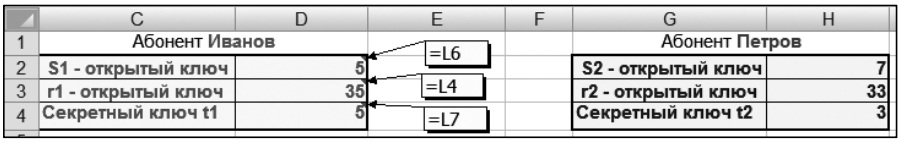
Сгенерируем ключи для абонентов под фамилиями Иванов и Петров. Запишем для справки первые простые числа (N2:P5).



		5	3	2	11	23
		7	11	3	13	29
		35	33	5	17	31
		24	20	7	19	37
		5	7			
		5	3			
		1	1			

Зададим для каждого абонента по паре простых чисел (L2:M3) и произведем вычисления r, f. Выберем значения s для каждого абонента. Затем последовательно, задавая значения t для каждого абонента, будем добиваться равенства 1 в ячейках L8 и M8. При этом t должно быть меньше f.

После вычисления ключей сформируем вспомогательные ячейки ключей (D2:D4 и H2:H4) для каждого абонента.



	Абонент Иванов				Абонент Петров	
	S1 - открытый ключ	5			S2 - открытый ключ	7
	r1 - открытый ключ	35			r2 - открытый ключ	33
	Секретный ключ t1	5			Секретный ключ t2	3

Перейдем к этапу шифрования сообщений. Шифровка сообщений осуществляется с помощью открытых ключей абонента, которому они будут отправлены.

Создадим массив алфавита (B7:B39) и сформируем цифровые коды k для букв (C7:C39).

Необходимо отметить, что для буквы «а» выбрана число «2». Единица не используется при кодировании в силу того, что единица в любой степени равняется единице.

В столбец I7:I39 также занесли алфавит для удобства.

Предварительно определим шифр R для всех символов по следующему алгоритму:

$$R = (k^s) mod(r).$$

Реально все это сводится к вычислению для каждого символа алфавита числа k^s.

А далее рассчитываем модуль числа k^s от ключа r с использованием функции ЦЕЛОЕ() по выражению $R = k^s - \text{ЦЕЛОЕ}(k^s/r) * r$ (см. примечания).

	B	C	D	E	F	G	H	I
6	Символы	Число	Число в степени s1	Остаток от деления на r1 (шифр символа)		Число в степени s2	Остаток от деления на r2 (шифр символа)	Символы
7	а	2	32	32		128	29	а
8	б	3	243	33		2187	9	б
9	в	4	1024	9		16384	16	в
10	г	5	3125	10		78125	14	г
11	д	6	7776	6		279936	30	д
12	е	7	16807	7		823543	28	е
13	ё	8	32768	8		2097152	2	ё
14	ж	9	59049	4		4782969	15	ж
15	з	10	100000	5		10000000	10	з
16	и	11	161051	16		19487171	11	и
17	й	12	248832	17		35831808	12	й
18	к	13	371293	13		62748517	7	к
19	л	14	537824	14		105413504	20	л
20	м	15	759375	15		170859375	27	м
21	н	16	1048576	11		268435456	25	н
22	о	17	1419857	12		410338673	8	о
23	п	18	1889568	23		612220032	6	п
24	р	19	2476099	24		893871739	13	р
25	с	20	3200000	20		1280000000	26	с
26	т	21	4084101	21		1801088541	21	т
27	у	22	5153632	22		2494357888	22	у
28	ф	23	6436343	18		3404825447	23	ф
29	х	24	7962624	19		4586471424	18	х
30	ц	25	9765625	30		6103515625	31	ц
31	ч	26	11881376	31		8031810176	5	ч
32	ш	27	14348907	27		10460353203	3	ш
33	щ	28	17210368	28		13492928512	19	щ
34	ь	29	20511149	29		17249876309	17	ь
35	ъ	30	24300000	25		21870000000	24	ъ
36	э	31	28629151	26		27512614111	4	э
37	ю	32	33554432	2		34359738368	32	ю
38	я	33	39135393	3		42618442977	0	я
39	пробел	34	45435424	34		52523350144	1	пробел

=C7^$D$2

=D7-ЦЕЛОЕ((D7/$D$3))*$D$3

=C7^$H$2

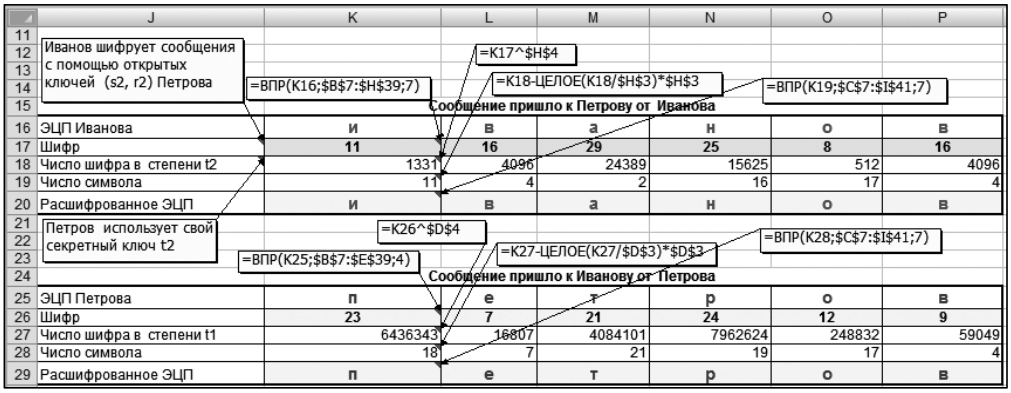
=G7-ЦЕЛОЕ((G7/$H$3))*$H$3

С использованием открытых ключей (*s,r*) произвели шифрование алфавита для каждого абонента (E7:E39 и H7:H39). Это упростит формирование сообщений.

Необходимо остановиться на следующем моменте. У нас в алфавите 33 символа. Если открытый ключ $r = 33$, то R для отдельных символов будет ранен 0 и 1. Эти числа не обеспечат дешифрование. Нужно значение открытого ключа было больше количества символов в алфавите. Если значение ключа меньше, чем количество символов, то ряд символов будут иметь одинаковые числовые шифры. В этом можно убедиться при моделировании.

Создадим элемент модели шифрования, передачи и дешифрования.

Каждый абонент посылает другому свою фамилию в качестве ЭЦП (J16:P16 и J25:P25). В ячейки записываем буквы фамилий. В нижних строках с помощью функции ВПР() выбираем числа шифра соответствующих букв (см. примечания). При этом выбираются шифры, сформированные открытыми ключами получателя сообщений. Эти зашифрованные фамилии дешифрируются получателями с использованием своих открытых и секретных ключей.



	J	K	L	M	N	O	P
16	ЭЦП Иванова	и	в	а	н	о	в
17	Шифр	11	16	29	25	8	16
18	Число шифра в степени t2	1331	4096	24389	15625	512	4096
19	Число символа	11	4	2	16	17	4
20	Расшифрованное ЭЦП	и	в	а	н	о	в
25	ЭЦП Петрова	п	е	т	р	о	в
26	Шифр	23	7	21	24	12	9
27	Число шифра в степени t1	6436343	16807	4084101	7962624	248832	59049
28	Число символа	18	7	21	19	17	4
29	Расшифрованное ЭЦП	п	е	т	р	о	в

Сначала шифр буквы возводится в степень, которая соответствует секретному ключу R^t. Далее нужно определить числовой код буквы k с использованием открытого ключа r по формуле

$$k = (R^t) mod(r)$$ (см. примечания).

Окончательную расшифровку ЭЦП (J20:P20 и J29:P29) получаем после выбора символов алфавита по их кодам с помощью функции ВПР() из массива алфавита.

Модель ЭЦП можно использовать для исследований по формированию ключей и шифрованию символов алфавита.

Следует отметить, что при попадании открытого ключа в посторонние руки им могут воспользоваться для передачи ложной ЭЦП. Для исключения таких случаев разработаны другие алгоритмы ЭЦП.

2.5. Другие модели

2.5.1. Простейший блокнот

Разработаем простейшую модель для записи сообщений с помощью стилистического изображения букв и знаков препинания. Индицировать сообщение будем в области построения диаграмм.

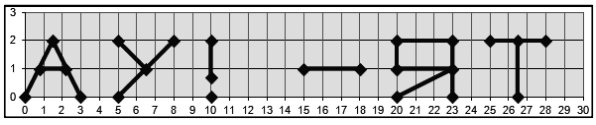
Для простоты создадим алфавит для фразы «Ау! Я тут!». Как видно нужно создать образы 4-х букв, символ восклицательного знака, обеспечить формирование пропуска между словами, а также переход на следующую строку.

Высоту и ширину букв возьмем одинаковую. Для каждой буквы определим минимальное количество элементов, которое обеспечивает их однозначное распознавание. Создадим массив букв и в нем зададим координаты отрезков элементов в столбцах B и C. Каждый отрезок элемента буквы занимает 2-е строчки.

	A	B	C	D	E	F	G
4	Буква	Координаты элементов букв		Смещение для просмотра алфавита		Координаты алфавита	
5		X	Y	ΔX	ΔY	X+ΔX	Y+ΔY
6	А	0	0	0	0	0	0
7	9	1,5	2	0	0	1,5	2
8							
9		3	0	0	0	3	0
10		1,5	2	0	0	1,5	2
11							
12		0,8	1	0	0	0,8	1
13		2,2	1	0	0	2,2	1
14							
15	У	0	0	5	0	5	0
16	6	3	2	5	0	8	2
17							
18		0	2	5	0	5	2
19		1,5	1	5	0	6,5	1
20							
21	!	0	0	10	0	10	0
22	5						
23		0	0,7	10	0	10	0,7
24		0	2	10	0	10	2
25							
26	-	0	1	15	0	15	1
27	3	3	1	15	0	18	1
28							
29	Я	0	0	20		20	0
30	15	3	1	20		23	1
31							
32		3	0	20		23	0
33		3	2	20		23	2
34							

В первой строчке элемента буквы в столбце A вводим обозначение буквы. Координаты символов алфавита привязываем к началу координат. Чтобы обеспечить просмотр создаваемого алфавита создаем ячейки для смещения изображений и массив координат алфавита для просмотра. Буква *А* у нас первая и поэтому смещение равно нулю. Алфавит маленький и смещение зададим только по горизонтали (A2). Смещение одинаково для всех букв и не зависит от их ширины.

Для наблюдения и корректировки изображения букв с запасом выделяем массив координат алфавита (F6:G48) и строим диаграмму *Точечная с прямыми отрезками и маркерами*. Получаем алфавит для составления сообщений. При необходимости можно увеличить алфавит.



Теперь можно приступить к созданию массива координат букв сообщения, которые должны туда записываться последовательно с необходимыми знаками препинания. Следовательно, в зависимости от положения в строке и номера строки должны быть вычислены координаты эле-

ментов букв. Также при этом учитывать расстояние между буквами, то есть формировать пропуск. Имитировать клавиатуру будем управляющими кнопками с соответствующими названиями.

Создадим управляющую кнопку для буквы *A*. При щелчке на кнопке формируются координаты элементов буквы с учетом ее положения в строке и номером строки. Координаты записываются в массив букв сообщения (H6:I250). Массив взят с запасом. Для буквы *A* в массив нужно записать 8 строк. Программа кнопки представлена ниже.

```
Private Sub CommandButton3_Click() 'A
Cells(2, 3) = Cells(2, 3) + 1 'Подсчет числа букв
Cells(2, 4) = Cells(2, 4) + 1 'Начисление номера позиции буквы
If (Cells(2, 4) = 1 And Cells(2, 5) = 1) Then
Np = 0 'Номер строки в массиве сообщения
Else
Np = Cells(2, 6)
End If
For i = 0 To 7 'Копирование образа буквы с учетом текущей позиции в массив сообщения
Cells(6 + Np + i, 8) = Cells(6 + i, 2) + 4 * (Cells(2, 4) - 1) 'Запись координат X
Cells(6 + Np + i, 9) = Cells(6 + i, 3) - 4 * (Cells(2, 5) - 1) 'Запись координат Y
Next i
Cells(6 + Np + 2, 8) = nul 'Пробелы между элементами букв
Cells(6 + Np + 2, 9) = nul 'в массиве сообщения
Cells(6 + Np + 5, 8) = nul
Cells(6 + Np + 5, 9) = nul
Cells(2, 6) = Np + Cells(7, 1) 'Следующий номер строки в массиве сообщения
End Sub.
```

При щелчках производится начисление числа букв (символов) в сообщении и номера позиции. Число букв используем для простой индикации, а номер позиции для смещения координат элементов букв в строке и по строкам. Смещение равно 4 единицам, как в строке, так и по строкам (см. оператор цикла). Этого достаточно с учетом того что высота букв равна 2, а ширина 3 единицам (кроме символов).

После начисления счетчиков проверяется условие: является буква 1-ой ? В этом случае текущий номер позиции в массиве равняется 0. Запись начинается в ячейки 6-й строки. С помощью оператора цикла производится перепись координат элементов из массива координат элементов букв с учетом смещения.

После оператора цикла формируем пустые строки между строками координат элементов буквы, которые соответствуют строкам массива B6:C13.

Последняя операция позволяет сформировать номер позиции в массиве сообщения с учетом количества строк элементов буквы и 1 строки, которая обеспечивает отделение следующего символа от записанных координат буквы *А*. Необходимая величина смещения записана для всех букв в ячейках под их названием (A7, A16, A22 и т.д.).

Представим программу процедуры кнопки буквы *У*:

```
Private Sub CommandButton5_Click() 'У
Cells(2, 3) = Cells(2, 3) + 1 'Подсчет числа букв
Cells(2, 4) = Cells(2, 4) + 1 'Начисление номера позиции буквы
If (Cells(2, 4) = 1 And Cells(2, 5) = 1) Then
Np = 0 'Номер строки в массиве сообщения
Else
Np = Cells(2, 6)
End If
For i = 0 To 5 'Копирование образа буквы с учетом текущей позиции
Cells(6 + Np + i, 8) = Cells(15 + i, 2) + 4 * (Cells(2, 4) - 1) 'Запись координат X
Cells(6 + Np + i, 9) = Cells(15 + i, 3) - 4 * (Cells(2, 5) - 1) 'Запись координат Y
Next i
Cells(6 + Np + 2, 8) = nul  'Пробелы между элементами букв
Cells(6 + Np + 2, 9) = nul 'в массиве сообщения
Cells(6 + Np + 5, 8) = nul
Cells(6 + Np + 5, 9) = nul
Cells(2, 6) = Np + Cells(16, 1) 'Следующий номер строки в массиве сообщения
End Sub.
```

Отличия от предыдущей процедуры заключается в количестве циклов записи координат элементов буквы (6-ть, а не 8-мь), количеством формируемых пробелов и константой смещения строки (6-ть, а не 9-ть). Аналогично создаются процедуры для других букв и символов.

Остановимся на кнопке пробел. Программа кнопки представлена ниже.

```
Private Sub CommandButton10_Click() 'Пробел
Cells(2, 3) = Cells(2, 3) + 1 'Подсчет числа букв
Cells(2, 4) = Cells(2, 4) + 1 'Начисление номера позиции буквы
If (Cells(2, 4) = 1 And Cells(2, 5) = 1) Then
Np = 0 'Номер строки в массиве сообщения
Else
```

```
Np = Cells(2, 6)
End If
Cells(2, 6) = Np + 0 'Следующий номер строки в массиве сообщения
End Sub.
```

Пробел фактически является элементом сообщения, но записывать в массив что-то не обязательно. Просто увеличиваем счетчик позиции для следующей буквы (символа). Смещение в последнем операторе равняется 0. Хотя этот оператор можно было бы удалить.

Остается рассмотреть кнопки перехода к следующей строке, обнуления сообщения и подготовки счетчиков к следующей записи другого сообщения.

При щелчке на кнопку под названием Enter обнуляется счетчик номера позиции в строке и увеличивается счетчик строки, что можно увидеть в процедуре кнопки:

```
Private Sub CommandButton2_Click() 'Переход на новую строку
Cells(2, 4) = 0 'Номер позиции буквы в строке
Cells(2, 5) = Cells(2, 5) + 1 'Начисление номера строки
End Sub.
```

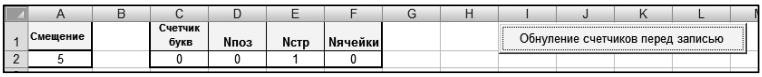
В конце моделирования сообщения ее удаляем с помощью кнопки под названием Del. При этом определяется область обнуления с использованием записанных строк в сообщении с добавлением максимального количества строк отводимых на букву (символ) в процедуре кнопки:

```
Private Sub CommandButton4_Click() 'Удаление сформированного сообщения
N = Cells(2, 6) + 9 'последний номер в массиве сообщения
For i = 0 To N
Cells(6 + i, 8) = nul 'Очистка массива сообщения по координате X
Cells(6 + i, 9) = nul 'Очистка массива сообщения по координате Y
Next i
End Sub.
```

Перед началом формирования нужно установить начальные значения счетчиков. Это производится с помощью кнопки *Обнуление счетчиков перед записью*, процедура которой представлена ниже.

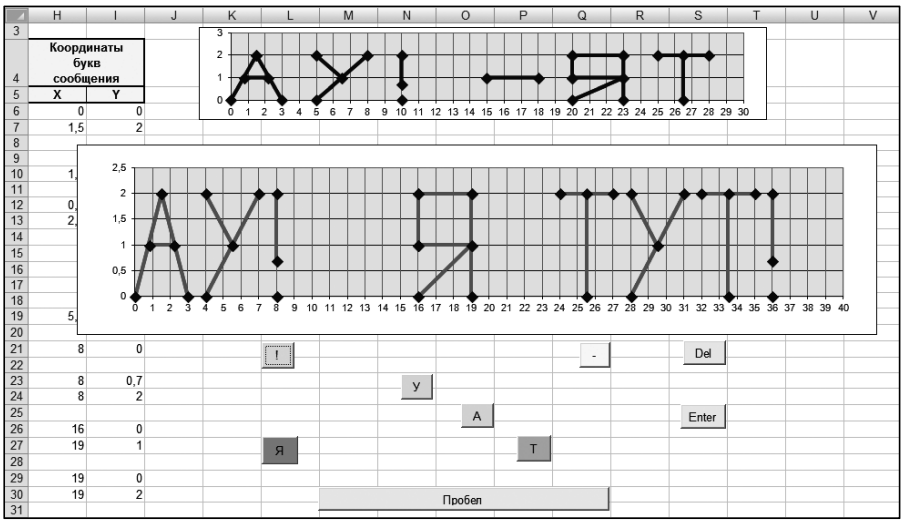
```
Private Sub CommandButton1_Click() 'Обнуление счетчиков перед записью нового сообщения
Cells(2, 3) = 0 'Счетчик букв в сообщении
Cells(2, 4) = 0 'Номер позиции буквы в строке
Cells(2, 5) = 1 'Начальный номер строки
Cells(2, 6) = 0 'Номер ячейки в массиве для текущей записи
End Sub.
```

Таким образом, в начале формирования сообщения нужно щелкнуть на кнопке *Обнуление счетчиков перед записью,* затем последовательно



щелкать на кнопках букв, символов, *Пробел* и для перехода на следующую строку на кнопке *Enter*.

В конце или перед новой запись нужно удалять сообщение с помощью кнопки *Del*.



Желательно при индикации сообщения в диаграмме установить фиксированные минимальное и максимальное значения для оси X. Минимальное равняется нулю, а максимальное – из предположения максимальной длины сообщения (см. рисунок).

При создании дополнительных кнопок алфавита можно обеспечить создание различных сообщений. Целесообразно кнопки располагать в порядке принятом для клавиатуры ПК.

Модель блокнота фактически представляет собой векторный графический редактор, созданный на основе «графических матриц» букв и символов.

2.5.2. Курица, которая пьет воду

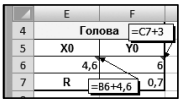
Создадим модель курицы, которая пьет воду из чашки.

Начнем с создания стилизованного изображения курицы с помощью отрезков прямых. Запишем координат прямых контура курицы (B6:C48) в массив *Исходные данные*. Для создания возможностей для смещения изображения зададим опорные точки, к которым будем привязывать координаты других элементов изображения курицы. В качестве опорных точек возьмем координаты ноги курицы (B6, C6, C7). Желательно для наблюдения за построением изображения сразу выделить необходимое количество ячеек массива *Исходные данные* и обратиться к диаграмме *Точечная с*

прямыми отрезками и маркерами. После построения можно будет скорректировать диапазон данных для диаграммы.

На рисунке представлен пример формирования координат ноги курицы с использованием координат опорных точек. Поэтому принципиальных трудностей для получения изображения курицы не должно быть. Если нужно будет переместить курицу по оси X нужно будет внести новое значение в ячейку B6.

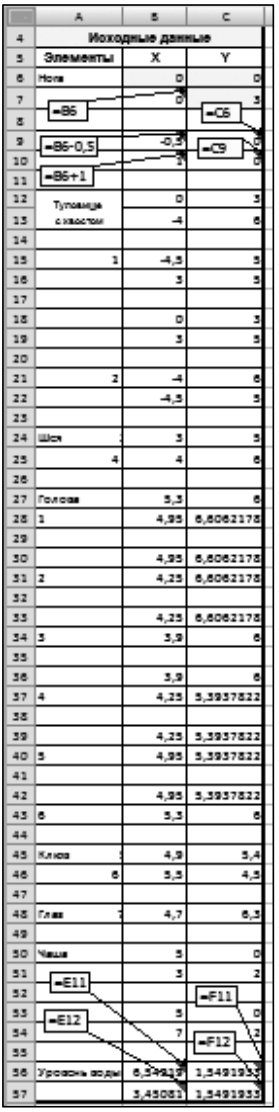
Голова курицы изображается в виде вписанного шестиугольника в окружность. Координаты центра (E6, F6) и радиус окружности (F7) задаем во вспомогательном массиве *Голова*. Методика изображения окружности рассматривалась ранее в других моделях.



Остается задать координаты чаши, форму которой выбираем в виде конуса (B50:C54). Чтобы лучше наблюдать уменьшение уровня воды не создаем изображение верхней кромки чаши.

В массиве выделены ячейки для изображения текущего уровня воды (B56:C57) в виде линии. В начале моделирования считаем, что чаша заполнена водой полностью.

Теперь можно начать моделирование наклона курицы к чаше. Чтобы напиться курица должна наклониться и коснуться клювом уровня воды. Наклон будем моделировать поворотом тела курицы вокруг точки, которая соответствует верхней точки ее ноги (B7, C7).



Сначала определим относительные радиусы и углы наклона характерных точек элементов изображения в исходном положении и создадим вспомогательный массив *Радиусы точек и угол* (E15:G25). Всего выделили 11 характерных точек изображения курицы. Это позволит не усложнять вычисления координат элементов изображения при наклонах.

	D	E	F	G
14		Радиусы точек и угол		
15		1	4,9244289	2,7233683
16		2	4,6055513	0,49809154
17		3	3,6055513	0,5880026
18		4	4,5	0,64350111
19		5	5,4561891	0,45545107
20		6	5,7008771	0,26625205
21		7	5,7428216	0,6121525
22		8	6,090156	0,51507282
23		9	6,1243209	0,62961736
24		10	5,5738054	0,70363528
25		11	4,8777754	0,5129536

=((B15-$B$7)^2+(C15-$C$7)^2)^0,5

=ATAN2((B15-$B$7);(C15-$C$7))

Вычисление радиусов и углов наклона производим по формулам

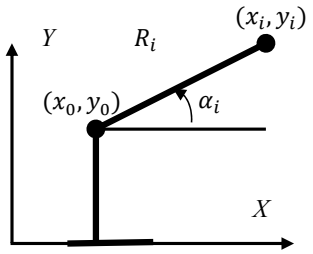
$$R_i = ((x_i - x_0)^2 + (y_i - y_0)^2)^{0,5},$$
$$\alpha_i = arctg((y_i - y_0)/((x_i - x_0))),$$

где R_i - радиус i-ой точки,

x_i, y_i - координаты i-ой точки,

x_0, y_0- координаты опорной 0-ой точки,

α_i- начальный угол i-ой точки.



Следует отметить, что радиус вычисляется относительно опорной точки, а угол наклона вычисляется относительно горизонтальной линии, проходящей через точку (см. примечания).

Определим уровень воды при каждом глотке. Считаем, что чаша в исходном положении заполнена полностью, в ней всего 5-ть глотков воды. Можно задать любое количество глотков. Глотки воды имеют одинаковый объем. Уровень воды зависит от выпитого объема воды. После 5-го глотка вода в чаше отсутствует.

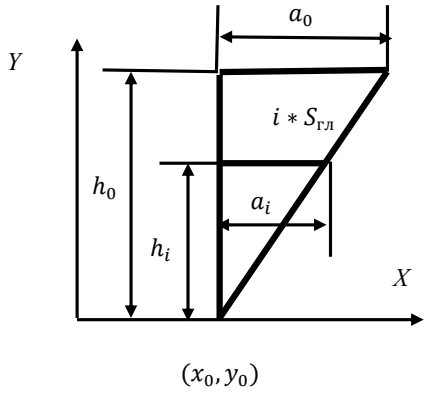
Полный объем воды в чаше определяется объемом конуса. Так как объем конуса пропорционален площади треугольника, который изображен на рисунке, то для простоты полный объем пусть равняется

$$S = a_0 * h_0/2 = (h_0{}^2 * \text{tg} \propto)/2,$$

где a_0 – радиус чаши,

h_0 - высота чаши,

$\text{tg} \propto = a_0/h_0$.



Хочется заметить, что тангенс является неизменным.

Так как в чаше у нас 5-ть глотков, то объем одного глотка будет

$$S_{гл} = S/5.$$

В результате i-го глотка объем оставшейся воды в чаше будет равен

$$S_i = S - i * S_{гл}.$$

Можно записать следующие выражение

$$i * S_{гл} = S - S_i$$

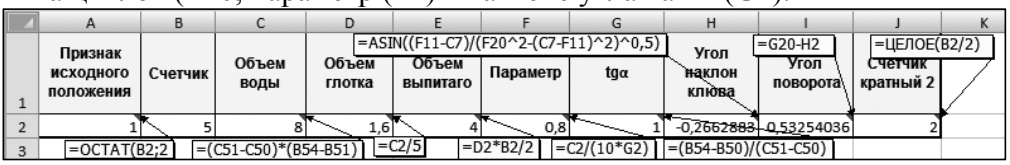
и сделаем подстановки

$$i * S_{гл} = (h_0{}^2 * \text{tg} \propto)/2 - (h_i{}^2 * \text{tg} \propto)/2.$$

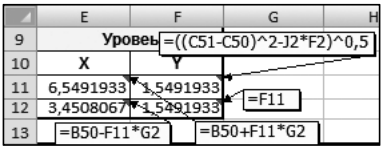
Из последнего выражения находим h_i, где в качестве параметра используем

$$p = S/10 * \text{tg} \propto.$$

Предварительно в ячейках вычисляем полный объем воды в чаше (C2), объем глотка (D2), количество выпитой воды в зависимости от счетчика циклов (E20, параметр (F2) и тангенс угла чаши (G2).



Вычисления координат уровня воды в конкретном цикле моделирования производим в массиве *Уровень*. Расчет высоты уровня производим с использованием счетчика кратного 2-м основного счетчика (J11). Далее вычисляем значения координат линии воды с учетом боковых границ чаши через тангенс. Определяем $a_i = h_i * \mathrm{tg} \propto$ и записываем уровень воды в чашке на i-м глотке.



Вычисленные значения переписываем в массив *Исходные данные* для индикации изображения курицы перед наклоном к воде в чаше.

Введение двух счетчиков связано с упрощением индикации модели курицы. При нечетном значении счетчика (B2) курица находится в вертикальном положении, а при четном – в наклонном положении. Индикация четности осуществляется в ячейке A2, где индицируется 0 или 1 (см. примечание). Так как уровень воды должен изменяться не каждый такт счетчика (B2), а только 1 раз за интервал в 2 цикла, то для расчета уровня используется счетчик в ячейке J11.

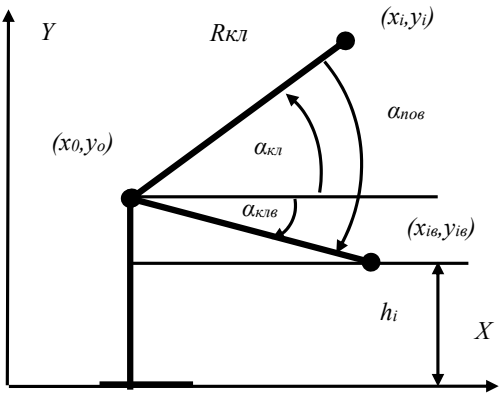
Пора перейти к расчету углов наклона радиусов выбранных точек изображения к текущему уровню воды в чаше. Вначале определим угол наклона радиуса клюва (F20) к уровню воды (см. рисунок).

$$\propto_{\text{клв}} = \sin^{-1}(y_0 - h_i)/(x_{i\text{в}} - x_0).$$

В этой формуле неизвестно значение $x_{i\text{в}}$, которое можно найти из выражения

$$R_{\text{кл}}{}^2 = (y_0 - h_i)^2 + (x_{i\text{в}} - x_0)^2.$$

Хотя нам для вычисления угла нужна величина $(x_{i\text{в}} - x_0)$.

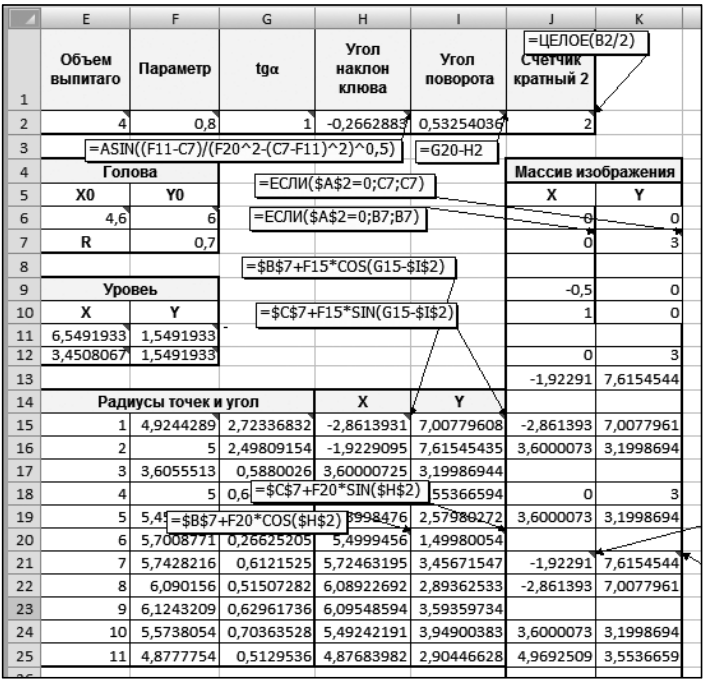


Угол наклона клюва рассчитывается в ячейке H2. Это значение используется для определения координат клюва при наклоне курицы (H20,I20). Все остальные точки концов радиусов поворачиваются на угол поворота, который определяется выражением

$$\alpha_{пов} = \alpha_i - \alpha_{клв}.$$

Прямоугольные координаты точек рассчитываются в зависимости от значения угла наклона клюва курицы при наклонах (H15:I25).

Эти значения перемещающихся точек используются в массиве изображения модели.



	E	F	G	H	I	J	K
1	Объем выпитаго	Параметр	tgα	Угол наклон клюва	Угол поворота	Счетчик кратный 2	
2	4	0,8	1	-0,2662883	0,53254036	2	
4	Голова					Массив изображения	
5	X0	Y0				X	Y
6	4,6	6				0	0
7	R	0,7				0	3
9	Уровень					-0,5	0
10	X	Y				1	0
11	6,5491933	1,5491933					
12	3,4508067	1,5491933				0	3
13						-1,92291	7,6154544
14	Радиусы точек и угол			X	Y		
15	1	4,9244289	2,72336832	-2,8613931	7,00779608	-2,861393	7,0077961
16	2	5	2,49809154	-1,9229095	7,61545435	3,6000073	3,1998694
17	3	3,6055513	0,5880026	3,60000725	3,19986944		
18	4	5	0,6		55366594	0	3
19	5	5,4		3998476	2,57980272	3,6000073	3,1998694
20	6	5,7008771	0,26625205	5,4999456	1,49980054		
21	7	5,7428216	0,6121525	5,72463195	3,45671547	-1,92291	7,6154544
22	8	6,090156	0,51507282	6,08922692	2,89362533	-2,861393	7,0077961
23	9	6,1243209	0,62961736	6,09548594	3,59359734		
24	10	5,5738054	0,70363528	5,49242191	3,94900383	3,6000073	3,1998694
25	11	4,8777754	0,5129536	4,87683982	2,90446628	4,9692509	3,5536659

Имитация движения курицы для простоты состоит из двух положений:

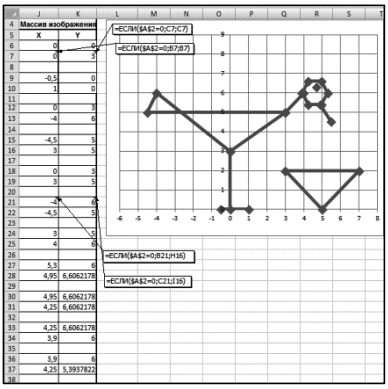
исходного вертикального положения;

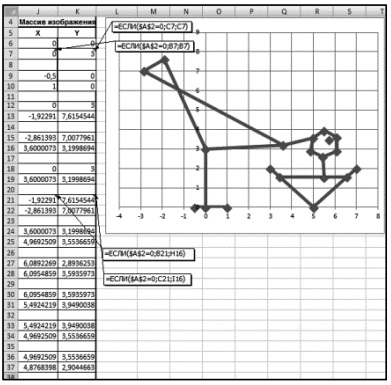
наклона туловища до касания уровня воды в чаше.

При возвращении курицы в исходное положение уровень воды в чаше уменьшается с учетом объема глотка и формы чаши.

Все исходные положения определяются нулевым значением ячейки *Признак исходного положения* (A2). При проверке нулевого значения в *Массив изображения* записываются значения координат элементов курицы, чаши и уровня воды из массива *Исходные данные.* Если условия не выполняются, то формируются координаты подвижных элементов с помощью данных в массиве H15:I25. Элементы обоих массивов полностью соответствуют между собой. Только в конкретные моменты моделирования отличаются значения координат подвижных частей курицы и уровня воды.

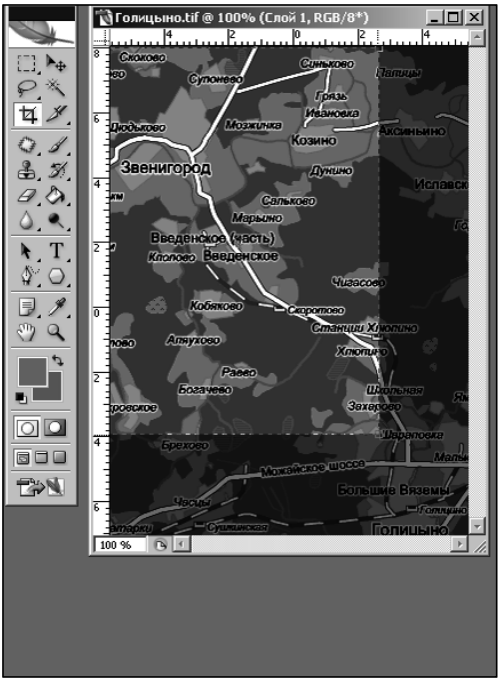
Теперь можно моделировать процесс, в котором только 5 циклов наклона. Для этого нужно только записывать в *Счетчик* (B2) значения от 0 до 10. Автоматически при моделировании фиксируется количество выпитой воды. На рисунках приведены два фрагмента модели.





2.5.3. Построения маршрута движения на карте

С помощью электронной таблицы MS Excel можно изобразить на карте маршрут движения и рассчитать его характеристики. Для этого в первую очередь нужно иметь электронную географическую карту местности, по которой проходит маршрут движения (похода). Сейчас нетрудно найти в Интернете карту любого участка местности.



Методика построения маршрута состоит из следующих основных этапов:

1. Определения «размеров» участка местности, использование его в качестве фона области построения диаграммы с установкой начала и диапазона изменения координат.
2. Построения маршрута.
3. Расчета характеристик маршрута.

В Интернете находим географическую карту нужного участка местности. Самое главное оценить масштаб и систему координат, в которой представлена карта. Обычно походы имеют небольшую протя-

женность. Следовательно, при построении маршрута можно смело применять прямоугольную систему координат.

Итак, выбрав участок местности, нужно сохранить его в файле или сфотографировать экран с картой (нажав кнопку *PrtSc*). Последний вариант более подходит в силу своей универсальности. Поэтому и остановимся на нем.

В приложении Photoshop через меню *Файл* и команду *Новый* создаем файл с записью в диалоговом окне соответствующего имени.

Далее через меню *Редактирование* и команду *Вклеить* вставляем изображение экрана с картой. С помощью инструмента *Рамка* выделяем участок карты. Инструментом выделяем участок карты так, чтобы размеры изображения в километрах имели целые значения. Это обеспечит в дальнейшем целые значения координат основных или промежуточных значений шкал на диаграмме, с которыми более привычно работать. Здесь требуется достаточная аккуратность. Для облегчения работы устанавливаем линейки через меню *Просмотр*, где можем контролировать размеры рамки. Сохраняем выделенный участок карты с кадрированием изображения (меню *Файл*, команда *Сохранить как*) и расширением TIFF. Часто карты имеют недостаточную яркость и контрастность. Можно достаточно просто улучшить изображение через меню *Изображение*, команду *Коррекция* и диалоговое окно *Кривые*.

После этого определяем линейные размеры изображения участка карты в километрах с учетом масштаба. Через меню *Изображение* и команду *Размер изображения* в диалоговом окне находим размеры изображения в сантиметрах (Высота и Ширина). Далее используем масштаб карты (Масштаб) для вычисления размеров изображения (В, Ш) в километрах

В=Масштаб*Высоту и Ш=Масштаб*Ширину,

где В – высота в километрах,

Ш – ширина в километрах.

Вызываем MS Excel и подготавливаем таблицу исходных данных для построения маршрута, где должны быть указаны координаты начала и конца каждого участка маршрута. Количество строк должно соответствовать предполагаемому количеству участков маршрута.

Обращаемся к мастеру диаграмм, выбираем тип диаграммы *Точечная*. По усмотрению значения диаграммы можно соединять отрезками или сглаживающими линиями. Через контекстное меню диаграммы вызываем диалоговое окно *Формат области построения* и выбираем пункт *Заливка*. Далее включаем *Рисунок или текстура* и через *Вставить из* щелкаем на кнопке *Файл*.

Выбираем файл карты и вставляем в качестве фона. Теперь можно по карте установить начало координат. Начало координат целесообразно привязать к точке, из которой начинается маршрут. Для этого используем

последовательно контекстное меню осей x и y. В диалоговом окне *Формат оси* для каждой оси на закладке *Шкала* устанавливаем максимальное и минимальное значения, при которых начало координат совпадает с точкой начала маршрута. При этом соответствующие диапазоны координат должны соответствовать ширине (Ш) или высоте (В) изображению карты. Также задаем цену основных и промежуточных делений. Устанавливаем флажки пересечения осей в максимальных значениях для помещения значений координат по периметру карты.

После задания системы координат приступаем непосредственно к построению маршрута. Для этого в первой строке таблицы заносим нули в ячейки координат x и y. Далее в последующие ячейки заносим значения координат в соответствии с намеченным маршрутом движения и системой координат. При этом на карте автоматически отображается маршрут, который корректируется путем изменения значений координат. Линия маршрута форматируется через *Формат рядов данных*. Там же можно выбрать нужные параметры для линии, маркера и установить стрелки на линиях отрезков маршрута.

После построения маршрута рассчитываем параметры участков: длину участка (d), время (t) и азимут (A).

Длину участка определяем по координатам начала и конца участка (см. примечания)

$$d_{i+1} = ((x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2)^{0,5} ,$$

где d_{i+1} - длина ($i+1$) участка маршрута,

(x_i, y_i) и (x_{i+1}, y_{i+1}) - координаты начала и конца ($i+1$) участка маршрута,

$i=0,1,2,....$

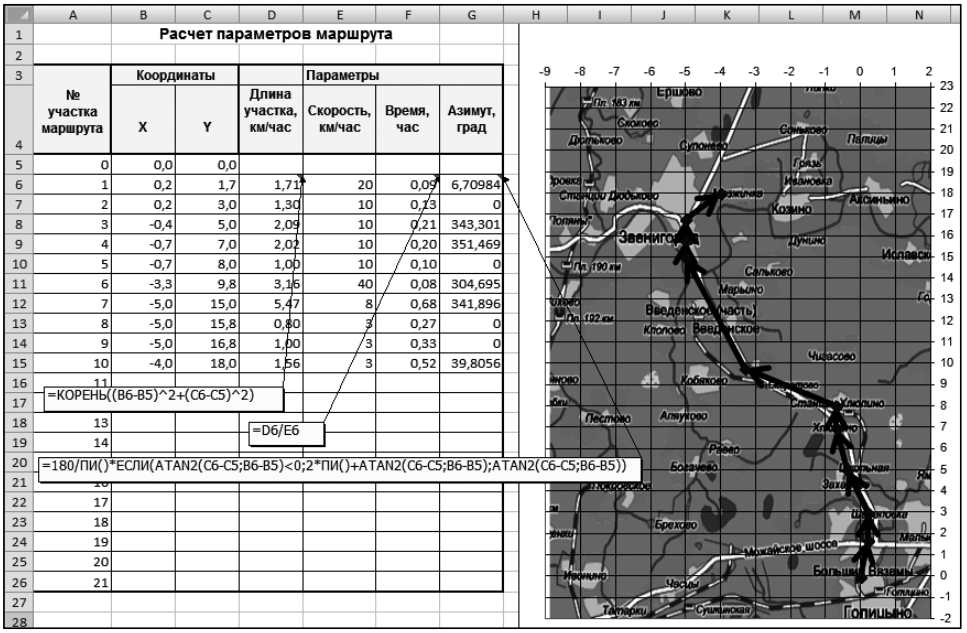
Время рассчитываем по заданной скорости движения (v) и длине участка

$$t_{i+1} = d_{i+1}/v_{i+1}.$$

Азимут считаем по формуле

$$A_{i+1} = arctg((x_{i+1} - x_i)/(y_{i+1} - y_i)) .$$

Для перевода значений азимута из радиан в градусы производим умножение на 180/π. Так как азимут определяется по часовой стрелке от направления на север, то с учетом особенностей расчета функции арктангенса вводим дополнительные условия для оценки азимута (см. примечание).



Расчет параметров маршрута

№ участка маршрута	Координаты X	Координаты Y	Длина участка, км/час	Параметры Скорость, км/час	Параметры Время, час	Параметры Азимут, град
0	0,0	0,0				
1	0,2	1,7	1,711	20	0,09	6,70984
2	0,2	3,0	1,30	10	0,13	0
3	-0,4	5,0	2,09	10	0,21	343,301
4	-0,7	7,0	2,02	10	0,20	351,469
5	-0,7	8,0	1,00	10	0,10	0
6	-3,3	9,8	3,16	40	0,08	304,695
7	-5,0	15,0	5,47	8	0,68	341,896
8	-5,0	15,8	0,80	3	0,27	0
9	-5,0	16,8	1,00	3	0,33	0
10	-4,0	18,0	1,56	3	0,52	39,8056
11						
13						
14						
17						
18						
19						
20						
21						

=КОРЕНЬ((B6-B5)^2+(C6-C5)^2)

=D6/E6

=180/ПИ()*ЕСЛИ(ATAN2(C6-C5;B6-B5)<0;2*ПИ()+ATAN2(C6-C5;B6-B5);ATAN2(C6-C5;B6-B5))

В заключении можно с помощью функции автосуммы вычислить длину маршрута, общее время в пути.

2.5.4. Часы

Создадим модель часов с часовой, минутной и секундной стрелками. Часы должны показывать реальное время.

Начнем с циферблата часов. Обычно на циферблате по окружности располагаются 60 меток, которые соответствуют числу секунд в минуте. Зададим в ячейках угловые шаги для делений секунд, минут и часов (A3:C3). Для секунд нужно разделить 360^0 на 60. Чтобы изобразить окружность зададим в условных единицах ее радиус (E3). Вычислим для 60 значений углов координаты в прямоугольной системе координат. Центр циферблата расположим в начале координат. Создадим массив A6:D66, в котором вычислим координаты для всех точек минутных меток. Для замыкания окружности в последней ячейке массива продублируем координаты первой точки. Выделим с запасом ячейки массива и обратимся к диаграмме *Точечная с гладкими кривыми и маркерами.*

Шаг в радианах			Радиус	
Секунд	Минут	Часа	Циферблата	Метки
0,10472	0,00873	0,523599	10	9
=2*ПИ()/60	=2*ПИ()/(60*6*2)	=2*ПИ()/(12)		

Секунды	Угол, рад	X	Y
0	0	0,00	10,00
1	0,10472	1,05	9,95
2	0,20944	2,08	9,78
3	0,31416	3,09	9,51
4	0,41888	4,07	9,14
5	0,5236	5,00	8,66
6	0,62832	5,88	8,09
7	0,73304	6,69	7,43
8	0,83776	7,43	6,69
9	0,94248	8,09	5,88
10	1,0472	8,66	5,00
11	1,15192	9,14	4,07
12	1,25664	9,51	3,09
13	1,36136	9,78	2,08
14	1,46608	9,95	1,05

=A6*$A$3

=$E$3*SIN(B6)

=$E$3*COS(B6)

В ячейках C68: D75 будем вычислять коор-

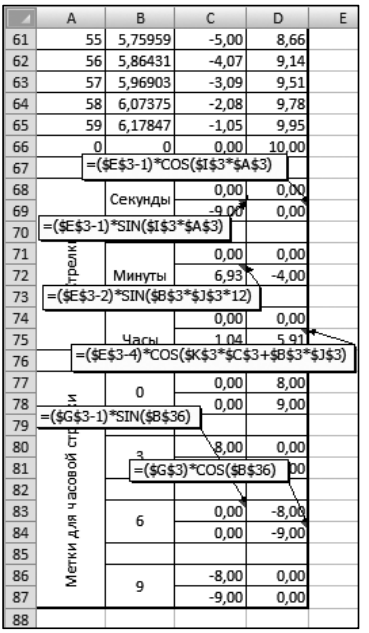
динаты стрелок. Начало отрезков стрелок привязываем к началу координат. Координаты окончаний стрелок рассчитываем с учетом текущих значений часов, минут, секунд и цен угловых соответствующих делений. Формулы вычислений углов стрелок приведены ниже:

$$\alpha_s = \Delta\alpha_s * s,$$
$$\alpha_m = \Delta\alpha_m * m * 12,$$
$$\alpha_h = \Delta\alpha_h * h + \Delta\alpha_m * m,$$

где $\alpha_s, \alpha_m, \alpha_h$ - углы секундной, минутной и часовой стрелок, отсчитываемые от оси ординат по ходу часовой стрелке,

$\Delta\alpha_s$, $\Delta\alpha_m$, $\Delta\alpha_h$ - шаги изменения углов секундной, минутной и часовой стрелок соответственно,

s, m, h - текущие значения секунд, минут, часов реального времени.

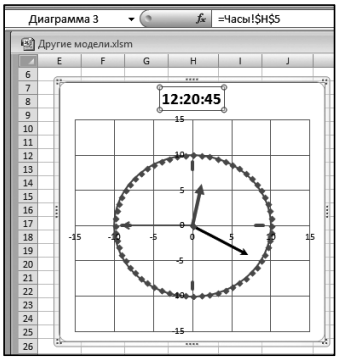


При таком задании углов секундная стрелка пробегает по всем меткам циферблата. Минутная стрелка также, но только с интервалами равными 60 секундам. Часовая – движется в угловом секторе часа с интервалами равными 1 минуте. Текущие значения времени выбираются из ячеек I3:K3.

	I	J	K
1	Время		
2	Секунды	Минуты	Часы
3	45	20	12

Сформируем для наглядности на циферблате 4-е метки для часовой стрелки. Для этого продолжим массив и сформируем укороченные отрезки меток для 0,3,6 и 9 часов. Длина отрезков равна 1 условной единице (см. примечание).

Изменим при необходимости диапазон данных в диаграмме с учетом того, что массив увеличился C6:D87. Реалистичность изображения стрелок и меток обеспечим форматированием маркеров и линий отрезков. Щелкаем дважды на нужном маркере отрезка и через контекстное меню *Формат точки данных* в диалоговом окне выбираем *Параметры маркера* и устанавливаем *Нет*, формируем стрелки в пункте *Тип линий*, а также толщину линий. Можно изменить цвет линий. Аналогично убираем маркеры для меток часовой стрелки и делаем их в виде штрихов.



	H	I	J	K
1		Время		
2		Секунды	Минуты	Часы
3		45	20	12
4				
5	12:20:45		Будильник	15:58:00

Осталось формировать текущие значения времени для моделирования движения стрелок.

В первую очередь определимся с ячейками, в которых будут записываться значения текущего времени. В ячейке H5 индицируется текущее время. В ячейках I3:K3 значения секунд, минут и часов, которые используются для вычисления положения соответствующих стрелок. В ячейку K5 заносим значение времени для будильника. В ячейках, где указываются часы, минуты и секунды, устанавливаем формат ч:м:с.

Для индикации времени в цифровом формате активизируем элемент названия диаграммы. Затем в окне *Строка формул* ставим знак равенства и щелкнем по ячейке H5. После щелчка по *Enter* в элементе названия диаграммы появится реальное время в цифровом масштабе.

Создаем через *Разработчик* и *Элементы управления* три кнопки и через их свойства формируем названия. Далее нужно вставить соответствующие программы, которые выполняются по щелчку. Дважды щелкаем на каждой кнопке в режиме *Конструктора* и получаем заготовки для программ. В файле *Основы графики и счетчики* открываем лист *Счетчик PMB* и копируем последовательно программы соответствующих кнопок, которые вставляем во вновь созданные заготовки. Производим их коррекцию с учетом значений ячеек времени.

Получаем следующие программы кнопок

```
Sub CommandButton1_Click() 'Запуск часов по щелчку
Call Bklshetchika 'Вызов процедуры включения счетчика
 End Sub
Sub CommandButton2_Click() 'Останов часов по щелчку
Call Stopchet 'Вызов процедуры остановки счетчика
End Sub

Sub CommandButton3_Click() 'Обнуление часов по щелчку
Cells(3, 9) = 0 'Обнуление ячейки счетчика Секунды
Cells(3, 10) = 0 'Обнуление ячейки счетчика Минуты
Cells(3, 11) = 0 'Обнуление ячейки счетчика Часы
Cells(5, 8) = Null 'Обнуление ячейки счетчика времени Ч.М.С.
End Sub.
```

Остается скопировать процедуры PMB и вставить их в проект *Часы*. Процедуры так же корректируем.

```
Dim Zadaniecledtime As Date
Sub Zapistime() 'Процедура записи текущего значения времени
Cells(5, 8) = Now 'Запись текущего значения времени
Cells(3, 11) = Hour(Now)
Cells(3, 10) = Minute(Now)
Cells(3, 9) = Second(Now)
Call Bklshetchika
```

*If ((Hour(Now) * 60 + Minute(Now)) = (Hour(Cells(5, 11)) * 60 + Minute(Cells(5, 11)))) Then*

MsgBox "Пора вставать!", vbExclamation, "Выключить будильник" 'Включение будильника

End If

End Sub

Sub Bklshetchika() 'Запуск счетчика с интервалом 1 секунда

Zadaniecledtime = Now + TimeValue("00:00:01") 'Запись значения времени для запуска счетчика с учетом интервала

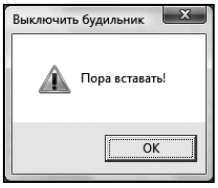
Application.OnTime Zadaniecledtime, "Zapistime" 'Вызов процедуры записи счетчика времени в заданное время

End Sub

Sub Stopchet() 'Процедура остановки счетчика

Application.OnTime EarliestTime:=Zadaniecledtime, Procedure:="Zapistime", Schedule:=False 'Формирование запрета вызова процедуры записи значения времи счетчика

End Sub.



Дополнительно в процедуру записи текущего значения счетчика вставляем проверку для отображения окна сообщения при совпадении текущего значения времени с временем будильника с точностью до минуты (выделено курсивом). Для продолжения функционирования часов по истечении минуты индикации окна будильника нужно щелкнуть на кнопке ОК. До этого момента стрелки часов замрут. Можно момент появления сообщения определить с точностью до секунды. В этом случае в условия проверки нужно добавить значения секунд.

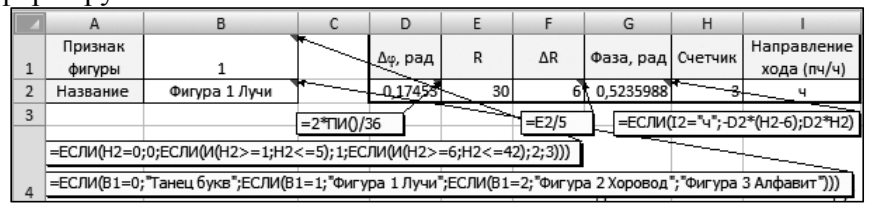
Все модель часов создана.

2.5.5. Алфавитный хоровод

Нужно знать свой родной алфавит. Для его запоминания и углубления знаний по работе с элементами диаграмм создадим модель движения букв алфавита в области построения диаграммы.

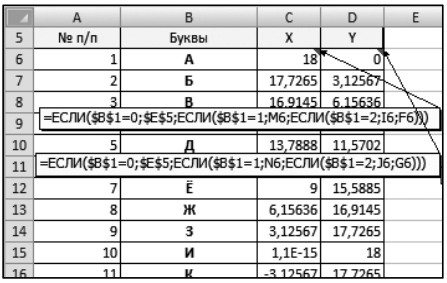
Выберем движения, которые связаны с танцами. Пусть будут в танце четыре положения: исходное, лучи, хоровод и алфавит. При исходном положении все буквы сосредоточены в начале координат. В следующем положении буквы от центра расходятся по лучам на расстояние равное заданному радиусу. При хороводе буквы совершают движение по окружности. При окончании моделирования буквы выстраиваются по нескольким горизонтальным рядам в виде таблицы.

Для индикации положения или фигуры создаем признак (B1), который формируется в зависимости от счетчика тактов.



Каждая фигура связана с текущим значением счетчика и продолжительностью. Начало и продолжительность задается произвольно (см. примечание к ячейке B1). Для индикации названий фигур сформируем ячейку B2, где по признаку фигуры формируется текст (см. примечание).

Теперь нужно создать массив, который будет использоваться для изображения фигур модели. В массиве A6:D37 заносим буквы и формируем ячейки координат для расположения букв в соответствии с признаком фигуры.



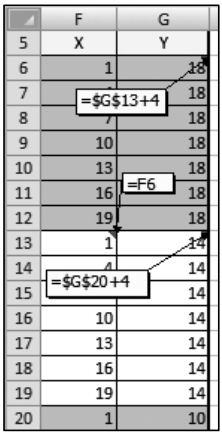
Для буквы А используем следующие выражения:

ЕСЛИ($B$1=0;$E$5;ЕСЛИ($B$1=1;M6;ЕСЛИ($B$1=2;I6;F6)))

ЕСЛИ($B$1=0;$E$5;ЕСЛИ($B$1=1;N6;ЕСЛИ($B$1=2;J6;G6))).

По признаку выбираются координаты из дополнительных массивов, где вычисляются координаты букв в соответствии с фигурой. В исходном положении все координаты берутся из пустой ячейки E5. Такой подход позволяет устранить нагромождения в вычислениях.

Для последней фигуры создаем массив F6:G37. В первой строчке формируем координаты для буквы А. Таблица состоит из строк, в которых 7 букв (в последней строке 4 буквы). Координаты по оси X формируем автозаполнением и копируем далее для других строк. Ячейки строк выделены цветом. Последние буквы алфавита располагаются в нижней строке. Поэтому для строк формируем координаты по оси Y с шагом 4. При этом учитываем координаты предыдущей строки. Можно задать координаты в другом удобном порядке путем занесения их значения с клавиатуры.



Для формирования движения букв по лучам задаем максимальный радиус окружности и разбиваем 360^0 на 36 (D2). Определяем количество тактов передвиже-

ния по лучу и вычисляем шаг изменения радиуса луча (F2). В ячейке H2 формируются счетчик тактов.

Координаты букв на луче в массиве M6:N37 определяются по формулам

$$x_i = \Delta R * t_i * \cos((n-1) * \Delta\varphi),$$

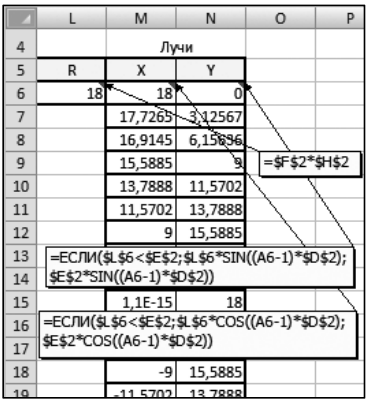
$$y_i = \Delta R * t_i * \sin((n-1) * \Delta\varphi),$$

где ΔR – шаг изменения радиуса луча,

t_i – значение счетчика тактов,

n – номер по порядку (для А равен 1),

$\Delta\varphi$ – шаг угла лучей.

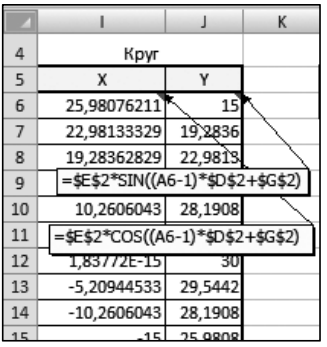


Однако расчет положения буквы на луче ведется до достижения значения радиуса максимального значения (E2). Далее фиксируется значение радиуса, который предварительно вычисляется для всех лучей в ячейке L6 по формуле $R = \Delta R * t_i$.

То есть буквы из начала координат за 5 тактов перемещаются по лучу и располагаются по окружности с радиусом 30 единиц. Буква А располагается на оси X, а остальные против часовой стрелки по принятому расположению букв в алфавите.

Остался массив хоровода букв (I6:J37), для которого вычисляет координаты букв с учетом их смещения на окружности в зависимости от значений счетчика тактов и заданного направления. Направление задаем в ячейке I2: по часовой стрелке заносим «ч», против часовой стрелки – «пч».

Положение вычисляем по вышеприведенным формулам, но радиус имеет максимальное значение R и учитывается текущая фаза φ, которая определяет положение букв на окружности при движении.



$$x_i = R * \cos((n-1) * \Delta\varphi + \varphi),$$

$$y_i = R * \sin((n-1) * \Delta\varphi + \varphi),$$

где $\varphi = \Delta\varphi * (t_i - t_0)$.

Однако для организации движения по часовой и против часовой стрелки производится проверка и вычисление по выражению

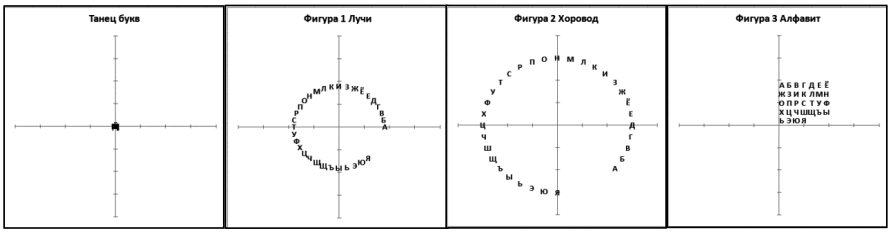
ЕСЛИ(I2="ч";-D2*(H2-6);D2*H2).

При этом $t_0 = 6$. До этого момента буквы перемещаются по лучам в течении 5 тактов. Если движение происходит по часовой стрелки, то фаза должна быть отрицательной.

Фактически сценарий определяется выражением в ячейке B1

ЕСЛИ(H2=0;0;ЕСЛИ(И(H2>=1;H2<=5);1;ЕСЛИ(И(H2>=6;H2<=42);2;3))).

В диапазоне от 1 до 5 тактов буквы движутся по лучам от центра координат, после на 6 такте до 42 такта совершают полный оборот и в конце выстраиваются в таблицу.



Построение изображения алфавитного хоровода начинается с выделения массива C6:D37 и обращения к диаграмме *Точечная с маркерами*. Желательно это делать при изображении хороводы. Далее через контекстное меню ряда выбираем пункт *Добавить подписи ряда*. Сразу можно через пункт *Формат подписей ряда* установить положение подписи *В центре.* Это позволит в дальнейшем симметрично установить буквы относительно начала координат. Так как положение букв будет меняться, то нужно зафиксировать максимальные и минимальные значения осей.

Заменим числа в окнах подписей данных на соответствующие буквы. Активизируем окно для буквы А, установим знак равенства в окне *Строка формул* и щелкнем на ячейке с буквой А (B6). Сформируется адрес на эту ячейку. После щелчка по *Enter* увидим букву А. Итак для всех подписей. По окончанию можно убрать маркеры.

Чтобы наблюдать индикацию названия фигур в поле названия диаграммы устанавливаем курсор, в окне *Строка формул* вставляем знак равенства и щелкаем по ячейке B2 с названиями фигур. В ходе моделирования для каждой фигуры будет высвечиваться свое название.

Сценарий последовательности и длительности выполнения фигур можно изменить. Конечно, можно добавить другие фигуры. Но трудности здесь заключаются в согласовании переходов между фигурами танца букв.

Литература

1. Алексеев А.П. Информатика 2002. М.: СОЛОН-Р, 2002.-400 с.

2. Бортаковский А.С., Закалюкин В.М., Скуридин А.М., Шапошников В.П. Экзаменационные задачи по математике: учеб. пособие.- М.: Изд-во МАИ, 1999. 256с.

3. Дубошин Г.Н. Небесная механика. Основные задачи и методы. М.: Наука, 1968.

4. Иванов Н.М., Лысенко Л.Н. Баллистика и навигация космических аппаратов. М.: Дрофа, 2004.

5. Кильдишов В.Д. Использование Microsoft Excel для построения маршрута движения. //Информатика и образование. 2007.№6.

6. Кильдишов В.Д. Разработка динамических информационных моделей с помощью электронной таблицы MS Excel. //Информатика и образование. 2008. №6.

7. Кильдишов В.Д. Использование электронной таблицы MS Excel для моделирования. //Информатика и образование. 2008. №8.

8. Кильдишов В. Д. Решение графо-аналитическим методом задач линейного программирования с помощью MS Excel. //Информатика и образование. 2009. №5.

9. Кильдишов В.Д. Использование пользовательского формата при моделировании фигур Лиссажу с помощью электронной таблицы MS Excel. //Информатика и образование. 2011. №9.

10. Кильдишов В.Д. Комплексное использование графоаналитического метода, условного форматирования и процедуры поиск решения для нахождения корней уравнений. //Информатика в школе. 2013. №9.

11. Кильдишов В.Д. Решение задач теории стрельбы и баллистики с использованием приложения MS Excel. Краснознаменск: издатель ИП Тюренкова Л.А. 2011.-77с.:ил.

12. Кильдишов В.Д., Склюева О.Н. Использование графических возможностей MS Excel для решения алгебраических задач. Материалы IV международной НПК в 4-х томах, 14-15 октября 2010 года. Одинцово, АНОО ВПО «ОГИ» 2010. Том 4 -298 с.

13. Кильдишов В.Д. Трехмерное моделирование полета спутника вокруг Земли с помощью электронной таблицы MS Excel. Материалы секционных заседаний VII НПК с международным участием 25-29 апреля 2011 года. В 4-х т, том 4. Одинцово, АНОО ВПО «ОГИ» 2011. -310 с.

14. Полет космических аппаратов: Примеры и задачи: Справочник/ Авдеев Ю.Ф., Беляков А.И., Брыков А.В. и др.; Под общ. Ред. Титова Г.С.. – 2-е изд., перераб. И доп. – М.: Машиностроение, 1990.

Издательство «СОЛОН-ПРЕСС» представляет КНИГИ-ПОЧТОЙ

А. А. Афонский, В. П. Дьяконов
Измерительные приборы и массовые электронные измерения

Серия «Библиотека инженера»
Под ред. проф. В. П. Дьяконова

Описаны самые современные измерительные приборы: измерители R, C и L, мультиметры, измерительные ВЧ- и НЧ-генераторы, импульсные и функциональные генераторы, аналоговые и цифровые стационарные и портативные осциллографы, в том числе уникальные. Особое внимание уделено массовым дешевым (бюджетным) приборам и технике измерений, в том числе с применением виртуальных и компьютеризированных лабораторий, и их применению в практике электронных измерений. Ряд материалов посвящен работе с современными цифровыми осциллографами и функциональными генераторами. Рассмотрена современная элементная база и схемотехника измерительных устройств. В книге около шестисот иллюстраций и осциллограмм. Для работников служб ремонта и сервиса сложной электронной техники, научных работников и инженеров, студентов, аспирантов, преподавателей и лаборантов вузов и университетов, а также для подготовленных радиолюбителей.

Б. Ю. Семенов
Микроконтроллеры MSP430: первое знакомство

Серия «Библиотека инженера»

Книга посвящена микроконтроллерам серии MSP430, которые производятся фирмой Texas Instruments. Едва ли удастся найти конкурента этим микроконтроллерам по величине потребляемого тока и производительности, если речь идет о разработке автономных малогабаритных приборов с низковольтным батарейным питанием. Книга в доступной форме поможет разобраться с архитектурой и системой команд MSP430, сориентирует читателя в многообразии «софта» для разработки программного обеспечения, расскажет о способах «прошивки» памяти MSP430, о существующих программаторах. Приведенные схемы, печатные платы и сборочные рисунки позволят собрать несколько несложных программаторов flash-памяти MSP430 самостоятельно.

Книга адресована инженерам, начинающим работать с микроконтроллерами, студентам радиотехнических специальностей вузов, специалистам, занимающимся обслуживанием и ремонтом электронной аппаратуры, радиолюбителям, а также всем тем, кто интересуется перспективной электронной техникой.

А. А. Титов
Транзисторные усилители мощности МВ и ДМВ. Расчет, изготовление, настройка

Серия «Библиотека инженера»

В книге впервые систематически изложены вопросы схемотехнической реализации и расчета наиболее известных и эффективных схемных решений построения отдельных узлов сверхширокополосных и полосовых усилителей мощности метрового и дециметрового диапазонов волн: схем стабилизации режимов; цепей коррекции, согласования, фильтрации и формирования амплитудно-частотных характеристик; устройств защиты усилителей от перегрузок; способов повышения выходной мощности.

Приведены описания схемных решений, методики изготовления и настройки 28 различных вариантов сверхширокополосных и полосовых усилителей мощности с представлением чертежей печатных плат и фотографий макетов усилителей.

Для разработчиков радиоаппаратуры, радиолюбителей, а также студентов и аспирантов.

В. Л. Карякин
Цифровое телевидение

Серия «Библиотека инженера»

Книга посвящена рассмотрению общих принципов построения систем цифрового телевидения, компрессии цифровых сигналов изображения и звука. Приводятся стандарты цифровой компрессии MPEG. Обсуждаются методы построения аппаратуры спутникового, кабельного и эфирного цифрового телевидения, а также особенности методов построения аппаратуры цифрового телевизионного вещания в сетях передачи данных. Даны примеры конфигурации систем цифрового телевизионного вещания.

Рассматриваются этапы развития нелинейного монтажа. Особое внимание уделяется современным цифровым системам видеомонтажа. Рассмотрены требования к оборудованию, предназначенному для монтажа видеофильмов. Дана сравнительная оценка методов линейного и нелинейного видеомонтажа. Приводится технология основных этапов создания видеофильмов.

Обсуждаются особенности технологии эксплуатации оборудования цифрового телевидения. Представлены методы телевизионных измерений, мониторинга качества в цифровом телевидении. Даны основные характеристики контрольно-измерительного оборудования для проверки и анализа потоков MPEG-2, приемо-передающего оборудования телевизионных центров. Приводятся эксплуатационные данные по отечественному и зарубежному оборудованию систем цифрового телевидения.

Книга предназначена для широкого круга читателей, интересующихся информационными технологиями обработки сигналов в цифровом телевидении, основами построения систем цифрового телевидения, перспективами перехода от аналогового телевидения к цифровому. Материал книги может быть полезен специалистам, занимающимся проектированием и эксплуатацией систем цифрового телевидения, студентам высших и средних специальных заведений, специализирующимся в области телевидения.

А. А. Афонский, В. П. Дьяконов
Цифровые анализаторы спектра, сигналов и логики
Под ред. проф. В. П. Дьяконова

Серия «Библиотека инженера»

Впервые в отечественной литературе дается описание современных цифровых анализаторов спектра, сигналов (в том числе близких к монохромным и телекоммуникационных), цепей и логических состояний цифровых устройств. Особое внимание уделено анализаторам спектров и сигналов реального времени. Описаны как стационарные, так и переносные приборы с питанием от аккумуляторных батарей, а также приборы-приставки к персональному компьютеру. Содержит наиболее полный обзор приборов этих классов на российском рынке. Приведены многочисленные примеры реальной работы с приборами. Для инженеров, научных работников, преподавателей, аспирантов и студентов вузов и университетов технического профиля.

Д. А. Соснин
Электрическое, электронное и автотронное оборудование легковых автомобилей (Автотроника-3)
Учебник для вузов

Серия «Библиотека студента»

Полноцветное издание.

В учебнике изложены основные сведения о конструктивных особенностях, принципах действия, эксплуатационных характеристиках электрических, электронных и автотронных систем бортовой автоматики современных легковых автомобилей. Описаны составные компоненты систем, диагностирование и ремонт некоторых из них. Уделено особое внимание нетрадиционным бортовым устройствам, которые ранее не применялись на автомобилях.

Учебник является переработанным и значительно дополненным третьим изданием учебного пособия «Автотроника» ранее написанного для студентов старших курсов автотранспортного факультета МАДИ (ГТУ) по элективной дисциплине «Электрическое и электронное оборудование импортных автомобилей».

Учебник ориентирован на студентов машиностроительных вузов и технических университетов, обучающихся на бакалавра и магистра по профессионально-образовательной программе «Электротехнические и электронные системы наземных транспортных средств». Книга может быть полезна инженерно-техническим работникам предприятий автомобильной промышленности.

Вам необходим прорыв в бизнесе? Обучите этому сотрудников!

Этим уже воспользовались Siemens, Samsung, Motorola, Mitsubishi, Nokia, Huawei...

Мировой и европейский опыт ТРИЗ-обучения в России

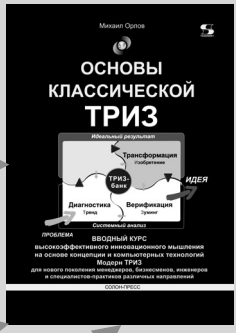
Впервые: непрерывное ТРИЗ-образование для создания инноваций в бизнесе, электронике, машиностроении, энергетике и ТЭК, экономике и гуманитарных областях ...

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС

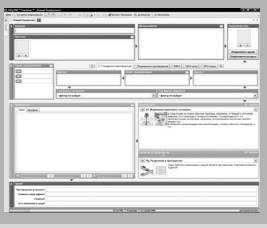
Для школьников, студентов любого профиля и всех начинающих – **сертификат МТРИЗ Ученик**

Для специалистов любого профиля, магистрантов и студентов – **сертификаты МТРИЗ Юниор и Практик**

Для продвинутых пользователей-практиков и как завершающая часть для **сертификата МТРИЗ Практик**



УЧЕБНО-ПРАКТИЧЕСКОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ (СОФТВЕР) трех уровней



Дистанционное Интернет-обучение и сертификация в Модерн ТРИЗ Академии

Инновационные подходы в методике обучения с использованием дистанционных форм и индивидуальных контактов, опирающиеся на 10-летний опыт успеха.

Партнер
международной академии **МТРИЗ** в России
Контактное лицо:
Митин Владимир Александрович

Заказ Интернет обучения, семинаров и WEB-конференций
www.solon-press.ru, e-mail: triz@coba.ru
Телефон: (499) 254-44-10, 795-73-26

Кильдишов Вячеслав Дмитриевич

Использование приложения MS Excel для моделирования различных задач

Ответственный за выпуск: Митин В. А.

Компьютерная верстка: Тюренков М. В.
Технический редактор: Тюренков М. В.

ООО «СОЛОН-Пресс»
123001, г. Москва, а/я 82
Телефоны: (499) 254-44-10, (499) 252-25-21
E-mail: avtor@solon-press.ru,
www.solon-press.ru

По вопросам приобретения обращаться:
ООО «ПЛАНЕТА АЛЬЯНС»
Тел: (499) 782-38-89, **www.alians-kniga.ru**

По вопросам подписки на журнал «Ремонт & Сервис» обращаться:
ООО «Ремонт и Сервис 21»
тел.: (499) 795-73-26, **www.remserv.ru**

ООО «СОЛОН-Пресс»
115142, г. Москва, Кавказский бульвар, д. 50
Формат 70×100/16. Объем 10 п. л. Тираж 200 экз.