МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ ИСЛАМА КАРИМОВА

Информационные технологии в технических системах

Методические указания к выполнению лабораторных работ

ТАШКЕНТ 2022

УДК 681.3

Методические указания к выполнению лабораторных работ по курсу «Информационные технологии в технических системах». Составители: Д.К.Каримова, Ч.С.Хашимова, С.М.Хамдамова - Т.:ТашГТУ, 2022-64с.

Методические указания предназначены для студентов-бакалавров энергетического факультета при проведении лабораторных работ по курсу «Информационные технологии в технических системах». Предлагаемые методические указания ставят своей целью, наряду с лекционными и практическими видами занятий, углубить знания по предмету «Информационные технологии в технических системах» и получить основные навыки и умения по базовым разделам учебного курса.

Печатаются по решению научно-методического совета Ташкентского государственного технического университета.

Протокол №10 от 29 июня 2022г.

Рецензенты:

к.ф.м.н., доц. Маматкулова М.Ш. (НУУз);
к.т.н., доц. М.М. Кадыров (ТашГТУ)

Введение

В методических указаниях приведены описания лабораторных работ, основные объекты и принципы автоматизации процессов управления электронной документацией, изучение MathCad и MatLab, изучение пакетов программ Compas3D, моделирование чертежей по отраслям энергетики, создание динамических сайтов, особенности поиска информации в сети Internet и поисковые работы в сетях, выявление ресурсов сети Internet на документационное обеспечение управления.

Описание каждой работы включает в себя краткую теоретическую часть, формулировку цели, упражнения и задания к лабораторной работе. После каждой работы приведены вопросы для самоконтроля.

Выполнение студентами лабораторных работ направлено на:
обобщение, систематизацию, углубление, закрепление полученных теоретических знаний по конкретным темам;
формирование умений применять полученные знания на лекции и практике, реализацию единства интеллектуальной и практической деятельности; развитие информационных умений: аналитических, проектировочных, конструктивных и др.
Методические указания содержат описание самих работ, необходимый краткий теоретический материал для их выполнения, примеры выполнения и варианты исходных данных для лабораторных работ. Кроме этого, есть примеры выполнения этих работ. После выполнения каждой работы студент должен предоставить в письменном виде отчёт о проделанной работе, в который входят исходные данные, полученные результаты и выводы.

Лабораторная работа №1
Проведение вычислительных экспериментов и численный анализ математических моделей при решении инженерных задач в системе MathCad.

Цель работы: Изучить методику реализации математических моделей в математическом редакторе MathCAD.

Порядок выполнения работы:

1. Изучить теоретическую часть.
2. Ознакомиться с методикой решения различных уравнений с использованием возможностей инструментария системы MathCAD.
3. Реализовать заданные математические модели и получить результаты.
4. Подготовить отчет по проделанной работе.

Теоретическая часть

Для решения уравнений в системе используются как аналитические, так и численные методы. Можно решать алгебраические, трансцендентные уравнения. Эти методы применяются также для систем уравнений.

Для уравнений вида f(x) = 0 решение находится с помощью функции: root(f(x),x,a,b), которая возвращает значение x, принадлежащее отрезку [a, b], при котором выражение или функция f(x) обращается в 0. Оба аргумента этой функции x и f(x) должны быть скалярами, а аргументы a, b – являются необязательными и, если используются, то должны быть вещественными числами, причем a < b. Предварительно, перед заданием функции должно быть указано значение начального приближения. Функция позволяет находить не только вещественные, но и комплексные корни уравнения (при выборе начального приближения в комплексной форме).

Если уравнение не имеет корней, они расположены слишком далеко от начального приближения, начальное приближение было вещественным, а корни – комплексные, функция f(x) имеет разрывы (локальные экстремумы между начальными приближениями корня), то появится сообщение (отсутствует сходимость). Причину ошибки можно выяснить, исследуя график f(x). Он поможет выяснить наличие корней уравнения f(x) = 0 и, если они есть, то определить приблизительно их значения. Чем точнее выбрано начальное приближение корня, тем быстрее будет сходиться функция root.

Если увеличить значение системной переменной TOL (tolerance), то функция root будет сходиться быстрее, но ответ будет менее точен, а при уменьшении TOL более медленная сходимость обеспечивает более высокую точность, соответственно. Последнее необходимо, если требуется различить два близко расположенных корня, или же, если функция f(x) имеет малый наклон около искомого корня, поскольку итерационный процесс в этом случае может сходиться к результату, отстоящему от корня достаточно

далеко. В последнем случае альтернативой повышения точности является замена уравнения f(x) = 0 на g(x) = 0.

Для выражения f(x) с известным корнем a нахождение дополнительных корней f(x) эквивалентно поиску корней уравнения h(x)=f(x)/(x-a). Проще искать корень выражения h(x), чем пробовать искать другой корень уравнения f(x)=0, выбирая различные начальные приближения. Подобный прием полезен для нахождения корней, расположенных близко друг к другу, он реализован в приведенном на рис. 1 и 2 фрагмента документа Mathcad:
Решение уравнений матричным способом.

Дано **A*X=B отсюда $X=A^{-1}B$**

В палитре инструментов выберите палитру матриц и введите значение матрицы. В рабочей области введите формулы, как показано на рис.1.

A				B
13.47	-2.29	3.29	4.75	2.32
2.75	11.11	2.28	-0.75	4.75
0.28	6.25	-9.21	0.79	2.25
3.21	2.21	0.49	7.87	-3.41

Рис 1.

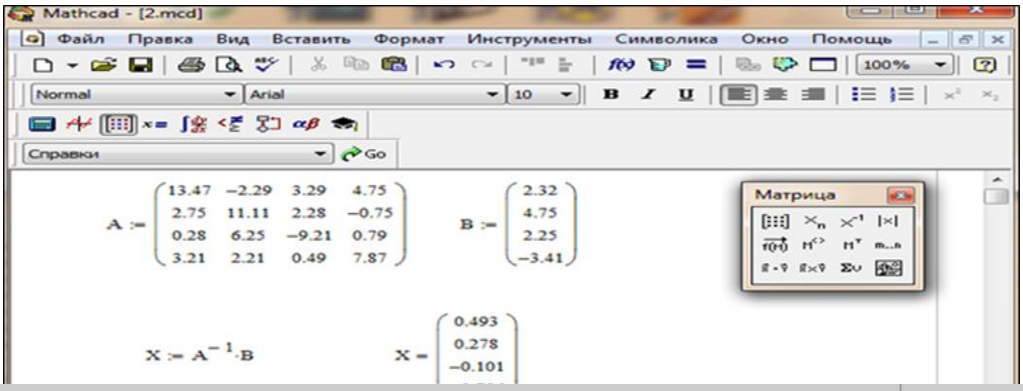


Рис 2.

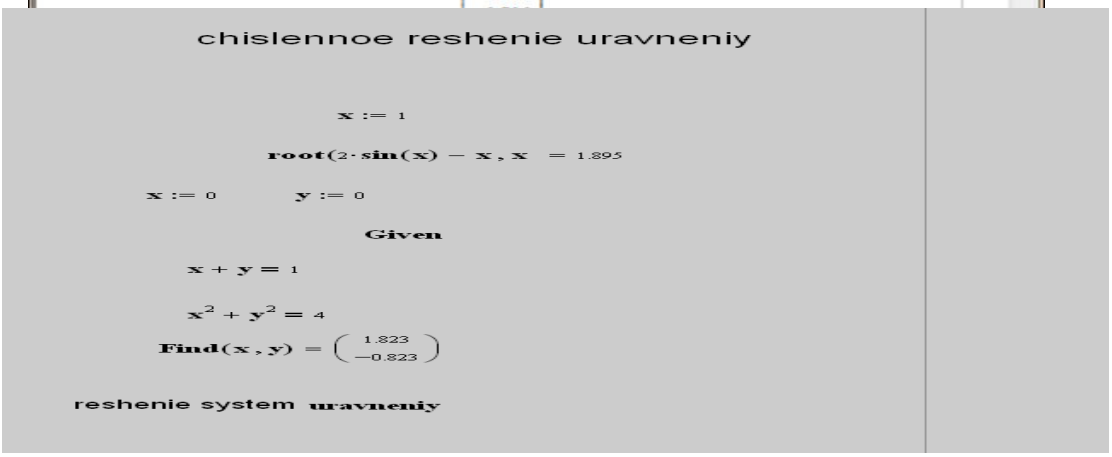


Рис 3. Применение функций Find и Root для реализации математических моделей

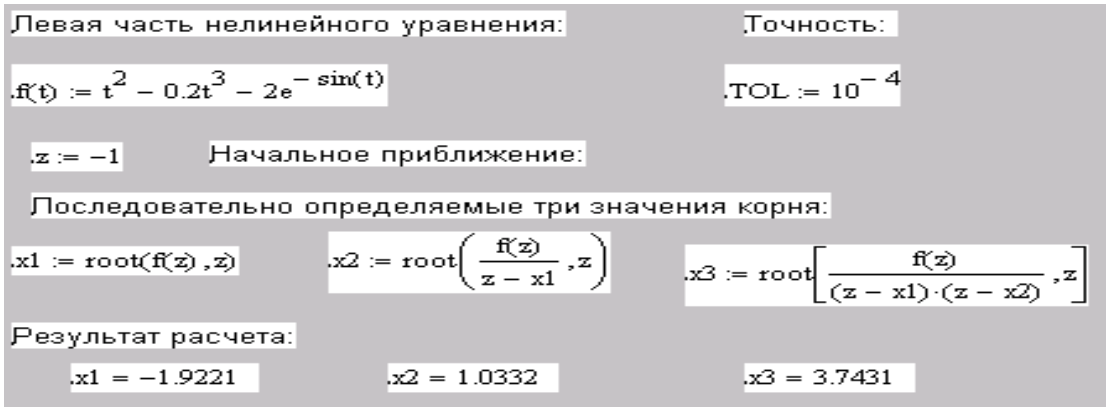


Рис.4 Применение функции root для решения нелинейных уравнений.

Альтернативным вариантом решения является изменение начального приближения представленное на фрагменте документа MathCad, который приведен на рис.5

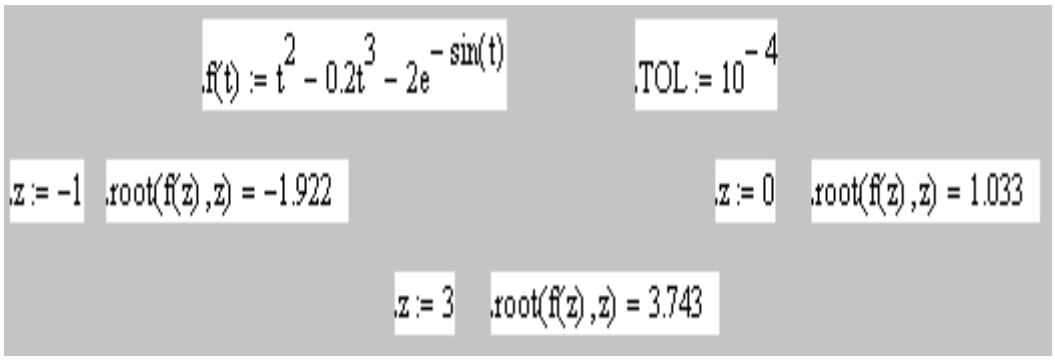


Рис.5 Альтернативный способ поиска корней с помощью функции

Блок решений с ключевыми словами (Given – Find или Given – Minerr) или функция solve позволяют найти решение произвольного нелинейного уравнения, если предварительно задано начальное приближение. Соответствующий фрагмент документа Mathcad приведен на рис.6.

Отметим, что между функциями Find и root наблюдается своеобразная конкуренция. С одной стороны, Find позволяет искать корни, как уравнений, так и систем. С этих позиций функция root как бы и не нужна. Но с другой стороны, конструкцию Given-Find невозможно вставить в Mathcad программы. Поэтому в программах приходится подстановками сводить систему к одному уравнению и использовать функцию root.

Практическая часть

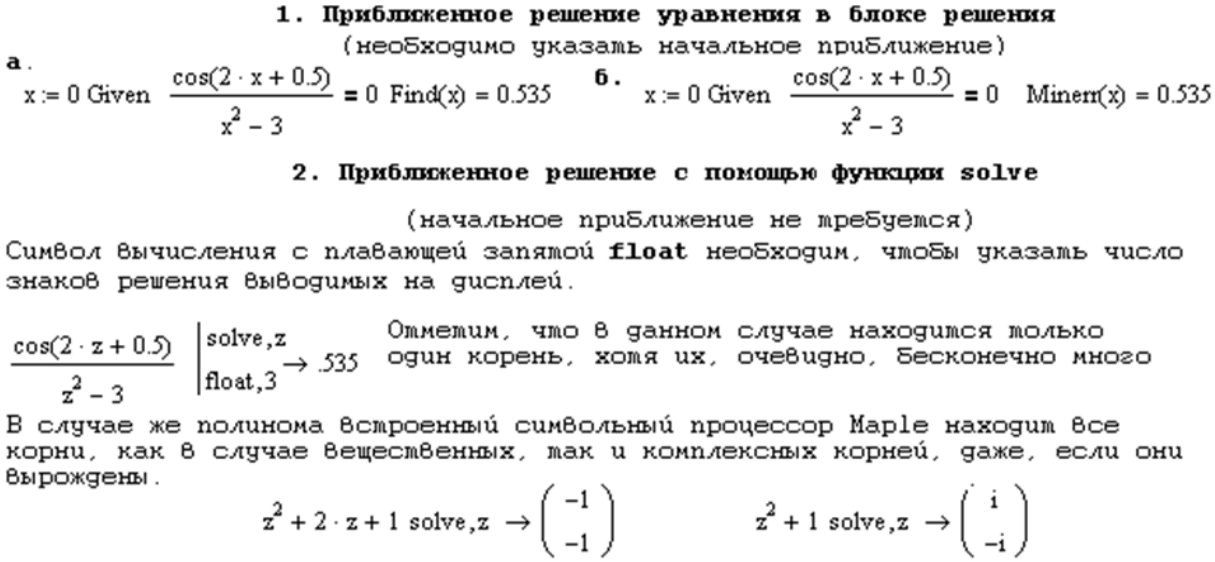


Рис.6. Применение функции solve и блока решений.

Mathcad освобождает пользователя от необходимости программирования алгоритма решения уравнений. Однако основной принцип работы в Mathcad – решение без программирования – имеет помимо очевидных достоинств и обратную сторону: неуверенность в результате вычислений. Эта неуверенность объясняется тем, что процесс решения скрыт от пользователя и не может быть проконтролирован непосредственно. Примеры вычислений с ошибочным результатом приведены ниже.

Зададим функцию, содержащую гиперболические синус и косинус:

$$f(x) = sh(x)/ch(x)^2$$

График этой функции в интервале –8 < x < 8 представлен на рис.7, которая представляет собой фрагмент соответствующего документа Mathcad. Корнем этой функции является x = 0. Слева и справа от этой точки f(x) имеет минимум и максимум, а при удалении от начала координат f(x) приближается к нулю. С формальной точки зрения решение этого уравнения не должно вызывать проблем, поскольку функция не содержит разрывов и имеет один корень во всей области определения неизвестного.

Причина ошибок кроется как в характере зависимости f(x), так и в особенностях работы процедуры, обеспечивающей решение. При начальном приближении x = -0,7 алгоритм root (в основу которого положен метод секущих, являющийся модификацией метода Ньютона) попадает на внешний, правый по отношению к x = 0, склон зависимости f(x) (см. рис.7) и "скатывается" по этому склону в поисках нуля f(x) в сторону +∞. Это видно по возвращаемым функцией **root** числам. Очевидно, что результаты решения неверны. Однако система не выдаёт никаких сообщений об ошибке. Это объясняется тем, что Mathcad считает корнем не то значение x, при котором

f(x) точно равна нулю, а то, при котором f(x) не превышает значения системной переменной TOL, равной по умолчанию 10-3. Данное условие во всех трёх случаях выполняется. С увеличением требований к точности расчёта (то есть с уменьшением TOL) возвращаемые root числа все больше отклоняются от корня x = 0, так как с ростом |x| функция f(x) приближается к нулю. Расчёты при различных начальных значениях x показывают, что границы области сходимости в рассматриваемой задаче примерно соответствуют условию |x| < 0,6.

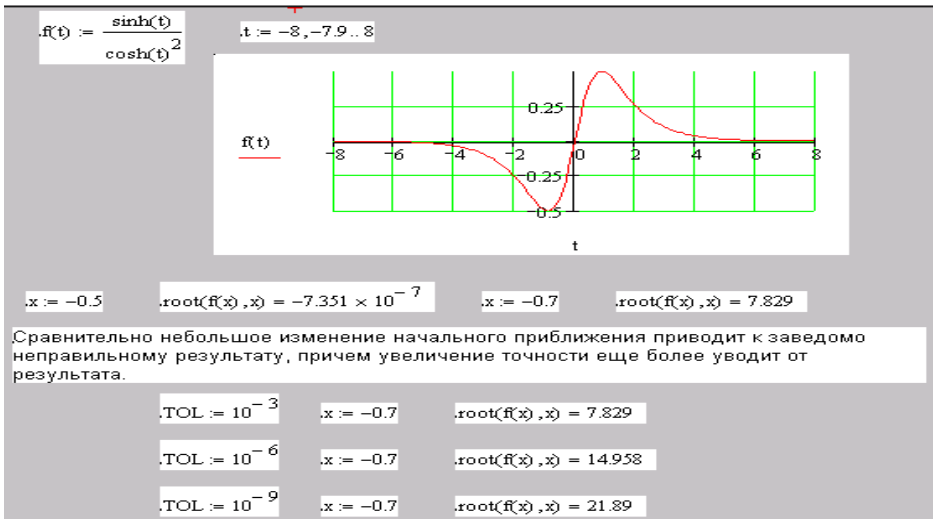


Рис.7 Пример неправильного действия функции root

К аналогичному решению приводит запись результата методом Ньютона, что представлено на рис.8.

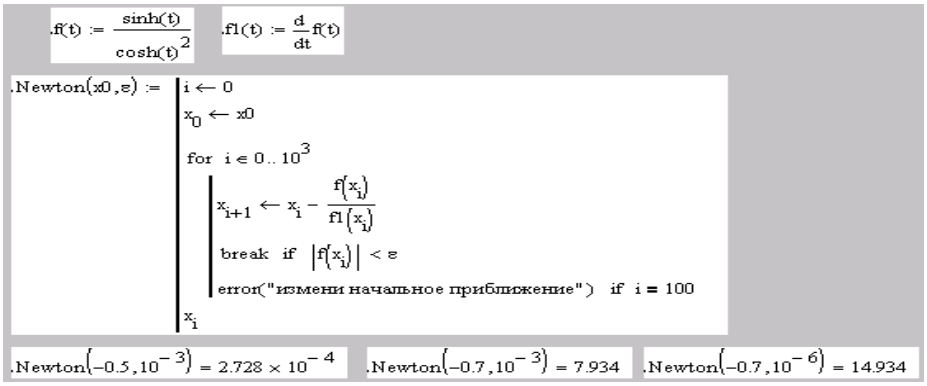


Рис. 8. Метод Ньютона решения нелинейного уравнения.

Для успешного решения уравнения необходимо правильно выбирать не только начальное приближение, но и критерий точности расчёта. Иллюстрацией этого служит пример решения модифицированного уравнения, отличающегося множителем 10-3:

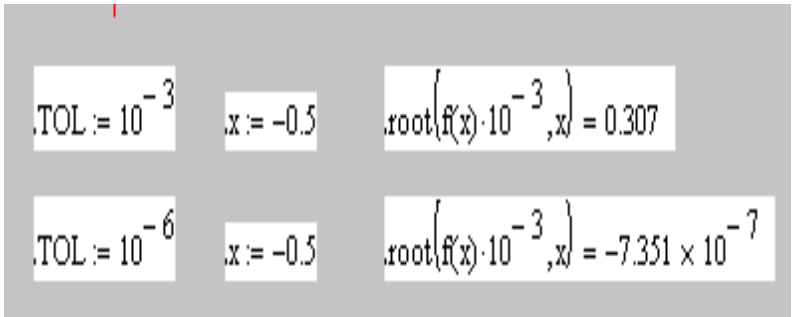


Рис.9.

Корни исходного уравнения f(x) = 0 и нового f(x)10-3 = 0 должны совпадать. Однако Mathcad выдаёт неверный результат. Эта ошибка объясняется тем, что функция f(x)10-3 при любых значениях x не превышает значения параметра TOL. Чтобы получить разумный результат, необходимо скорректировать требования к точности, выбрав, например, TOL = 10-6. В этом случае Mathcad возвращает правильный результат.

В ряде случаев особенности уравнения могут привести к неработоспособности алгоритма поиска корня. Так, на рис.10 приведен такой пример.

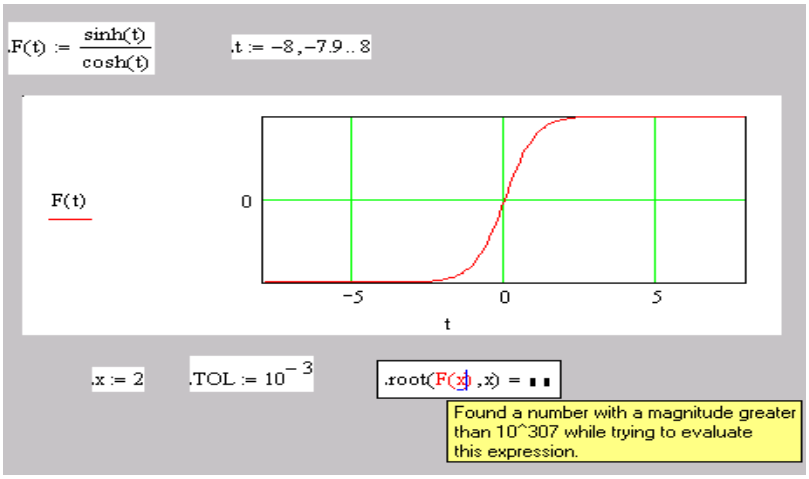


Рис. 10. Пример неработоспособности функции root

Неудача объясняется тем, что функция имеет пологие участки слева и справа от точки x = 0. Поскольку алгоритм root на каждом итерационном шаге делит значения функции f(x) на численный эквивалент её производной, возникает переполнение (overflow), так как производная при $|x| \geq 2$ близка к нулю.

Опасность ошибок, подобных рассмотренным выше, состоит в том, что они могут остаться незамеченными, поскольку MathCad не выдаёт никаких предупреждающих сообщений. Поэтому при решении уравнений желательно придерживаться следующих правил. Во-первых, необходимо сначала провести отделение корней уравнения. Во-вторых, желательно выполнить поиск решения несколько раз от различных начальных точек. Решение следует подвергать проверке, если его правильность не очевидна.

Символьное (аналитическое) решение уравнений в пакете MathCAD

Во многих случаях MathCAD позволяет найти аналитическое решение уравнения. Для того чтобы найти решение уравнения в аналитическом виде

необходимо записать выражение и выделить в нем переменную. После этого выбираем из пункта меню Symbolic (Символы) подпункт Variables (Переменные) и Solve (Вычислить). Другими вариантами нахождения решения в символьной форме являются – использование функции solve из палитры математических операций использование блока решения (с ключевыми словами Given - Find). Данные методы проиллюстрированы на рис.9, который представляет фрагмент соответствующего документа пакета Mathcad.

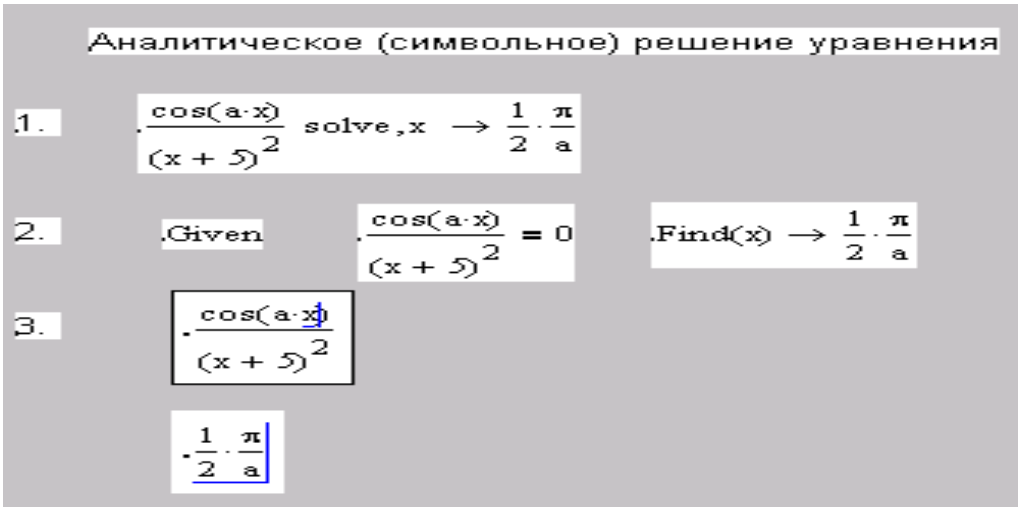


Рис 11.

Задания:

1. Решение нелинейных уравнений **(CTRL+)**

Given

2. $z^3 + 3\cdot z^2 + 2\cdot z - 6 = 0$

$\text{Find}(z) \rightarrow \left(1 \quad -2 + \sqrt{2}\cdot i \quad -2 - \sqrt{2}\cdot i\right)$

$x := 0$

$\text{root}(2 - \sin(x) - x, x) = 1.106$

2. Решение системы линейных уравнений матричный способ

$A := \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{pmatrix} \qquad B := \begin{pmatrix} 10 \\ 20 \\ 50 \end{pmatrix}$

$x := A^{-1} \cdot B \qquad x = \begin{pmatrix} 17.5 \\ -25 \\ 12.5 \end{pmatrix}$

3. Решение системы нелинейных уравнений

Применяем комбинацию клавиш **CTRL+** для всех

уравнений системы

▮ = ▮

▮ = ▮

4. Решение дифференциальных уравнений.

$\frac{d▮}{d▮}▮ = ▮$

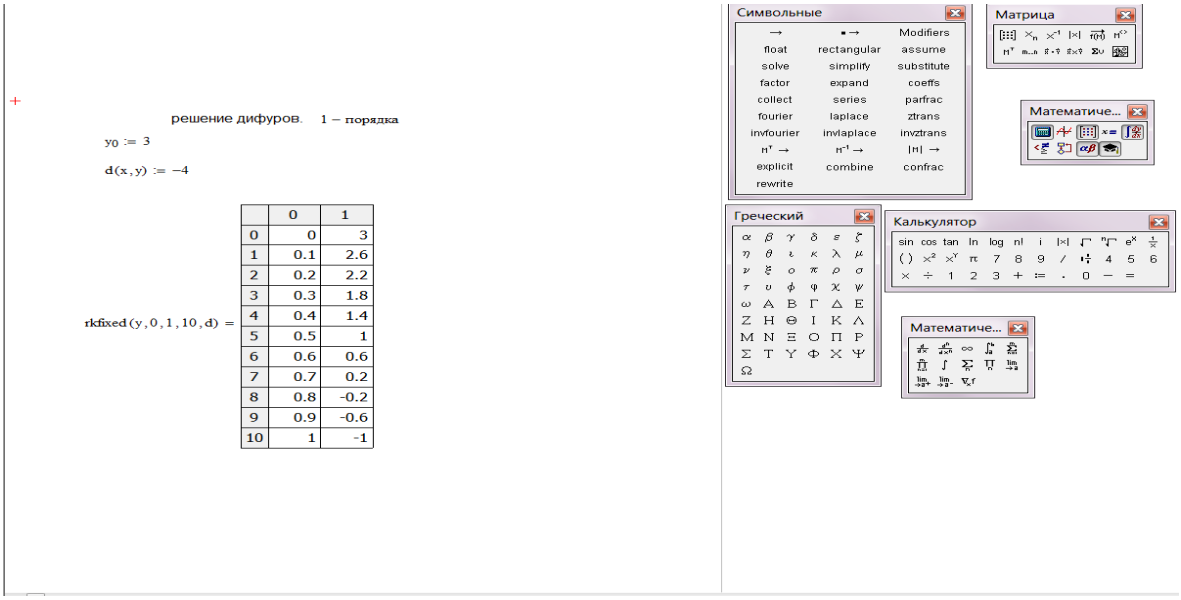
Примените методику решения для своего варианта.

1. Решение нелинейных уравнение.

1) 10sinx-x2=0
2) x^3-2x+2=0
3) 2^x-3cosx+1= 0
4) x^4-x3-2x+1= 0
5) Cos(2x+1)-3Sinx=0
6) x^3-Cos(x+0,5) +1=0
7) arctgx+ex+x=0
8) 3x3arctgx-1=0
9) x-3cos21,04^x=0
10) e^x+x^3-2=0
11) xsinx-3cosx+1=0
12) x3+3x-1=0
13) 3x-1-2sinx-4=0
14) x4+x3-2x+1=0
15) 5Sinx-xSinx=0
16) sin(x+p/2)-8cosx=0
17) arctg(ex+1)-sinx=0
18) 2x-arctg(x-1)=0
19) -cos0,387x=0
20) ex-2(x-1)2=0
21) lg(x+1)-2x+3x=0
22) x3-cosx=0
23) x2-10xlnx=0
24) x3-0.5x2-x+3=0
25) 2Cosx-xSinx=0
26) arcsinx+0,5x-1=0
27) 2x2+arcsinx+1=0
28) chx-2x-0,5=0
29) e-x+x2-2=0
30) 2x-2x2-1=0

3.Решение дифференциальных уравнений .

Решить дифференциальное уравнение 1-го порядка: **y′ + 4y =0**

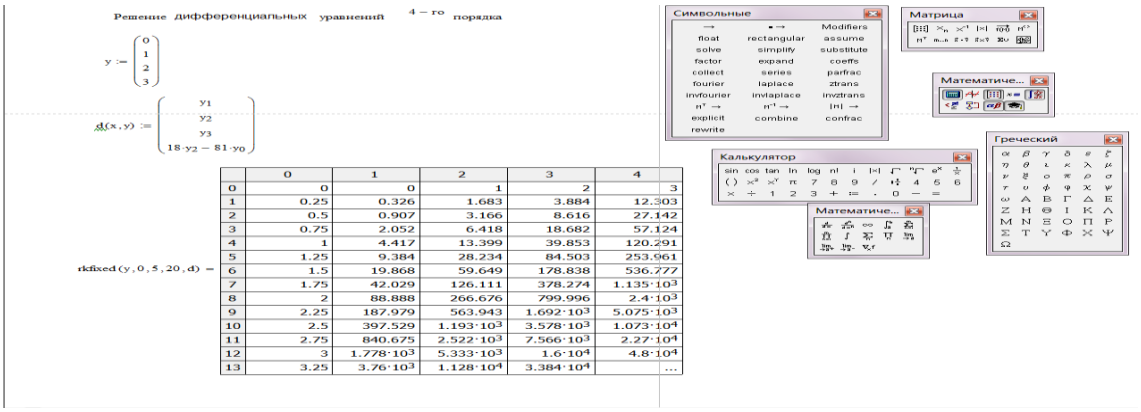


	0	1
0	0	3
1	0.1	2.6
2	0.2	2.2
3	0.3	1.8
4	0.4	1.4
5	0.5	1
6	0.6	0.6
7	0.7	0.2
8	0.8	-0.2
9	0.9	-0.6
10	1	-1

Рис.12

2-упражнение.

Решить дифференциольное уравнение 4-го порядка : **y′′′′-18y′′ +81y=0**



	0	1	2	3	4
0	0	0	1	2	3
1	0.25	0.326	1.683	3.884	12.303
2	0.5	0.907	3.166	8.616	27.142
3	0.75	2.052	6.418	18.682	57.124
4	1	4.417	13.399	39.853	120.291
5	1.25	9.384	28.234	84.503	253.961
6	1.5	19.868	59.649	178.838	536.777
7	1.75	42.029	126.111	378.274	$1.135 \cdot 10^3$
8	2	88.888	266.676	799.996	$2.4 \cdot 10^3$
9	2.25	187.979	563.943	$1.692 \cdot 10^3$	$5.075 \cdot 10^3$
10	2.5	397.529	$1.193 \cdot 10^3$	$3.578 \cdot 10^3$	$1.073 \cdot 10^4$
11	2.75	840.675	$2.522 \cdot 10^3$	$7.566 \cdot 10^3$	$2.27 \cdot 10^4$
12	3	$1.778 \cdot 10^3$	$5.333 \cdot 10^3$	$1.6 \cdot 10^4$	$4.8 \cdot 10^4$
13	3.25	$3.76 \cdot 10^3$	$1.128 \cdot 10^4$	$3.384 \cdot 10^4$	...

Рис.13

Задание: Решить свой вариант дифференциального уравнения 2-го порядка

Варианты для решения дифференциальных уравнений.

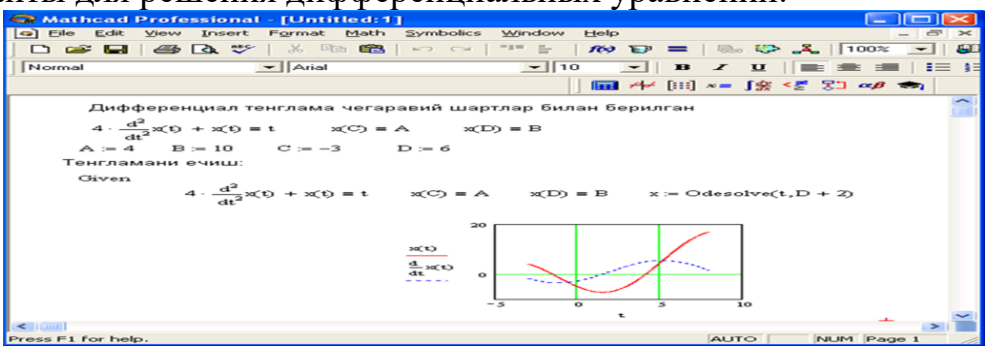


Рис.14

№	Дифференциальное уравнение	Начальные значения	Интервал	Шаг	Точное решение
1	$y'' + y = \frac{1}{\cos x}$	y(0)=1 $y'(0)=0$	[0; 0,5]	0,1	$\cos x + x \sin x + (\cos x) \ln \cos x$
2	$(1+x^2)y'' + (y')^2 + 1 = 0$	y(0)=1 $y'(0)=1$	[0; 0,5]	0,05	$1 - x + 2\ln(1+x)$
3	$y'' + 2y' + 2y = 2e^{-x}\cos x$	y(0)=1 $y'(0)=0$	[0; 0,5]	0,05	$e^{-x}(\cos x + \sin x + x \sin x)$
4	$y'' + 4y = e^{3x}(13x - 7)$	y(0)=0 $y'(0)=-4$	[0; 0,2]	0,02	$\cos 2x - \sin 2x + e^{3x}(x-1)$
5	$y'' + 4y' + 4y = 0$	y(0)=1 $y'(0)=-1$	[0; 1]	0,1	$(1+x)e^{-2x}$
6	$y'' - y = \sin x + \cos 2x$	y(0)=1,8 $y'(0)=-0,5$	[0; 2]	0,2	$e^x + e^{-x} - \frac{1}{2}\sin x - \frac{1}{5}\cos 2x$

7	$y''-3y'=e^{5x}$	y(0)=2,2 $y'(0)$=0,8	[0; 0,2]	0,02	$2+0{,}1(e^{3x}+e^{5x})$
8	$y''+4y=\cos 3x$	y(0)=0,8 $y'(0)$=2	[0; 1]	0,1	$\cos 2x+\sin 2x-0{,}2\cos 3x$
9	$y''-y'-6y=2e^{4x}$	y(0)=1,433 y`(0)=-0,367	[0; 1]	0,1	$0{,}1e^{3x}+e^{-2x}+\frac{1}{3}e^{4x}$
10	$y''-2y'+y=5xe^{x}$	y(0)=1 $y'(0)$=2	[0; 1]	0,1	$e^{x}+xe^{x}+5e^{x}\frac{x^3}{6}$
11	$y''+y'-6y=3x^2-x-1$	y(0)=-0,9 $y'(0)$=3,2	[0; 1]	0,1	$0{,}1e^{2x}-e^{-3x}-0{,}5x^2$
12	$8y''+2y'-3y=x+5$	y(0)=1/9 $y'(0)$=-7/12	[0; 1]	0,1	$e^{\frac{x}{2}}+e^{-\frac{3x}{4}}-\frac{x}{3}-\frac{17}{9}$
13	$x^2y''-2y=0$	y(1)=5/6 $y'(1)$=2/3	[1; 2]	0,1	$\frac{1}{2}x^2+\frac{1}{3x}$
14	$y''-4y'+5y=3x$	y(0)=1,48 $y'(0)$=3,6	[0; 0,5]	0,05	$e^{2x}(\cos x+\sin x)+\frac{3}{5}x+\frac{12}{25}$
15	$y''-5y'+6y=e^{x}$	y(0)=0 $y'(0)$=0	[0; 0,2]	0,02	$-e^{2x}+0{,}5e^{3x}+0{,}5e^{x}$
16	$y''-3y'+2y=x^2+3x$	y(0)=5,1 $y'(0)$=4,2	[0; 1]	0,1	$e^{x}+0{,}1e^{2x}+\frac{x^2}{2}+3x+4$
17	$y''+y=1+e^{x}$	y(0)=2,5 $y'(0)$=1,5	[0; 1]	0,1	$\cos x+\sin x+1+\frac{1}{2}e^{x}$
18	$y''+\frac{1}{x}y'-\frac{1}{x^2}y=8x$	y(1)=4 $y'(1)$=4	[1; 1,5]	0,05	$2x+\frac{1}{x}+x^3$
19	$x^2y''+xy'=0$	y(1)=5 $y'(1)$=-1	[1; 1,5]	0,05	$5-\ln x$
20	$y''-2y'+y=xe^{x}$	y(0)=1 $y'(0)$=2	[0; 0,5]	0,05	$(1+x)e^{x}+x^3e^{x}/6$
21	$y''-3y'+2y=2\sin x$	y(0)=2,6 $y'(0)$=3,2	[0; 1]	0,1	$e^{x}+e^{2x}+0{,}2\sin x+0{,}6\cos x$
22	$x^2y''+2{,}5y'x-y=0$	y(1)=2 $y'(1)$=3,5	[1; 2]	0,1	$3\sqrt{x}-x^{-2}$
23	$4xy''+2y'+y=0$	y(1)=1,3817 $y'(1)$=-0,15	[1; 2]	0,1	$\sin\sqrt{x}+\cos\sqrt{x}$

24	$x^2y'' - 4xy' + 6y = 2$	y(1)=1,433 $y'(1)$=2,3	[1; 2]	0,1	$x^2 + 0{,}1x^3 + \frac{1}{3}$
25	$y'' - y = e^{2x}(x-1)$	y(0)=11/9 $y'(0)$=-11/9	[0; 1]	0,1	$e^x + e^{-x} + e^{2x}(\frac{1}{3}x - \frac{7}{9})$

Контрольные вопросы:

1. Какие математические модели реализуются в Mathcad?
2. В чем различны численные и аналитические методы?
3. Какие операторы используются для решения задач технической системы?
4. В чем заключается визуализация решений заданных моделей.

Лабораторная работа №2
Создание и форматирование трехмерного графика в системе программирования Matlab.

Цель работы: Научиться созданию и форматированию графиков в системе MATLAB.

Порядок выполнения работы:

1. Ознакомиться с теоретической частью.
2. Изучить и выполнить работу создания графиков.
3. Изучить и выполнить работу форматирования графиков.
4. Выполнить анимационный график в MATLAB и сдать работу.

MATLAB имеет исключительно мощную систему для построения различных двухмерных и трехмерных графиков, а также их настройки, редактирования и форматирования. Типы и подтипы графиков MATLAB очень разнообразны. Список функций двумерного графика можно получить командой *help graph2d*, трехмерной –*help graph3d*.

Графики выводятся в отдельных графических окнах с помощью команды вида *figure(n)*, где *n* – номер графического окна. На одном графике можно построить несколько кривых, отличающихся цветом и типами линий и точек. Графики могут быть скопированы и вставлены в другие приложения:Word,Excel,PowerPoint и др. Для этого используется команда *Edit/ CopyFigure* окна графика.

Графическая система MatLAB, которая включает в себя команды высокого уровня для визуализации двух и трехмерных данных, обработки изображений, анимации и иллюстрированного графика. Она также включает в себя команды низкого уровня, позволяющие полностью редактировать внешний вид графика также как при создании Графического пользовательского интерфейса (GUI) для MatLab приложений.

Для визуализации вычислений в MATLAB широко используется машинный график. График в MATLAB имеется двух типов:

• обычная двумерный и трехмерный растровый график;
• специальный дескрипторный (handle) график.

Пока мы остановимся на обычном графике. С ним связано представление о графических объектах, имеющих определенные свойства. В большинстве случаев об объектах можно забыть, если только вы не занимаетесь объектно-ориентированным программированием задач графика. Связано это с тем, что большинство команд высокоуровневого графика, ориентированного на конечного пользователя, автоматически устанавливает свойства графических объектов и обеспечивает воспроизведение графика в нужной системе координат, палитре цветов, масштабе и т. д. Применение графика MATLAB практически исключает необходимость в сложных математических вычислениях, обычно необходимых для построения графиков. Средства графика в новых версиях MATLAB существенно дополнены. Новая позиция Graphics меню содержит три команды:

• New Figure – открывает пустое окно графика;
• Plot Tools – открывает окно нового мощного редактора графика;
• More Plots… – открывает окно доступа к различным видам графика.

На более низком уровне решения задач используется ориентированный на опытного программиста дескрипторный графика (Handle Graphics), при котором каждому графическому объекту в соответствие ставится особое описание – дескриптор, на который возможны ссылки при использовании графического объекта. Дескрипторный график позволяет осуществлять визуальное программирование объектов пользовательского интерфейса – управляющих кнопок, текстовых панелей и т. д.

Часто используемые команды при построении графиков

plot(t,y) % График непрерывной функции y(t)
plot(x1, y1, x2, y2) % Графики зависимостей y1 от x1 и y2 от x1
stem(x,y) %График дискретной функции (сигнала)y(x)
stairs(x,y) % График в виде ступенчатой линии
loglog(f,Y) %График с логарифмическими масштабами по x и y
semilogx(f,Y) %Логарифмический масштаб по x и линейный по y
polar(phi,r) % График в полярных координатах
title(‘ название’) % Вывод заголовка графика
xlabel(‘время’) % Метка по оси x
ylabel(‘Напряжение’) % Метка по оси y
legend(‘АЧХ системы‘) % Вывод поясняющей надписи
axis([xmin, xmax, ymin, ymax]) % Установка масштабов по осям x и y
xlim([xmin,xmax]) % Установка масштаба по оси x
ylim([ymin,ymax]) % Установка масштаба по оси y
figure(*n*) % Устанавливает фигуру (окно)*n* активной
subplot(*r,c,n*) % Разбивает графическое окно на*r* * *c* подокон и subplot(*rcn*) % устанавливает подокно *n* в качестве активного.
gridon% к графику добавляется сетка
holdon% позволяет построить несколько графиков в окне

holdoff% отменяет holdon для текущего графика
text% позволяет разместить текст на графике
zoomon/off% включение / выключение возможности увеличения % фрагментов графика с использованием % левой и правой кнопок мыши
Простые примеры:
>> x=0:0.01:2*pi;
>> y=sin(x);
Построение графика зависимости функции *y* от индекса массива (номера элемента)*x*
>> plot(y)

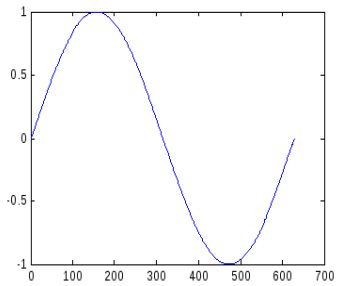


Рис.1.

Построение графика зависимости y(x)
>> plot(x,y)

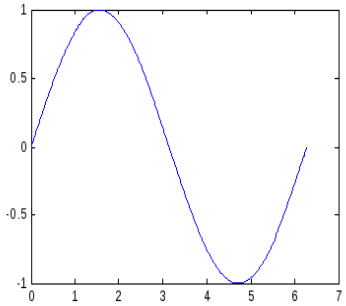


Рис.2.

Несколько пар аргументов в функции *plot()* позволяют построить несколько графиков в одном графическом окне. При этом MATLAB для каждого графика использует отдельный цвет линии.
Пример.
>> x = 0:pi/100:2*pi;
>> y = sin(x);
>>y2 = sin(x-.5);
>>y3 = sin(x+.5);
>>plot(x,y,x,y2,x,y3)
>> legend('sin(x)','sin(x-.5)','sin(x+.5)')

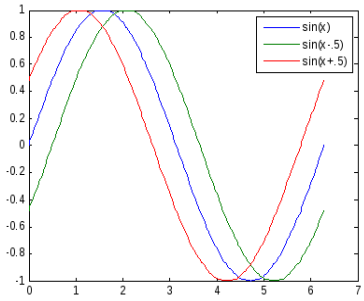


Рис.3

Цвет, тип линии и обозначение (тип) точек являются аргументами функции *plot*, соответствующие справочные сведения можно получить с помощью команды вызова справки *help plot* .

Для разбиения графического окна на подокна служит команда *plot(m,n,p)* или *plot(mnp)*,в которой *m*– число строк, *n*- число столбцов, *p*- номер подокна.

Пример построения графика функции $x(t) = t\,e^{-t}\cos(2\pi 4t)$ в двух подокнах с помощью функции *plot()* в одном случае и функции *stem()*в другом с разными пределами по оси аргумента (рис. 15):

```
t=linspace(0, 8, 401); % вычисление 402 точек в интервале [0,8]
x = t.*exp(-t).*cos(2*pi*4*t);
figure
subplot(2,1,1)
plot(t,x)
subplot(2,1,2)
stem(t,x)
axis([0 1 min(x) max(x)] )
```

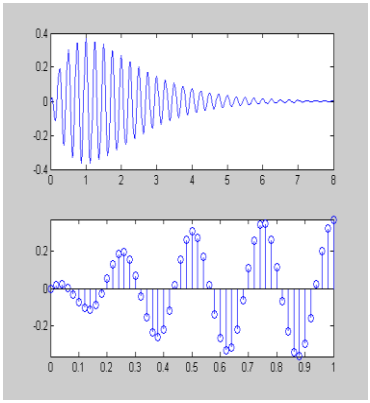


Рис. 4

Другой пример

```
Fs=1024; % Частота отсчетов
f1=50; % частота гармоники
f2=60;
N=512; % число отсчетов сигнала
n=0:(N-1);
t=0:1/Fs:(N-1)/Fs; % вектор времени
% генерирование сигнала
x=cos(2*pi*f1*t)+0.5*cos(2*pi*f2*t)+randn(1,length(t));
```

```
plot(t,x), grid % график сигнала
title('Сигнал')
xlabel('Время, с')
```

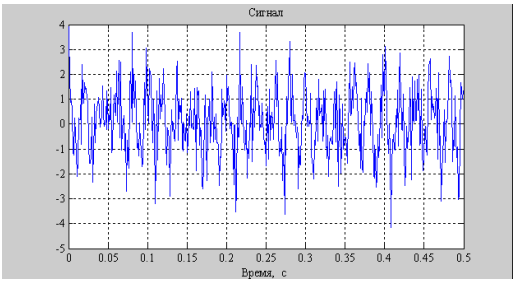


Рис. 5

Для добавления графиков к уже существующим применяют команду *hold on*

```
t=0:0.01:1;
x1=sin(2*pi*4*t);
x2=cos(2*pi*4*t);
plot(t,x1,'r')
hold on
plot(t,x2,'g')
plot(t,x1+x2, '--b')
legend('x1=sin(2*pi*4*t)', 'x2=cos(2*pi*4*t)','x1+x2')
```

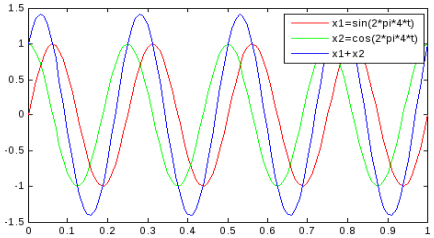


Рис. 6

Для отмены действия *hold on* (освобождения окна графики) используют *hold off*.

Пример построения графика в полярной системе координат

```
>> t=0:pi/100:2*pi;
>> polar(t,cos(6*t))
```

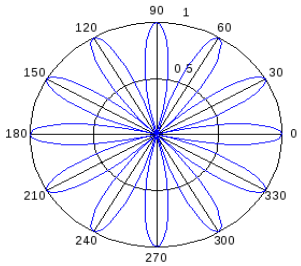


Рис. 7

В окне графики MATLAB позволяют выполнять разнообразную настройку графического окна и его объектов с помощью меню или панели инструментов (рис.21).

В окне редактора или с помощью контекстного меню по правой кнопке мыши производятся необходимые установки (цвет, размер, тип, толщина линии и др.) объекта окна графики.
Возможности для подобной интерактивной настройки графики - очень широкие. В первую очередь они обеспечиваются кнопкой Edit Plot инструментальной панели окна.

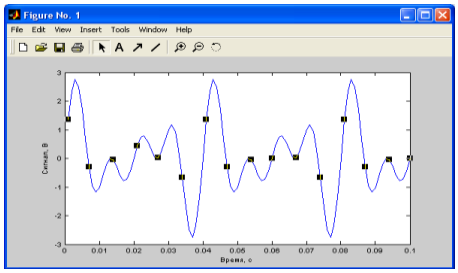


Рис. 8

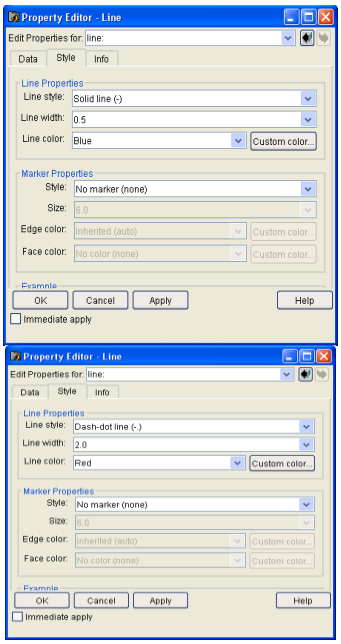


Рис. 9

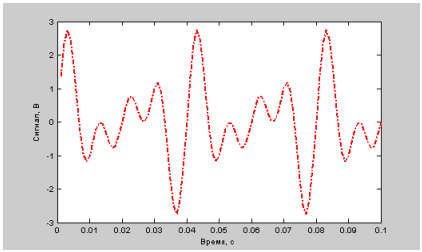


Рис.10

Трехмерный график MATLAB– очень развитый и многообразный, сам по себе очень важная часть программы, но в курсе «Сигналы и системы» она используется редко.
Некоторые из команд построения 3D– графиков
>> plot3(…) % строит аксонометрическое изображение 3D-поверхности
>> mesh(…) % строит трехмерные поверхности со специфицированной
% окраской
Пример.
>> [X,Y]=meshgrid([-3:0.1:3]);

```
>> Z=X.^2+Y.^2;
>> mesh(X,Y,Z)
```

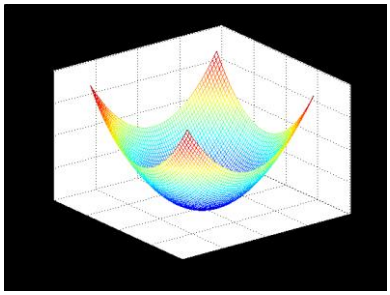


Рис.11.

Пример построения графика передаточной функции системы второго порядка с передаточной функцией $H(s)=\frac{s}{(s+1)^2+1}=\frac{s}{s^2+2s+2}$.

Нули и полюса системы : $z=0, p_1=-1+j, p_2=-1-j$

```
[x,y]=meshgrid(-2:0.01:1, -2:0.01:2);
s=x+y*j;
H=(s+0)./((s+1).^2+1);
mesh(x,y,abs(H))
```

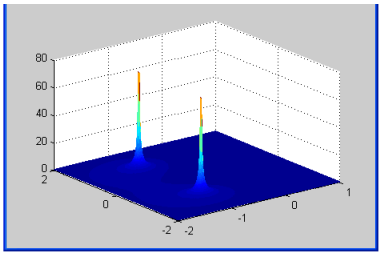


Рис 12.

Форматирование двумерных графиков.

Для вывода окна форматирования двумерного графика достаточно поместить указатель мыши в область графика и дважды щелкнуть левой кнопкой мыши. В окне документа появится окно форматирования. Оно имеет ряд вкладок. Вкладка становится активной, если установить на ее имя указатель мыши и щелкнуть левой кнопкой.

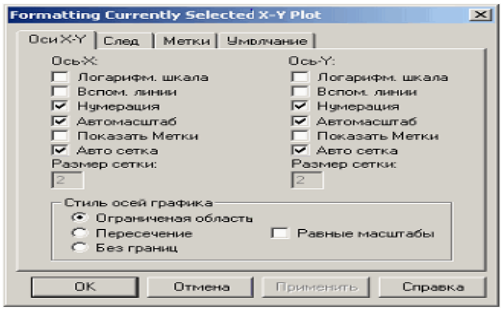


Рис 13.

Как видно на рисунке окно форматирования имеет четыре вкладки:

- оси X-У- задание параметров форматирования осей;
- линии графика – задание параметров форматирования линий графика;
- надписи – задание параметров форматирования меток осей;

- по умолчанию – назначение установленных параметров форматирования параметрами по умолчанию.

1. Форматирование осей графика.

На вкладке X-У оси содержатся следующие основные параметры, относящиеся к осям X и У (Axis X и Axis У):

- Логарифмический масштаб – установление логарифмического масштаба;
- Линии сетки – установка линий масштабной сетки;
- Пронумеровать – установка цифровых данных по осям;
- Автомасштаб – автоматическое масштабирование графика;
- Нанести риски – установка делений по осям;
- Автосетка – автоматическая установка масштабных линий;
- Число интервалов – установка заданного числа масштабных линий.

Группа Стиль осей позволяет задать стиль отображения координатных осей:

- Рамка – оси в виде прямоугольника;
- Визир – оси в виде креста;
- Ничего – отсутствие осей;
- Равные деления – установка одинакового масштаба по осям графика.
- **2. Форматирование линий графиков.**

Эта вкладка служит для управления отображением линий, из которых строится график. На этой вкладке представлены следующие параметры:

- Метка легенды – выбор типа линии в легенде;
- Символ – выбор символа, который помещается на линию, для отметки базовых точек графика;
- Линия – установка типа линии;
- Цвет – установка цвета линии и базовых точек;
- Тип – установка типа графика;
- Толщина – установка толщины линии.

Узловые точки (точки, для которых вычисляются координаты) графиков часто требуется выделить какой-нибудь фигурой. Список столбца Symbol позволяет выбрать следующие отметки для базовых точек графика каждой из функций:

- ничего – без отметки;
- x’s – наклонный крестик;
- +’x – прямой крестик;
- квадрат – квадрат;
- ромб – ромб;
- o’s – окружность.

Список в столбце Линия позволяет выбрать типы линий: непрерывная, пунктирная, штрих-пунктирная.
Раскрывающейся список столбца Type позволяет выбрать следующие типы линий графика:

- линия – построение линиями;
- точки – построение точками;
- интервалы – построение вертикальными черточками с оценкой интервала погрешностей;
- столбец – построение в виде столбцов гистограммы;
- ступенька – построение ступенчатой линией;
- протяжка – построение протяжкой от точки до точки.

3. Задание надписей на графиках.

Эта вкладка позволяет вводить в график дополнительные надписи. Для установки надписей служат поля ввода:

- Заголовок – установка титульной надписи к рисунку;
- Ось X – установка надписи по оси X;
- Ось Y – установка надписи по оси У.

В группе Заголовок имеются переключатели сверху и снизу для установки титульной надписи либо над графиком, либо под ним.

4. Параметры графиков по умолчанию.

Вкладка "По умолчанию" позволяет назначить установленные на других вкладках параметры форматирования параметрами по умолчанию. Для этого служит флажок установки "использовать по умолчанию". Щелкнув на кнопке "вернуть значения по умолчанию" можно вернуть стандартные параметры графика.

*Постройте график функции p(x)=5*x^6-3, задав свой цвет и стиль кривой.*

А теперь рассмотрим, как на одном рисунке отобразить несколько графиков, например, y=2*cos(x), y=sin(x)^2 и y=x.

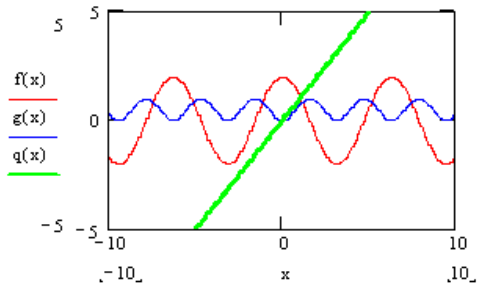


Рис.14

Алгоритм выглядит так:

1. $f(x) := 2 \cdot \cos(x)$ $g(x) := \sin(x)^2$ $q(x) := x$ ьте их, например f(x), g(x), q(x):

2. Вызвав шаблон графика, введите по оси X имя независимой переменной (или переменных, если их несколько), по оси Y введите f(x), поставьте запятую (при этом первое выражение уходит вверх, а под ним появляется место ввода), введите g(x), знак запятой и следующее выражение q(x).
3. Отведя указатель мыши за пределы графика, щелкните левой кнопкой мыши – появится график с тремя кривыми.

*Постройте на одном рисунке графики функций y=x^2+2*x, y=tg(x), y=x-5.*

После того, как мы освоили построение двумерных графиков одной или нескольких функций, рассмотрим построение графиков поверхностей (трехмерные или 3D-графики). С помощью системы MathCad такие графики строятся даже проще, чем двумерные.
Построим график функции z(x,y)=x^2 + y^2, для этого:

1. Задайте функцию двух переменных: $z(x,y) := x^2 + y^2$.
2. Используя палитру графики, введите шаблон трехмерного графика.
3. На единственное место ввода под шаблоном введите z.
4. Выведите курсор мыши за пределы графика и щелкните левой клавишей мыши – будет построен график в виде "проволочного каркаса".

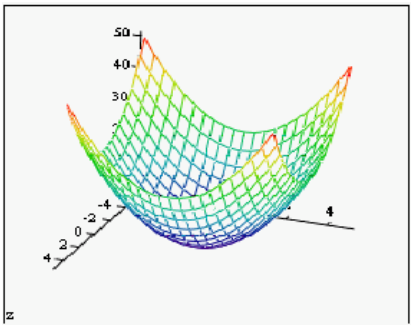
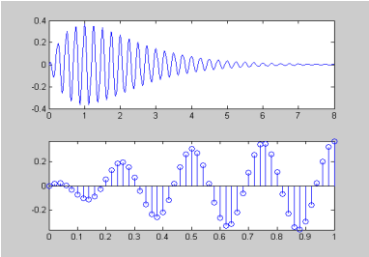


Рис.15.

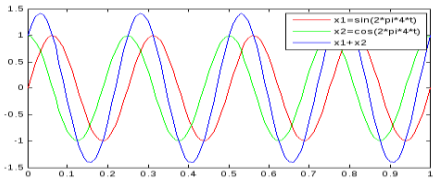
5.Постройте график функции z=cos(x)+sin(y).

```
x = t.*exp(-t).*cos(2*pi*4*t);
figure
subplot(2,1,1)
plot(t,x)
subplot(2,1,2)
stem(t,x)
axis([0 1 min(x) max(x)] )
```



Для добавления графиков к уже существующим применяют команду hold on

```
t=0:0.01:1;
x1=sin(2*pi*4*t);
x2=cos(2*pi*4*t);
plot(t,x1,'r')
hold on
plot(t,x2,'g')
plot(t,x1+x2, '--b')
legend('x1=sin(2*pi*4*t)', 'x2=cos(2*pi*4*t)','x1+x2')
```



В окне графики MATLAB позволяют выполнять разнообразную настройку графического окна и его объектов с помощью меню или панели инструментов (рис.9).

Рис.16.

Вариант 5.

Условия задачи: При помощи данного уравнения создать график эпициклоида и гипоциклоида на математических пакетах MathCAD и Matlab.

k=1;

k1=-1;

i=0.25;

α =0;

λ =1;

φ=0:π/360:2*π;

$$x = (1 + k * i) * \cos(\varphi) - \lambda * \cos(\frac{1 + k * i}{i} * \varphi - k * \alpha)$$

$$x_1 = (1 + k * i) * \sin(\varphi) - k * \lambda \sin(\frac{1 + k * i}{i} * \varphi - k * \alpha)$$

Текст программы на Matlab е:

```
k=1;
k1=-1;
i=0.25;
a=0;
l=1;
f=0:pi/360:2*pi;
x1=(1+k*i)*cos(f)-l*cos((((1+k*i)/i)*f)+k*a);
x2=(1+k*i)*sin(f)-k*l*sin((((1+k*i)/i)*f)+k*a);
x3=(1+k1*i)*cos(f)-l*cos((((1+k1*i)/i)*f)+k1*a);
x4=(1+k1*i)*sin(f)-k1*l*sin((((1+k1*i)/i)*f)+k1*a);
title('Работа студента Мирпулатова М.')
plot(x1,x2,x3,x4,sin(f),cos(f));
```

График.

Текст программы на MathCAD:

$$S(k,i,a,l,f) := \left| \begin{array}{l} x_0 \leftarrow (1+k\cdot i)\cdot\cos(f) - l\cdot\cos\left(\frac{1+k\cdot i}{i}\cdot f - k\cdot a\right) \\ x_1 \leftarrow (1+k\cdot i)\cdot\sin(f) - k\cdot l\cdot\sin\left(\frac{1+k\cdot i}{i}\cdot f - k\cdot a\right) \\ x \end{array} \right.$$

$$f := 0, \frac{\pi}{360} .. 2\cdot\pi$$

$$F(f) := S(1, 0.25, 0, 1, f)$$

$$G(f) := S(-1, 0.25, 0, 1, f)$$

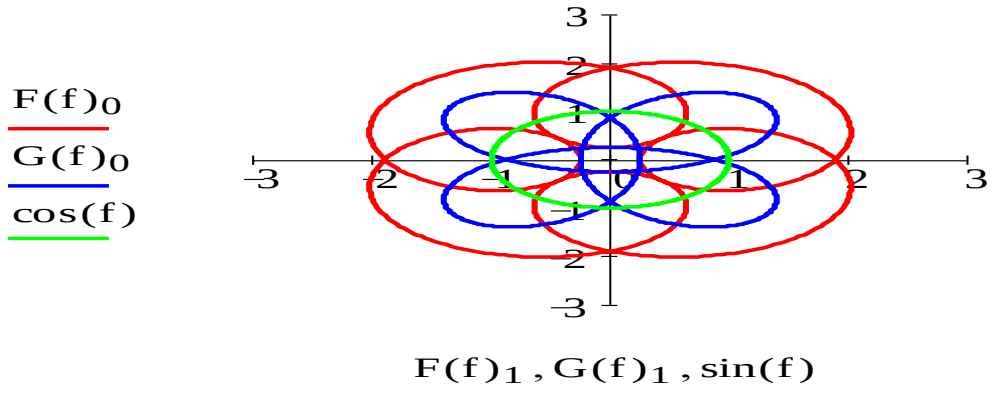


Рис.17

Лабораторная работа №3
Разработка и анализ имитационных моделей технических объектов в энергетике

Цель работы*:* Ознакомиться с программой MATLAB, изучить её основные возможности при решении задач моделирования.

Порядок выполнения работы:

1. Изучить теоретическую часть.
2. Ознакомиться с режимами работы системы.
3. Выбрать задачу по специальности для решения.
4. Произвести необходимые операции для решении данной задачи.

Теоретическая часть

Специализированная система Matlab. Модели Simulink.

При инсталляции MATLAB можно установить пакет визуального моделирования систем, процессов — Simulink. Чтобы создать новую модель, щёлкните мышкой по пиктограмме  или наберите команду **simulink** . Перед вами появятся два окна. В первом, можно из библиотеки выбрать модель, а во втором, как из кирпичиков, вы собираете общую модель системы:

Двойным щелчком, например по пиктограмме Source (Источники), вы откроете библиотеку источников. Открывая соответствующие библиотеки и перетягиванием мышкой выбранной модели в окно создания моделей системы вы можете набрать модель реальной системы. Далее, запустив модель, вы получите решение. В настройках можно выбрать время работы системы, метод решения и некоторые другие параметры.

«Matlab» является продуктом фирмы «The MathwoksInc» Первая версия пакета «Matlab» была разработана уже более 20 лет тому назад. Развитие и совершенствование этого пакета происходило одновременно с развитием средств вычислительной техники. Название пакета «Matlab» происходит от словосочетания MatrixLaboratory, он ориентирован, в первую очередь, на обработку массивов данных (матриц и векторов). Именно поэтому, несмотря на достаточно высокую скорость смены поколений вычислительной техники, «Matlab» успевал впитывать все наиболее ценное от каждого из них. Как следует из названия пакета, он ориентирован, в первую очередь, на обработку массивов данных (матриц и векторов). Это позволило его разработчикам существенно повысить эффективность процедур, работающих с указанными типами данных.

В результате к настоящему времени «Matlab» представляет собой богатейшую библиотеку функций моделей (более 800). Единственная проблема при работе, с ними заключается в умении быстро отыскать те модели и функции из них, которые нужны для решения поставленной задачи. Именно в сфере математического моделирования «Matlab» позволяет наиболее полно использовать все современные достижения компьютерных

технологий, в том числе средства визуализации и аудификации (озвучивания) данных, а также возможности обмена данными через Интернет. Кроме того, пользователь имеет возможность создавать средствами «Matlab» собственный графический интерфейс, отвечающий как его вкусам, так и требованиям решаемой задачи.

Для облегчения работы с пакетом специалистами различных областей науки и техники вся библиотека функций разбита на разделы. Те модели и функции из них, которые носят более общий характер, входят в состав основного ядра пакета «Matlab». Те же модели и функции, которые являются специфическими для конкретной области, включены в состав пакетов расширения (Toolboxes). Таким образом, «MatLab» - в первую очередь, это средство математического моделирования, обеспечивающее проведение исследований с точки зрения анализа и синтеза, практически во всех известных областях техники и науки. При этом структура пакета позволяет эффективно сочетать оба основных подхода к созданию модели: аналитический и имитационный.

SIMULINK

Особое место среди наборов инструментов занимает система визуального моделирования Simulink. В определенном смысле Simulink можно рассматривать как самостоятельный продукт фирмы MathWorks (который даже в некоторых случаях продается в «именной» упаковке). Однако он работает только при наличии ядра «Matlab» и использует многие функции, входящие в его состав.

Следует обратить внимание, что пакеты «MatLab», Simulink и их пакеты расширения (Toolboxes, Blocksets) постоянно развиваются и совершенствуются.

Simulink – это уникальный инструмент исследования, не требует от пользователя знания навыков программирования, позволяет приобрести мощнейшие ассоциации и уяснить аналогии между структурой математического описания объекта и структурой блок схемы, реализующей данную модель.

Simulink – интерактивный инструмент для моделирования, имитации и анализа динамических систем. Он дает возможность строить графические блок - диаграммы, имитировать динамические системы, исследовать их работоспособность систем и совершенствовать проекты.

Simulink – полностью интегрирован с «MatLab», обеспечивая немедленный доступ к широкому спектру инструментов анализа и проектирования

Сразу становятся ясными причинно-следственные отношения с цепочкой математических понятий: независимая переменная, дифференциальное уравнение, вынуждающая функция, начальные условия, решение дифференциального уравнения и с соответствующей цепочкой понятий: «аналоговое» моделирование – время моделирования, входной сигнал, структурная схема объекта, начальные условия на интеграторах,

выходной сигнал и т. д. При моделировании в среде «MatLab», Simulink - средств измерения, регулирования и систем управления у студентов, слушателей и исследователей прививаются:

-навыки выбора, наладки, эксплуатации более современных средств автоматизации;

- навыки использования современных технологий в процессе обучения;

- четкое определение уравнения статики и динамики изучаемого объекта;

- умение анализировать работу изучаемого объекта;

- умение синтезировать изучаемый объект с желаемой характеристикой.

Пакет «MatLab» располагает широким набором виртуальных элементов, модулей, функций, представленных в виде условных обозначений, которые обладают основными свойствами реальных простых физических элементов, а в совокупности простых устройств, модулей, агрегатов, преобразователей, приборов, регуляторов, систем контроля и регулирования и т.д.

Таким образом, «собрав» на экране монитора из соответствующих элементов требуемую виртуальную лабораторную работу, то - есть определенную структурную схему изучаемого объекта, можно выполнить ее полный анализ, изучить его в установившихся и переходных режимах и произвести его синтез с желаемыми характеристиками. При этом можно быть уверенным, что при корректной сборке схемы, (корректном выборе типовых преобразователей, установке необходимых коэффициентов типовых звеньев - элементов, правильном их конфигурировании между собой, с учетом их принципиальной схемы) и умелом проведении экспериментов, результаты исследований совпадут с результатами исследований в реальной схеме, а по точности превзойдет их. В этом - суть программных пакетов «MatLab», «Simulink» и их несомненные превосходства и достоинства.

Пакеты «MatLab», «Simulink» полезны, не только для студентов в процессе изучения различных предметов при всех формах обучения, но и инженерам, научным работникам, которые проектируют и испытывают новейшие устройства, ибо эти испытания можно провести на виртуальной модели. Это избавляет студентов и исследователей от необходимости строить физическую модель изучаемого устройства (выбирать необходимые устройства, их монтировать, заполнять необходимыми веществами, настраивать и эксплуатировать) и, следовательно, резко сокращает как материальные, так и временные затраты.

Исследование параметров конкретной системы(схемы устройства) с помощью процедур системы MatLab.

Расчёт параметров и исследование функциональных зависимостей средствами MathCad. Используются 2 режима работы в системе MatLab. Режим командного программирования и режим моделирования с использованием пакетов приложений.

Выберем в пакете Simulink блок "Matlab function" и в параметрах пропишем путь к нашему файлу с уравнениями (файл называется Mathfuncs_Diode). Входным параметром будет напряжение, которое измеряется вольтметром, а выходным -ток.

Всю систему поместим в один блок - Subsystem. Назовем наш блок с диодом DiodeModel. Создадим маску диода

Для тестирования модели диода соберем схему Блок XY Graph строит ВАХ диода, Multimeter - графики напряжения и тока на диоде и резисторе.

Рассмотрим реакцию системы с единичной ООС и апериодическим звеном в прямой цепи на единичное воздействие. Для её набора нам понадобятся библиотеки модели: Source|Step, Sinks|Scope, Linear|Transfer Fcn и Linear|Sum. Перетянув эти элементарные модели в окно редактирования Simulink, изменяем исходные параметры в соответствии с нашей системой (двойной щелчок мышью) и соединяем их.

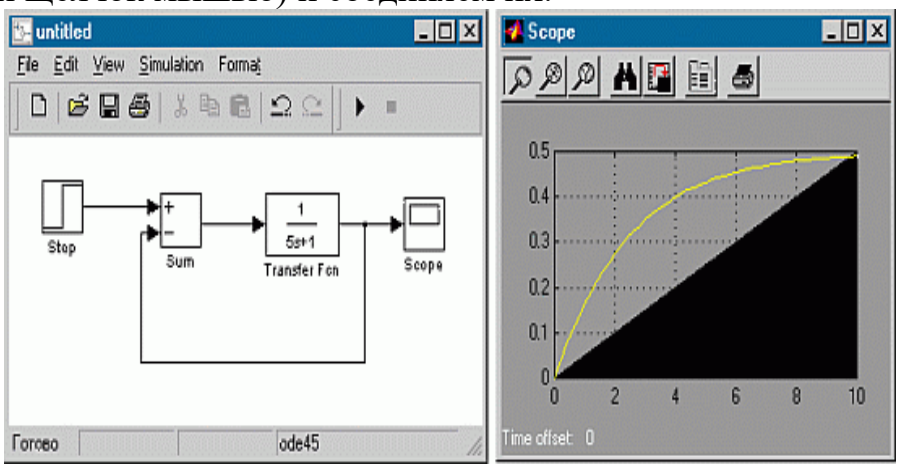


Рис.1

Чтобы можно было использовать полученное решение в сценарии Matlab нужно добавить компонент Connections|Out. Теперь вы имеете возможность работать с переменной. Данные, которые поступают на этот порт, доступны вам из среды Matlab. Чтобы запустить модель, вам достаточно набрать команду **sim** с необходимыми параметрами (наберите HELP SIM для подробной информации).

В последние годы в ВУЗах ряда стран при изучении студентами разных предметов в виде лекционных, лабораторных и практических занятий, при выполнении курсовых работ, проектов, выпускных работ и диссертаций все более широкое применение находит система моделирования и анализа с применением ряда программных комплексов. Среди программных комплексов пакет «Matlab» отличается своей простотой, компактностью, надежностью, расширенными возможностями, отвечающими современным требованиям, базирующимся на новых современных технологиях процесса обучения.

Особое место среди наборов инструментов занимает система визуального моделирования Simulink. В определенном смысле Simulink можно рассматривать как самостоятельный продукт фирмы MathWorks

(который даже в некоторых случаях продается в «именной» упаковке).
Однако он работает только при наличии ядра «Matlab» и использует многие функции, входящие в его состав.

Следует обратить внимание, что пакеты «MatLab», Simulink и их пакеты расширения (Toolboxes, Blocksets) постоянно развиваются и совершенствуются.

Simulink – это уникальный инструмент исследования, не требует от пользователя знания навыков программирования, позволяет приобрести мощнейшие ассоциации и уяснить аналогии между структурой математического описания объекта и структурой блок схемы, реализующей данную модель.

Simulink – интерактивный инструмент для моделирования, имитации и анализа динамических систем. Он дает возможность строить графические блок - диаграммы, имитировать динамические системы, исследовать их работоспособность систем и совершенствовать проекты.

Simulink – полностью интегрирован с «MatLab», обеспечивая немедленный доступ к широкому спектру инструментов анализа и проектирования

Сразу становятся ясными причинно-следственные отношения с цепочкой математических понятий: независимая переменная, дифференциальное уравнение, вынуждающая функция, начальные условия, решение дифференциального уравнения и с соответствующей цепочкой понятий: «аналоговое» моделирование – время моделирования, входной сигнал, структурная схема объекта, начальные условия на интеграторах, выходной сигнал и т. д. При моделировании в среде «MatLab», Simulink - средств измерения, регулирования и систем управления у студентов, слушателей и исследователей прививаются:

-навыки выбора, наладки, эксплуатации более современных средств автоматизации;

- навыки использования современных технологий в процессе обучения;

- четкое определение уравнения статики и динамики изучаемого объекта;

- умение анализировать работу изучаемого объекта;

- умение синтезировать изучаемый объект с желаемой характеристикой.

Пакет «MatLab» располагает широким набором виртуальных элементов, модулей, функций, представленных в виде условных обозначений, которые обладают основными свойствами реальных простых физических элементов, а в совокупности простых устройств, модулей, агрегатов, преобразователей, приборов, регуляторов, систем контроля и регулирования и т.д.

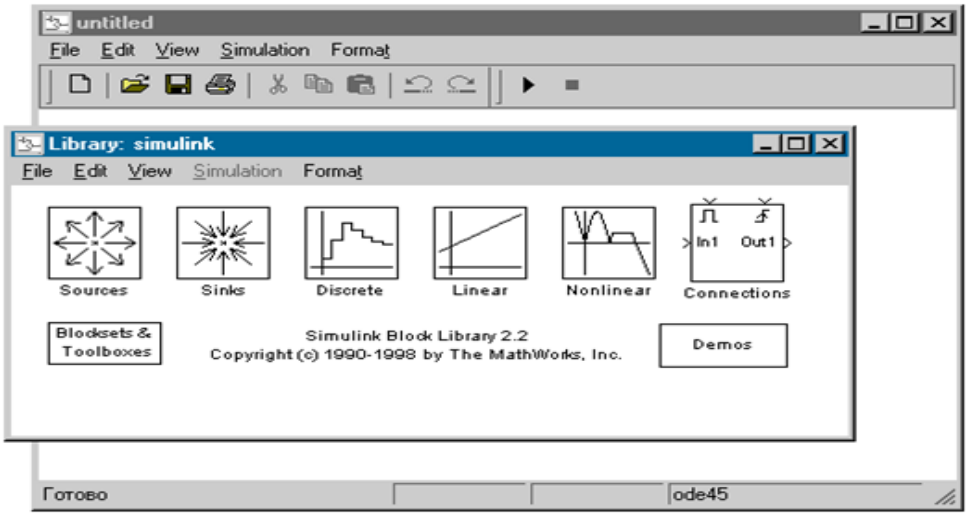


Рис 2.

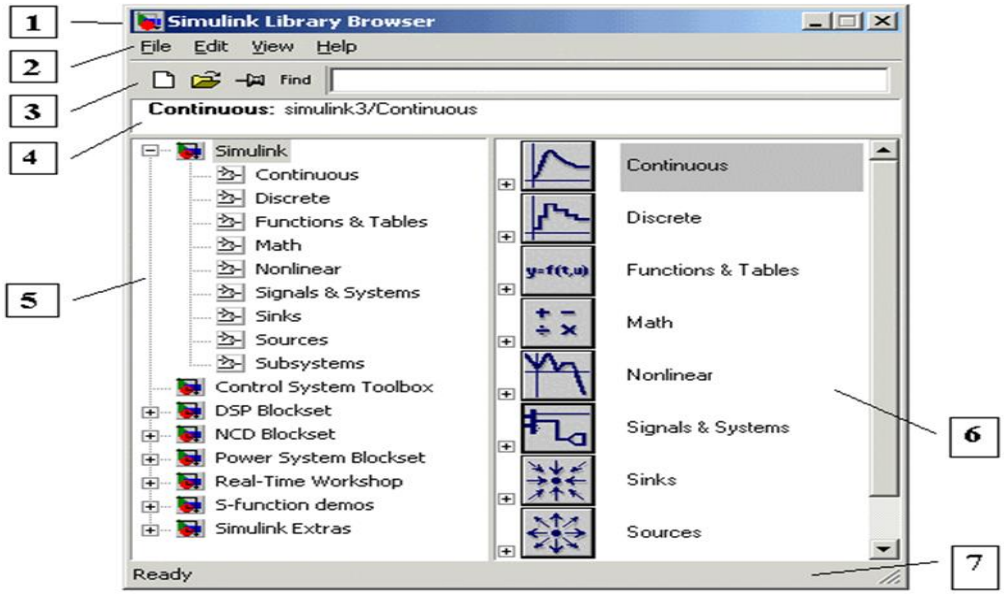


Рис.3

Таким образом, «собрав» на экране монитора из соответствующих элементов требуемую виртуальную лабораторную работу, то есть определенную структурную схему изучаемого объекта, можно выполнить ее полный анализ, изучить его в установившихся и переходных режимах и произвести его синтез с желаемыми характеристиками. При этом можно быть уверенным, что при корректной сборке схемы, (корректном выборе типовых преобразователей, установке необходимых коэффициентов типовых звеньев - элементов, правильном их конфигурировании между собой, с учетом их принципиальной схемы) и умелом проведении экспериментов, результаты исследований совпадут с результатами исследований в реальной схеме, а по точности превзойдет их. В этом суть программных пакетов «MatLab», «Simulink» и их несомненные превосходства и достоинства.

Пакеты «MatLab», «Simulink» полезны, не только для студентов в процессе изучения различных предметов при всех формах обучения, но и инженерам, научным работникам, которые проектируют и испытывают новейшие устройства, ибо эти испытания можно провести на виртуальной

модели. Это избавляет студентов и исследователей от необходимости строить физическую модель изучаемого устройства (выбирать необходимые устройства, их монтировать, заполнять необходимыми веществами, настраивать и эксплуатировать) и, следовательно, резко сокращает как материальные, так и временные затраты.

Практическая часть

Исследование параметров конкретной системы(схемы устройства) с помощью процедур системы MatLab.

Расчёт параметров и исследование функциональных зависимостей средствами системы. Используются 2 режима работы в системе MatLab. Режим командного программирования и режим моделирования с использованием пакетов приложений.

Выберем в пакете Simulink блок "Matlab function" и в параметрах пропишем путь к нашему файлу с уравнениями (файл называется Mathfuncs_Diode). Входным параметром будет напряжение, которое измеряется вольтметром, а выходным- ток.

Всю систему поместим в один блок - Subsystem. Назовем наш блок с диодом DiodeModel. Создадим маску диода

Для тестирования модели диода соберем схему Блок XY Graph строит ВАХ диода, Multimeter - графики напряжения и тока на диоде и резисторе.

Рассмотрим реакцию системы с единичной ООС и апериодическим звеном в прямой цепи на единичное воздействие. Для её набора нам понадобятся библиотеки модели: Source|Step, Sinks|Scope, Linear|Transfer Fcn и Linear|Sum. Перетянув эти элементарные модели в окно редактирования Simulink, изменяем исходные параметры в соответствии с нашей системой (двойной щелчок мышью) и соединяем их.

Чтобы запустить модель, вам достаточно набрать команду **sim** с необходимыми параметрами (наберите HELP SIM для подробной информации).

1) Выполнить следующую последовательность

a. Launch Pad ->Simulink->Library Browser

b. File->New->Model

c. В новой модели разместите следующие блоки:

- Sources->Signal Generator
- Sinks->Scope
- Math->Abs
- Sinks->Display

д. Соедините элементы в следующей последовательности и получите результат:

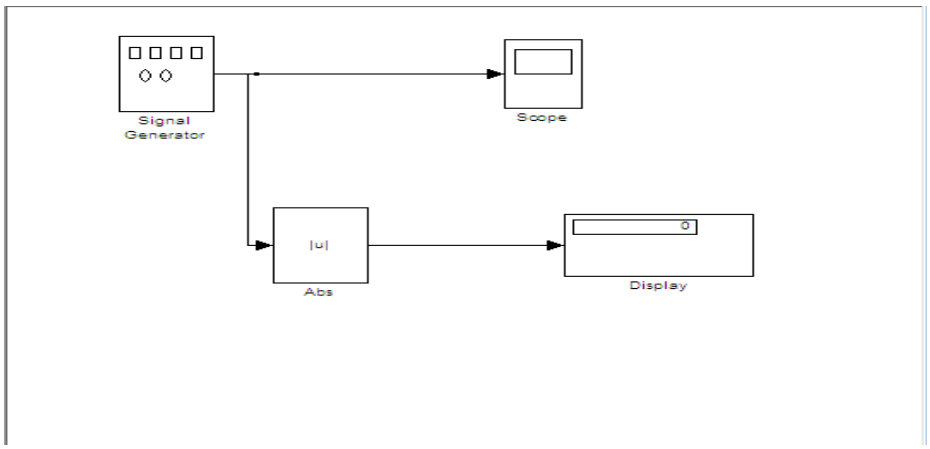


Рис.4

2) a. Launch Pad ->Simulink->Library Browser

b. File->New->Model

c. В новой модели разместите следующие блоки:

- Sources->Signal Generator
- Sinks->Scope
- Sinks->Display

д. Соедините элементы в следующей последовательности и получите результат.

Упражнение1. Собрать заданные схемы моделирования с использованием библиотеки.

1.Запустить Matlab и Simulink.

2. Настроить имя директории для хранения создаваемых файлов (команда: Home→Set Path→Add Folder1), указав свою папку на диске :

3. Создать новую модель с помощью верхнего меню окна Simulink Library Browser (команда: File→New→Model).

4.Добавить в окно модели блок Transfer Fcn (передаточная функция) из группы Continuous библиотеки Simulink, предназначенной для получения функциональных блок-схем различных устройств. Поместить блок в окне в соответствии со структурной схемой моделируемой системы.

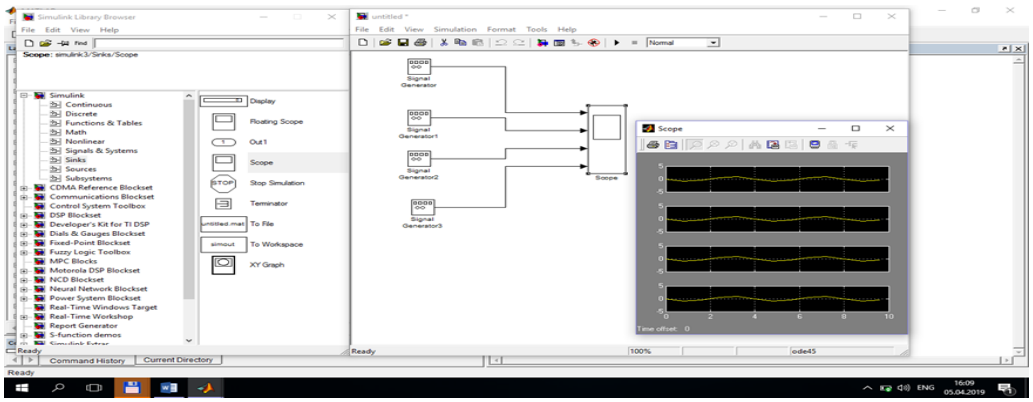


Рис.5

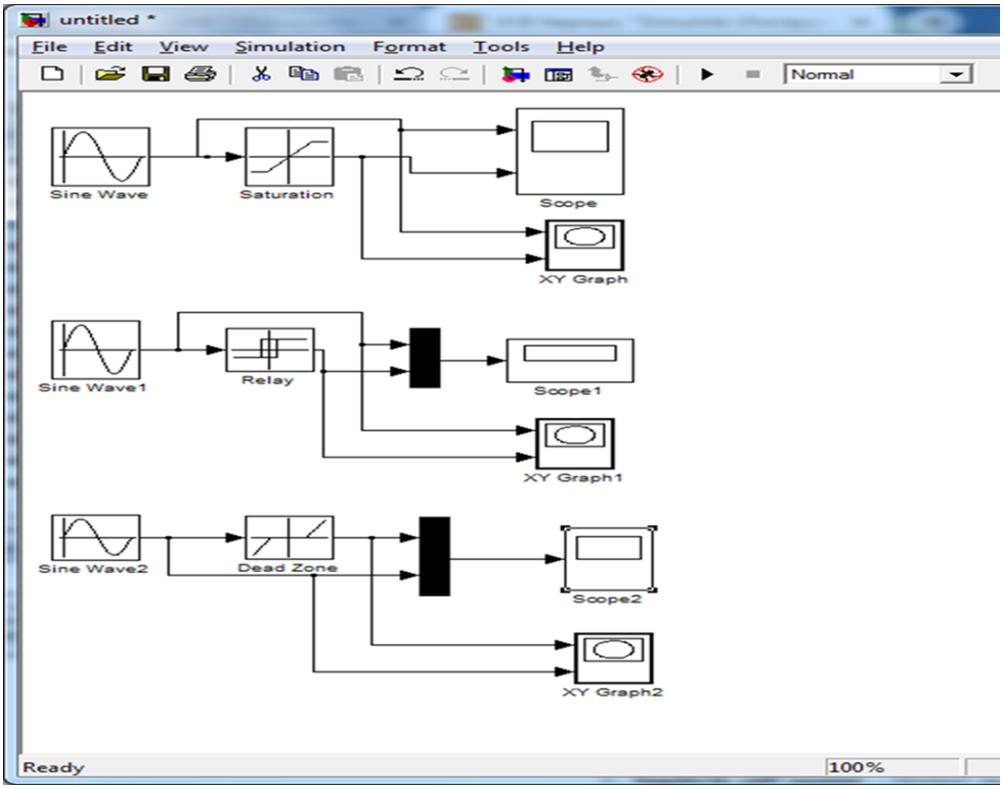


Рис.6

Полученные результаты:

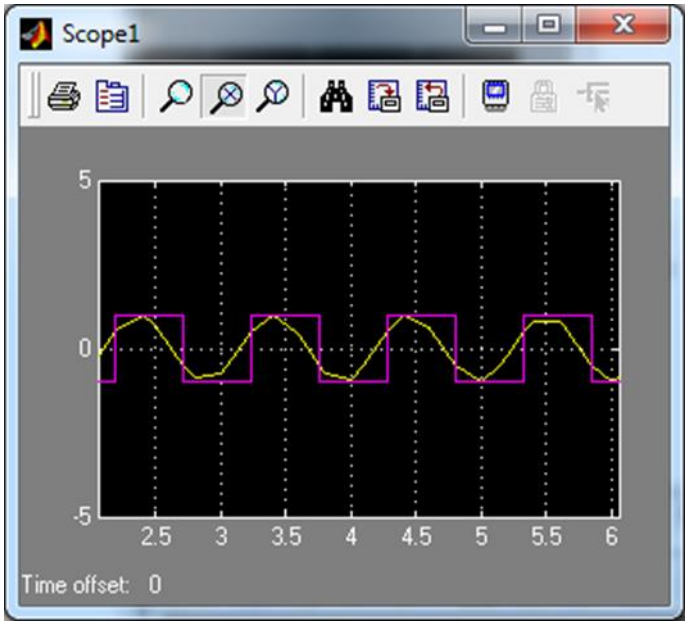


Рис.7

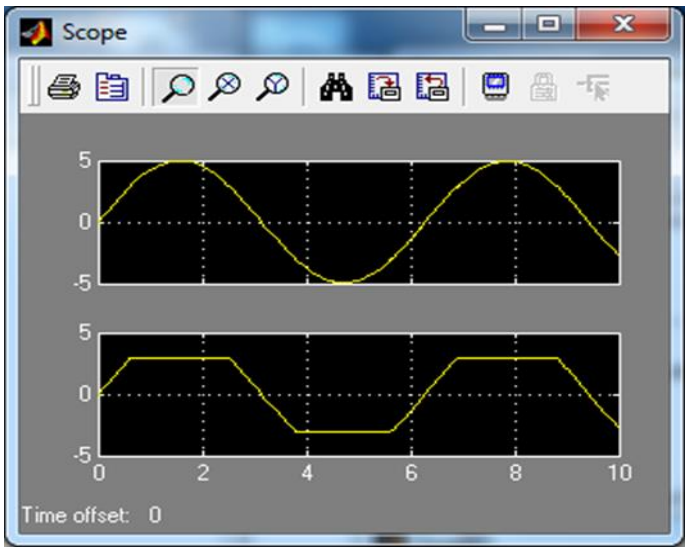


Рис.8

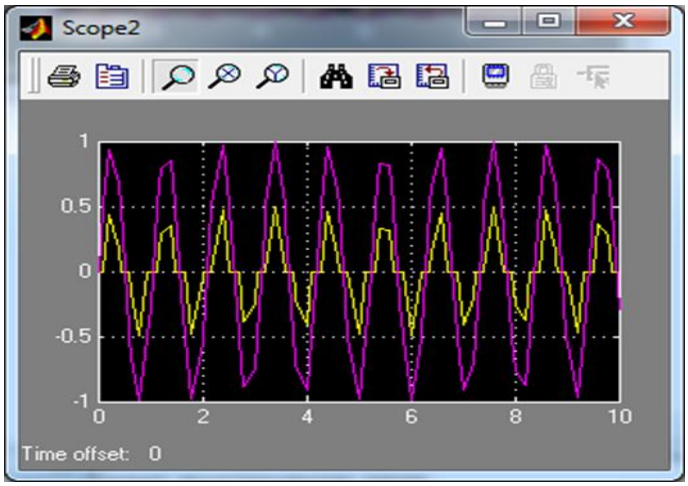


Рис.9

Упражнение 2. Построить схемы в системе Simulink по вариантам:

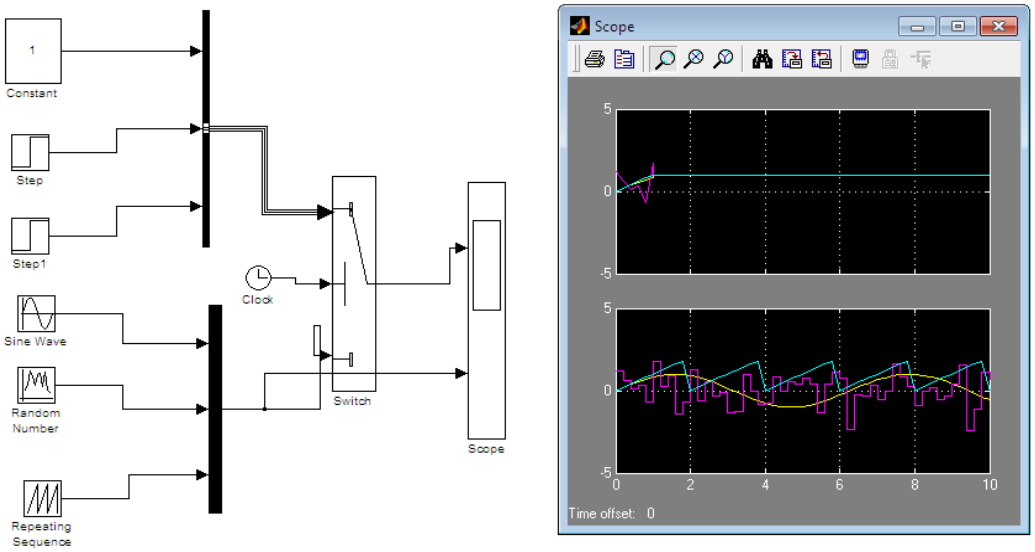


Рис.10

Упражнение 3.

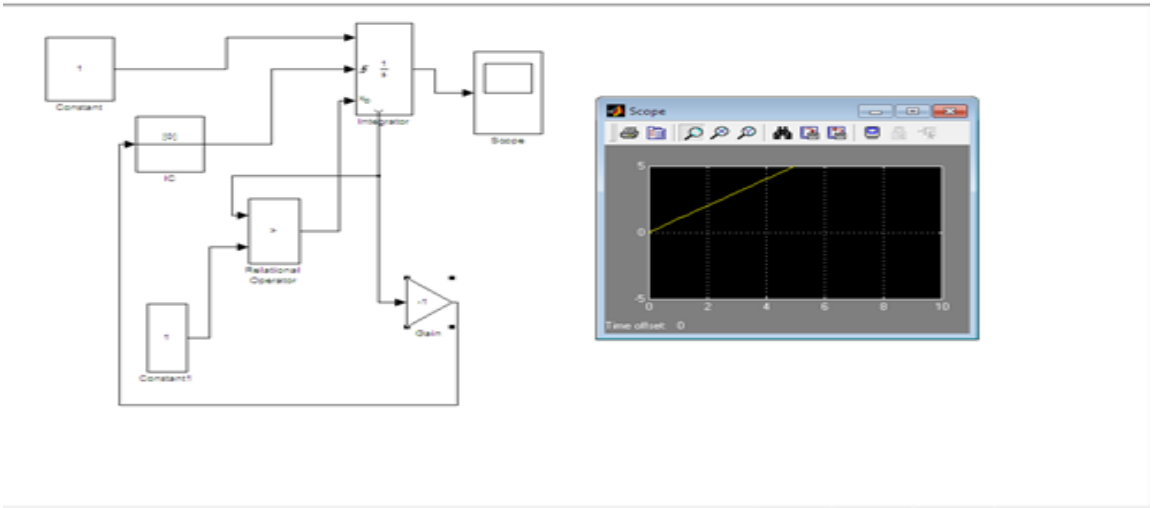


Рис.11

Упражнение 4.

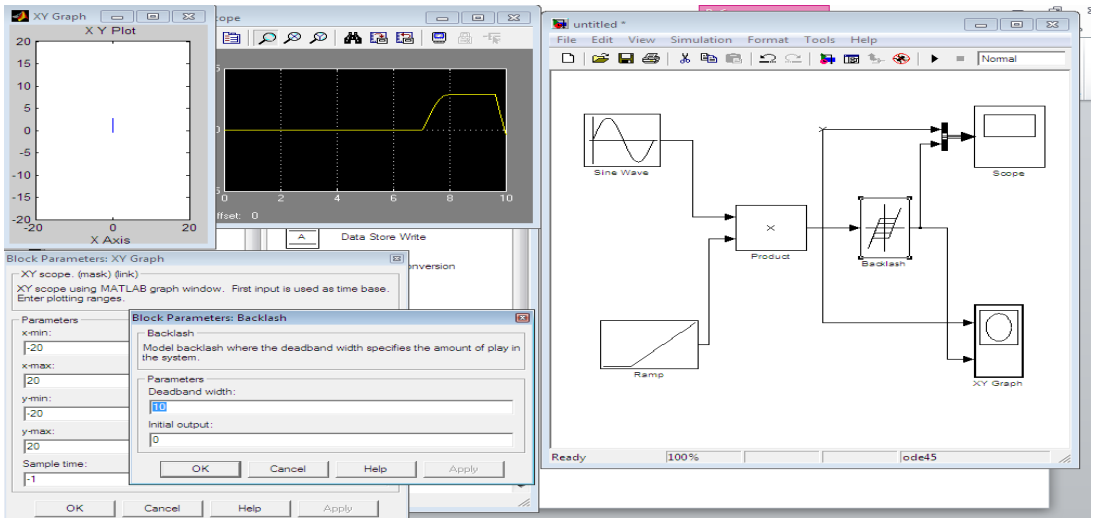


Рис.12

Упражнение 5

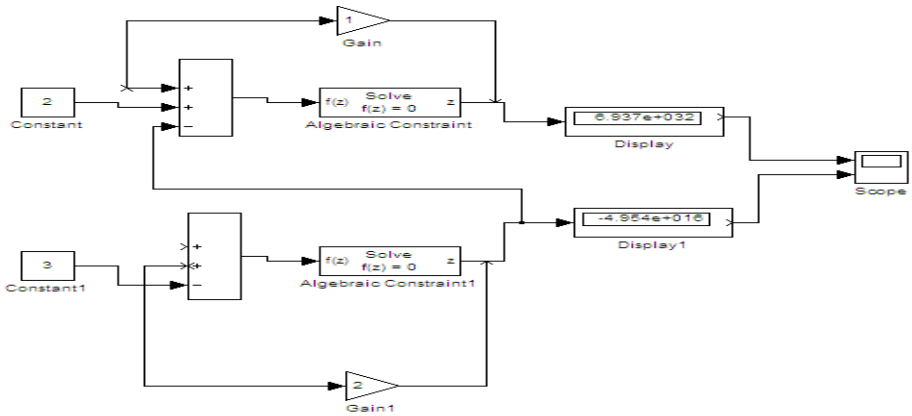


Рис.13

Контрольные вопросы

1. Из каких библиотек состоит пакет Simulink?
2. Как собрать модель в пакете Simulink?
3. Как изменить параметры моделирования?
4. Какие существуют способы визуализации процесса моделирования?
5. Перечислите возможности системы MATLAB.

Лабораторная работа №4
Моделирование энергетических задач в программе COMPAS 3D

Цель занятия: Создание моделей в программе COMPAS3D и связать с задачами энергетики.

Задания

1.Изучить теоретическую часть.
2.Ознакомиться с интерфейсом окна COMPAS 3D.
3.Создание моделя детали в 3D формате.
4.Научится навыками использования инструментов.
5.Использовать Азбуку COMPAS 3D, нарисовать детали в окне фрагмент.
6.Сделать отчёт о проделанной работе.

Теоретическая часть

Модель – это мысленно представляемая или материально реализованная система, которая, отображая или воспроизводя объект исследования, способна замещать его так, что её изучение дает новую информацию об этом объекте. Модель, представляющая собой совокупность математических соотношений, называется математической. В конечном итоге под моделью системы понимается описание системы (оригинала), отображающее определенную группу её свойств. Углубление описания – детализация модели.

Моделирование – замещение одного объекта другим с целью получения информации о важнейших свойствах объекта-оригинала с помощью объекта-модели. Таким образом, моделирование может быть определено как представление объекта моделью для получения информации об этом объекте путем проведения экспериментов с его моделью. Теория замещения одних объектов (оригиналов) другими объектами (моделями) и исследования свойств объектов на их моделях называется теорией моделирования.

Создание S- или SPS-модели.

Для создания S - или SPS модели необходимо последовательно выполнить ряд действий:

1)Вызвать окно браузера и окно модели (п.1.2.1).

2)Расположить блоки в окне модели.

3)Изменить параметры блока (если требуется), установленные программой «по умолчанию».

4)Выполнить соединение элементов (блоков) модели.

5)Сохранить модель в виде файла на диске.

Основные операции при создании и редактировании модели

Перемещение блоков из библиотеки Simulink в окно модели. Для этого необходимо открыть соответствующий раздел библиотеки (например, Sources – Источники).

Выбрать нужный блок, нажать ЛКМ и, не отпуская её, «перетащить» блок в созданное окно модели.

Выделение блоков. Для выделения отдельного объекта в модели необходимо подвести к нему курсор мыши и выполнить однократный клик ЛКМ по пиктограмме блока. Произойдет выделение объекта, о котором будут свидетельствовать маркеры (четыре квадратных метки) по углам объекта. При этом снимается выделение со всех ранее выделенных объектов. Если кликнуть ЛКМ на любой свободной области модели вне блока, то он становится не выделенным.

Для выделения нескольких объектов необходимо выполнять однократные клики ЛКМ по пиктограммам блоков с удержанием клавиши [**Shift**]. Для выделения фрагмента модели (группы блоков) необходимо установить курсор мыши вблизи группы объектов, нажать ЛКМ и, удерживая ее, начать перемещение мыши. В результате вокруг выделяемых объектов появится пунктирная рамка, размеры которой будут изменяться при перемещении мыши. Все охваченные рамкой объекты становятся выделенными.

Выделить все объекты модели можно, используя команду **Edit** (Правка)**Select All** (Выделить все) или сочетание клавиш [**Ctrl**] + [**A**].

Копирование блока. В процессе создания и редактирования модели необходимо копировать блоки, в том числе и из другой модели в текущую модель.

Для этого необходимо открыть окно модели-прототипа, выделить нужный блок и перетащить его, нажав и не отпуская правую клавишу мыши (ПКМ), в окно создаваемой (редактируемой) модели. Скопированный блок получает те же значения настраиваемых параметров, что и блок-оригинал. Копировать блоки можно и другим способом. Для этого необходимо выполнить следующие действия:

–выделить блок;

–выбрать команду **Edit** (Правка) →**Copy** (Копировать) или нажать кнопку  на панели инструментов, также можно нажать ПКМ (курсор предварительно навести на выделенный блок) и выбрать в появившемся контекстном меню команду **Copy** (Копировать).

–сделать активным окно, в которое нужно скопировать блок;
–в этом окне на панели инструментов выбрать команду **Edit →Paste** (Правка →Вставить) или нажать на кнопку  , также можно нажать ПКМ на свободном месте в окне модели и выбрать в появившемся контекстном меню команду **Paste**(Вставить).
Чтобы создать копию блока внутри модели достаточно выделить нужный блок, а затем выполнить одно из следующих действий:
– нажать сочетание клавиши [**Ctrl**] + ЛКМ и, не отпуская их, перенести блок на свободное место в окне модели;
– нажать ПКМ и, удерживая ее, перенести блок на свободное место (при перемещении блок обозначается пунктирной линией, справа от блока появляется знак «+»);
– воспользоваться вышеописанными командами и кнопками **Copy** и **Paste**.
Каждому из скопированных блоков автоматически присваивается имя с добавлением порядкового номера. Пользователь может переименовать блок. Необходимо отметить, что при выполнении данных операций объекты помещаются в собственный буфер *MatLAB* и недоступны для других приложений. Использование команды ***Edit Copy model to clipboard*** позволяет поместить графическое изображение модели в буфер Windows и соответственно, делает его доступным для остальных программ.

Перемещение блоков в модели. Любой блок модели можно переместить, выделив его, и передвинув, держа нажатой ЛКМ. При этом автоматически будут перерисованы линии связей перемещенного блока с другими (соединительные линии не разрываются, а лишь сокращаются или увеличиваются в длине). Также блок можно переместить кнопками ↑, ↓, ←, → расположенными на клавиатуре, предварительно блок следует выделить. Для одновременного перемещения нескольких блоков вместе с соединительными линиями следует выделить нужную область рамкой, а затем переместить с помощью ЛКМ один из выделенных блоков в нужное место. Остальные блоки займут новые места. Все относительные расстояния между выделенными блоками и линиями при этом сохранятся.

Удаление блоков. Для удаления блоков из модели, необходимо их выделить и нажать клавишу [**Delete**] или [**Backspace**] на клавиатуре, также можно воспользоваться командами ***Edit →Delete*** (Правка → Удалить),***Edit →Cut*** (Правка → Вырезать) или кнопкой  на панели инструментов. В случае использования команды ***Edit →Cut*** (Правка → Вырезать) или кнопки  объекты будут вырезаны (перемещены в буфер) и их можно вставить обратно в модель, воспользовавшись командой ***Edit →Paste*** (Правка → Вставить) или кнопкой на панели инструментов. При наличии соединения удаляемого блока с другим блоком линия соединения останется, но изменит свой цвет, станет пунктирной и должна быть либо удалена отдельно, либо подключена к другому блоку.

Установка параметров блока. Параметры блока устанавливаются в диалоговом окне его настройки, которое вызывается двойным кликом ЛКМ на пиктограмме блока. Следует отметить, что при задании численных параметров в качестве десятичного разделителя должна использоваться точка, а не запятая; числа могут быть представлены в виде

1e-3=10-3 или 1.5e5 = 150000;

число π обозначается как «pi»,

бесконечность – inf. После внесения изменений нужно закрыть окно кнопкой **OK**.

Создание соединительных линий. Сигналы в модели передаются по линиям связи. Линия может передавать сигнал различного вида: скалярный, векторный, матричный, комплексный. Линия соединяет выходной порт одного блока с входным портом другого или с входными портами нескольких блоков через разветвление линии.

Создание линии между блоками реализуется двумя способами:

1)Подвести курсор мыши к выходному порту блока, при этом курсор превратится в крест из тонких линий. Нажать ЛКМ и, не отпуская ее, переместить курсор ко входному порту нужного блока, при этом курсор примет вид креста из тонких сдвоенных линий. Отпустить кнопку мыши.

2)Подвести курсор мыши к выходному порту блока. Выделить блок, кликнув по пиктограмме ЛКМ, и нажать клавишу [**Ctrl]**, при этом рядом с курсором мыши появиться знак «+». Не отпуская клавишу [**Ctrl**], переместить курсор мыши на вход другого блока и вновь нажать ЛКМ. Отпустить кнопку мыши вместе с клавишей **[Ctrl].**

В результате применения одного из способов между блоками появится соединительная линия, стрелка на конце которой будет указывать направление передачи сигнала. Если между блоками появилась связь, то линия будет сплошной и черного цвета, если связи нет – пунктирной и красного цвета.

Создание разветвления линии. Для создания точки разветвления в соединительной линии необходимо подвести курсор мыши к предполагаемому узлу и, нажав ПКМ, протянуть линию к входному порту нужного блока. Можно воспользоваться и другим способом: входной порт блока подключить к уже существующей линии.

Создание петли линии выполняется аналогично перемещению блока. Линия соединения выделяется, и затем нужная часть линии перемещается.

Удаление линий выполняется также как и удаление любых других объектов.

Отделение блока от линии. Для отделения блока необходимо подвести к нему курсор мыши, нажать клавишу [**Shift]** + ЛКМ, не отпуская их переместить блок на другое место.

Изменение размеров блока. Для изменения размеров блока необходимо выделить блок, установить курсор мыши в один из углов блока и, нажав ЛКМ, изменить размер блока (курсор при этом превратится в двухстороннюю стрелку).

Изменение имени блока. Чтобы ввести новое имя блока выполняется однократный клик ЛКМ по старому имени. Имя блока будет обведено рамкой и появится текстовый курсор, после чего можно редактировать имя.

Не допускается отсутствие имени блока (пустая строка) и наличие в одном окне блоков с одинаковыми именами. При работе с блоками можно изменять пиктограммы и их окраску, разворачивать на плоскости, изменять и перемещать их названия (блочные подписи), шрифт текста и т. д. Команды форматирования блоков располагаются в меню **Format** (Формат), а также и в контекстном меню, вызываемом нажатием ПКМ на блоке.

Форматирование блока. Некоторые команды форматирования блоков:

Font – Форматирование шрифта надписей и текстовых блоков.

Show*/*Hide name – Отображение или скрытие подписи блока.

Flip name – перемещение подписи блока.

Flip block – поворот блока на 180 относительно вертикальной оси.
Rotate block – вращение блока на 90^0 по часовой стрелке.

Show drop shadow – отображение тени от блока.

Show port labels – показ меток портов.

Foreground color – выбор цвета линий для выделенных блоков.

Background color – выбор цвета фона выделенных блоков.

Добавление текстовых надписей. Для повышения наглядности модели удобно использовать текстовые надписи. Для создания надписи нужно указать мышью место надписи и дважды кликнуть ЛКМ. После этого появится прямоугольная рамка с курсором ввода. Необходимо отметить, что ранние версии пакета Simulink не адаптированы к использованию кириллических шрифтов, и их применение может привести к таким последствиям как: отображение надписей в нечитаемом виде, обрезание надписей, сообщения об ошибках, а также невозможность открыть модель после ее сохранения. Поэтому применение надписей на русском языке для многих версий Simulink *не желательно.*

Построение детали в COMPAS 3D

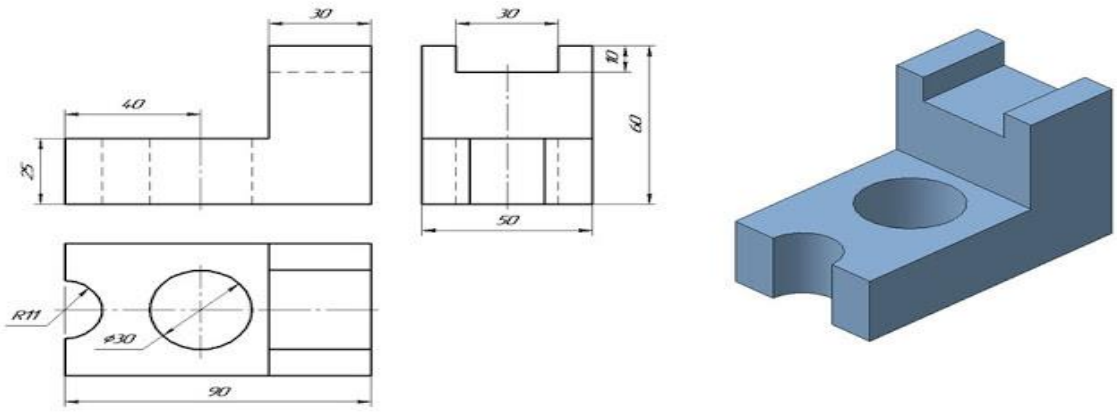


Рис.1.

Контрольные вопросы

1. Дайте определение трехмерного графика.
2. Какие шаги требуются для получения трехмерного изображения?
3. Опишите содержимое шагов для получения трехмерного изображения.
4. Перечислите основное программное обеспечение для 3-D моделирования.
5. Расскажите об истории развития трехмерного моделирования.
6. Расскажите об основных системах моделирования.

Лабораторная работа № 5

Решение задач по отраслям с помощью объектно-ориентированных систем программирования.

Цель работы: Изучить интегрированную среду C++Builder6 и методику реализации программ разветвляющихся структур.

Задания:

1. Изучить теоретическую часть
2. Загрузить среду C++ Builder6. Изучить структуру окон.
3. Ознакомиться с визуальным и консольным режимами программы C++ Builder6.
4. Реализовать разветвление в среде C++ Builder6.
5. Подготовить отчет на выполненную работу.

Теоретическая часть

Технология логического программирования включает возможности и средства языка. Эта технология реализуется управляющими операторами языка C++. В результате мы имеем программы разветвляющей и циклической структуры. Программа разветвляющейся структуры обязательно содержит условия, в зависимости от которых предусматривают выбор одного из вариантов последовательностей операторов из нескольких заданных. Для организации разветвлений в программах используются

операторы условия и выбора. Условный оператор обеспечивает выполнение или невыполнение некоторых операторов в зависимости от соблюдения определенного условия. Блок операторов, следующий за конструкцией **if** выполняется лишь при истинности некоторого условия. Формат условного оператора:

i f (выражение) оператор_ 1 ; [**else** оператор _ 2 ;]

Вначале определяется значение *выражения,* оно может иметь арифметический тип или тип указателя. Если выражение равно нулю (то есть не равно true), выполняется первый оператор, а иначе – второй оператор. Часть else можно опустить, тогда если условие не выполняется, управление переходит к оператору, следующему за первым.

К примеру. В зависимости от года рождения, хранящегося в переменной b__year, выведем сообщение, гласящее, достиг ли пользователь совершеннолетия (текущий год хранится в переменной year);

```
if ( year b_year > 18)
    cout << "Вы достигли совершеннолетия " ;
else
  cout << "Вам осталось еще  " << 18 -- (y e  a r   --  b __y e a r )  <<
          <<  " лет до совершеннолетия.";
```

Структура выбора switch.

Оператор **switch** предназначен для разветвления процесса вычислений на несколько направлений. Его формат:

```
switch ( выражение )  {
case константное_ выражение_1 : [список_операторов_1]
case константное_ выражение_2 : [список_операторов_2]
  . . .
case константное_ выражение_n : [список_операторов_n]
[default: опеаторы ]
}
```

Работает эта структура следующим образом: вначале вычисляется выражение (оно должно быть целочисленного типа), затем среди конструкций ***case*** ищется такая, константное выражение которой совпадает со значением выражения в условии. Если такое совпадение найдено, происходит выполнение соответствующего списка операторов. Чтобы организовать выход из структуры ***switch*** в этих списках используется оператор ***break***.

Если не было найдено ни одного совпадения с константными выражениями, выполняется ветка **default,** которая может и отсутствовать

Для примера использования структуры **switch** в реальной программе, рассмотрим случай тестовой программы, когда за каждый из ответов на тестовый вопрос начисляется разное количество баллов. Допустим, есть

четыре варианта ответа (a, b, c, d), введенный ответ содержится в переменной answer типа char, а текущее количество баллов в переменной total:

```
switch (answer) {
   case ‘a’, ’A’:  total += 2;  break;
   case ‘b’, ’B’:  total += 1;  break;
   case ‘c’, ’C’:  total += 4;  break;
   case ‘d’, ’D’:  break;   // 0 баллов, сумма не меняется
   default: cout << “Вы ввели неверный ответ!”;
}
```

Обратите внимание: структура switch может проверять выражение на совпадение только с константами. Если необходимо проверить некоторый диапазон или значение переменной, приходится использовать if else if.

Решение одного варианта

Задача 1: Создать программу на языке C++Bulder6 в консольном режиме на нижеприведённую функцию:

$$y = \begin{cases} 4 - x^2, & 0 < x < 4 \\ 0, & x = 0 \\ x^3, & x < 0 \\ 1, & \textit{в других случаях} \end{cases}$$

```
//---------------------------------------------------------------------------
#include <iostream.h>
#include<conio.h>
#include<math.h>
#include <vcl.h>
#pragma hdrstop
//---------------------------------------------------------------------------
#pragma argsused
int main(int argc, char* argv[])
{  float x,y;
   cout<<"x=";
   cin>>x;;
     if  ((x>0) && (x<4))  y= 4-pow(x,2);
      else  if   (x<0)   y=pow(x,3);
            else  if   (x=0)  y=0;
            else     y=1;
    cout<<"y="<<y;
    getch();
          return 0;
}
//---------------------------------------------------------------
```

Задача 2. Написать программу преобразования цифр от 0 до 9 в слова.

```
 //---------------------------------------------------------------------------
 #include <iostream.h>
 #include<conio.h>
```

```
#include <vcl.h>
#pragma hdrstop
//---------------------------------------------------------------------------
#pragma argsused
int main(int argc, char* argv[])
{  int a;
   cout<<"Vvedite sifru";
   cin>>a;;
switch (a)
{
case 0 : cout<<"nol"; break;
case 1 : cout<<"odin";  break;
case 2 : cout<<"dva";   break;
case 3 : cout<<"tri";;   break;
case 4 : cout<<"chetire";  break;
case 5 : cout<<"pyat";  break;
case 6 : cout<<"shest";  break;
case 7 : cout<<"sem";   break;
case 8 : cout<<"vosem";   break;
case 9 : cout<<"devyat";   break;
default : cout<<"sifra ne opredelena"  ;
  }
getch();
 return 0;
  }
//---------------------------------------------------------------------------
```

В визуальном режиме программирования для осуществления выбора из вариантов можно использовать компоненту **RadioCroup**  , которая входит в палитру компонентов **Standart.**

Задача 3. Вычислить при условии x>0 x^3, x<0 x^2 , x если x=0.

$$Y=\begin{cases} x^3 & x>0 \\ x^2 & x<0 \\ 0 & x=0 \end{cases}$$

В этой лабораторной работе используются компоненты RadioGroup, Lable, Edit, Button: Число Edit зависит от того, сколько данных нужно ввести вручную пользователю. Button1 используется для вычисления X. В компоненте RadioGroup можно увидеть, как меняется способ вычисления X, это зависит от введенных пользователем данных (рис.4). Если ввести отрицательное значение, то решение будет проводится по способу: $x=X^2$. Если ввести положительное значение, то решение будет проводится по способу: $x=X^3$. Если ввести нулевое значение, то решение будет проводится по способу: x=X. Чтобы установить кнопки для выбора условий в Object

Inspector устанавливаем свойства Items, задав строки условий в окне ***String List Editor*** :

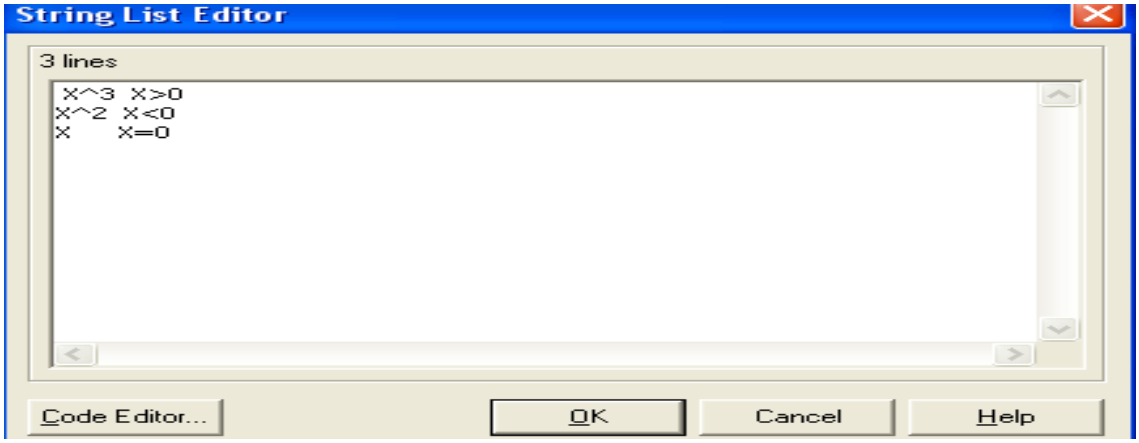


Рис.1.

Код программы

```
//---------------------------------------------------------------------------
#include <math.h>
#include <vcl.h>
#pragma hdrstop

#include "Unit15.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button1Click(TObject *Sender)
{
float x;
   x= StrToFloat(Edit1->Text);
   if (StrToFloat(Edit1->Text)>0)
   {        RadioGroup1->ItemIndex =0;
        x=pow(x,3);
        Label1->Caption="Otvet: "+FloatToStr(x);
    }
  else if (StrToFloat(Edit1->Text)<0)
     {   RadioGroup1->ItemIndex = 1;
        x=x*x;
        Label2->Caption="Otvet : " + FloatToStr(x);
```

```
    }
  else
    {   RadioGroup1->ItemIndex = 2;
      Label3->Caption="Otvet : X=0" ;
     }
}
//---------------------------------------------------------------------------
```

В результате выполнения программы будут получены следующие результаты (рис.2):

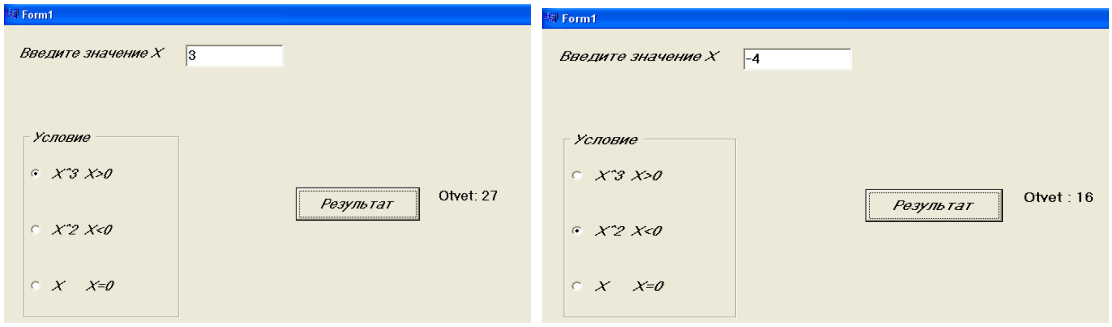


Рис.2. Форма с компонентом RadioGroup и полученные результаты

Варианты заданий для выполнения лабораторной работы:

$$1.\ a=\begin{cases}(f(x)+y)^2-\sqrt{f(x)y}, & xy\rangle 0\\ (f(x)+y)^2+\sqrt{|f(x)y|}, & xy\langle 0\\ (f(x)+y)^2+1, & xy=0\end{cases}$$

$$2.\ c=\begin{cases}f(x)^2+y^2+\sin(y), & x-y=0\\ (f(x)-y)^2+\cos(y), & x-y\rangle 0\\ (y-f(x))^2+tg(y), & x-y\langle 0.\end{cases}$$

$$3.\ d=\begin{cases}(f(x)-y)^3+arctg(f(x)), & x\rangle y\\ (y-f(x))^3+arctg(f(x)), & y\rangle x\\ (y+f(x))^3+0.5, & y=x\end{cases}$$

$$4.\ e=\begin{cases}i\sqrt{f(x)}, i-ток, & x\rangle 0\\ i/2\sqrt{|f(x)|}, i-жуфт, & x\langle 0\\ \sqrt{|if(x)|}, & акс\ холда\end{cases}$$

5. $g=\begin{cases} e^{f(x)-|b|}, & 0.5\langle xb\langle 10 \\ \sqrt{|f(x)+b|}, & 0.1\langle xb\langle 0.5 \\ 2f(x)^2, & \text{акс холда} \end{cases}$

6. $s=\begin{cases} e^{f(x)}, & 1\langle xb\langle 10 \\ \sqrt{|f(x)+4*b|}, & 12\langle xb\langle 40 \\ bf(x)^2, & \text{акс холда} \end{cases}$

7. $j=\begin{cases} \sin\left(5f(x)+3m|f(x)|\right), & -1\langle m\langle x \\ \cos\left(3f(x)+5m|f(x)|\right), & x\rangle m \\ (f(x)+m)^2, & x=m. \end{cases}$

8. $l=\begin{cases} 2f(x)^3+3p^2, & x\rangle|p| \\ |f(x)-p|, & 3\langle x\langle|p| \\ (f(x)-p)^2, & x=|p|. \end{cases}$

9 $k=\begin{cases} \ln\left(|f(x)|+|q|\right), & |xq|\rangle 10 \\ e^{f(x)+q}, & |xq|\langle 10 \\ f(x)+q, & |xq|=10 \end{cases}$

Варианты 2 сложности

1. $y=a\ln(1+x^{1/5})+\cos^2[\varphi(x)+1]$, где $x=\begin{cases} z^2; & z<1; \\ z+1; & z\ge 1. \end{cases}$

2. $y=\dfrac{2a\varphi(x)+b\cos\sqrt{|x|}}{x^2+5}$, где $x=\begin{cases} 2+z; & z<1; \\ \sin^2 z; & z\ge 1. \end{cases}$

3. $y=-\pi\varphi(x)+a\cos^2 x^3+b\sin^3 x^2$, где $x=\begin{cases} z; & z<1; \\ \sqrt{z^3}; & z\ge 1. \end{cases}$

4. $y=2a\cos^3 x^2+\sin^2 x^3-b\varphi(x)$, где $x=\begin{cases} z^3+0{,}2; & z<1; \\ z+\ln z; & z\ge 1. \end{cases}$

5. $y=a\varphi(x)-\ln(x+2{,}5)+b(e^x-e^{-x})$, где $x=\begin{cases} -z/3; & z<-1; \\ |z|; & z\ge -1. \end{cases}$

6. $y=\dfrac{2}{3}a\sin^2 x-\dfrac{3b}{4}\cos^2\varphi(x)$, где $x=\begin{cases} z; & z<0; \\ \sin z; & z\ge 0. \end{cases}$

7. $y = \sin^3[c\varphi(x) + d^2 + x^2]$, где $x = \begin{cases} z^2 - z; & z < 0; \\ z^3; & z \ge 0. \end{cases}$

8. $y = \sin^2 \varphi(x) + a\cos^5 x^3 + c\ln x^{2/5}$, где $x = \begin{cases} 2z + 1; & z \ge 0; \\ \ln(z^2 - z); & z < 0. \end{cases}$

9. $y = \dfrac{b\varphi(x)}{\cos x} + a\ln\left|tg\dfrac{x}{2}\right|$, где $x = \begin{cases} z^2/2; & z \le 0; \\ \sqrt{z}; & z > 0. \end{cases}$

10. $y = \dfrac{d\varphi(x)e^{\sin^3 x} + c\ln(x+1)}{\sqrt{x}}$, где $x = \begin{cases} z^2 + 1; & z < 1; \\ z - 1; & z \ge 1; \end{cases}$

11. $y = \dfrac{2{,}5a \cdot e^{-3x} - 4bx^2}{\ln|x| + \varphi(x)}$, где $x = \begin{cases} \dfrac{1}{z^2 + 2z}; & z > 0; \\ 1 - z^3; & z \le 0. \end{cases}$

12. $y = a\sin^3[\varphi(x)^2 - 1] + c\ln|x| + e^x$, где $x = \begin{cases} z^2 + 1; & z \le 1; \\ 1/\sqrt{z-1}; & z > 1. \end{cases}$

13. $y = \sin[n\varphi(x)] + \cos kx + \ln mx$, где $x = \begin{cases} z; & z > 1; \\ z^2 + 1; & z \le 1. \end{cases}$

14. $y = b\cos[a\varphi(x)] + \sin\dfrac{x}{5} + ae^x$, где $x = \begin{cases} \sqrt{z}; & z > 0; \\ 3z + 1; & z \le 0. \end{cases}$

15. $y = 2\varphi(x)[a\sin x + d \cdot e^{-(x+3)}]$, где $x = \begin{cases} -3z; & z > 0; \\ z^2; & z \le 0. \end{cases}$

$y = a\ln|x| + e^x + c\sin^3[\varphi(x)^2 - 1]$, **где** $x = \begin{cases} z^2 + 1; & z \le 1; \\ 1/\sqrt{z-1}; & z > 1. \end{cases}$

Контрольные вопросы:

1. Что такое разветвляющийся вычислительный процесс?
2. Какие виды условного оператора вы знаете?
3. Когда используется оператор варианта?
4. Какого типа константы используются в качестве меток в операторе варианта?
5. Какие компоненты C++ используются в программе?
6. Для чего используется компонента RadioGroup?
7. Каким образом устанавливаются кнопки выбора в компоненте RadioGroup?

Лабораторная работа №6
Создание структурированных приложений в объектно-ориентированных системах программирования.

Цель работы: Изучить методику и приобрести навыки по реализации программ со структурными (**struct)** типами данных.

Задания:
1. Изучить теоретическую часть.
2. Загрузить систему C++ Builder 6.
3. Создать программу для предложенного варианта в визуальном режиме.
4. Реализовать программу и проанализировать результат.
5. Подготовить отчет о проделанной работе.

Теоретическая часть

Довольно часто, вполне оправданным, является представление некоторых элементов данных в качестве составных частей другой, более крупной логической единицы. Представляется естественным сгруппировать информацию, например, о номере дома, названии улицы и города в единое целое и назвать адресом, а объединенную информацию о дне, месяце и годе рождения назвать датой.

На языке C++ для описания такого типа данных определен тип структура. В отличие от массива, все элементы которого однотипны, структура может содержать элементы разных типов. В языке C++ структура является видом класса и обладает всеми его свойствами, но во многих случаях достаточно использовать структуры так, как они определены в языке C:

```
struct [ имя_типа ]
{тип_1 элемент_1;
 тип_2 элемент_2;
 тип_n элемент_n;}
[ список_описателей ];
```

Элементы структуры называются ***полями структуры*** и могут иметь любой тип, кроме типа этой же структуры, но могут быть указателями на него. Если отсутствует имя типа, должен быть указан список описателей переменных, указателей или массивов. В этом случае описание структуры служит определением элементов списка:

```
struct
{
char fio[30];
int date, code;
float salary;
}stuff[100], *ps;        /*определение массива структур и указателя на структуру */
```

Если список отсутствует, описание структуры определяет новый тип, имя которого можно использовать в дальнейшем наряду со стандартными типами, например:

```
struct Worker
{  //описание нового типа Worker
char fio[30];
int date, code;
float salary;
};          //описание заканчивается точкой с запятой
Worker stuff[100], *ps; /* определение массива типа Worker */
                                    /* и указателя на тип Worker */
```

Для **инициализации структуры** значения её элементов перечисляют в фигурных скобках в порядке их описания:

```
Struct
{
char fio[30];
int date, code;
float salary;
}worker = {"Страусенко", 31, 215, 3400.55};
```

Для переменных одного и того же структурного типа определена операция присваивания, при этом происходит поэлементное копирование. Доступ к полям структуры выполняется с помощью операций выбора . (точка) при обращении к полю через имя структуры и -> при обращении через указатель, например:

```
Worker worker, stuff[100], *ps;
...
worker.fio = "Страусенко";
stuff[8].code = 215;
ps->salary = 0.12;
```

Решение одного варианта:

Задача 1. Дан массив, содержащий записи о книгах. Сведения о каждой из книг - название книги, фамилия автора, год издания. Программа определяет количество книг, год издания которых позже или равен 2008 г. Код программы в консольном режиме запишем в следующем виде:

```
//---------------------------------------------------------------------------
#include <iostream.h>
#include <conio.h>
#include <vcl.h>
#pragma hdrstop
//---------------------------------------------------------------------------
#pragma argsused
int main(int argc, char* argv[])
{ typedef struct {
      char   title[40];
```

```
        char   author[20];
        int    entry;   }book ;
    int sum=0;
    book  k;
    book b[10];
    int  i;
    for (i=1; i<=5; i++)
    { cout<<"naimenovanie knigi"<<endl;
       cin>>b[i].title;
     cout<<"avtor"<<endl;
        cin>>b[i].author;
     cout<<"god izdanie"<<endl;
         cin>>b[i].entry;
     }
    for (i=1; i<=5; i++)
       if (b[i].entry<=2008)  sum=sum+1;
     cout<<"Summa knig sum="<<sum;
    getch();
     return 0;
    }
    //---------------------------------------------------------------------------
```

Задача 2. Задана квадратная матрица. Вычислить сумму всех элементов массива.(визуальный режим). В программе используются компоненты Lable, Button, **StringGrid (**свойство Cells определяет индекс компоненты). Button1(кнопка) используется для заполнения массива (**Stringrid**). Button2 - для вычисления суммы всех элементов массива согласно коду программы.

Код программы:

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
int sum;
int i,j, a[3][3];
```

```
randomize();
for(i=0;i<3;i++)
for(j=0;j<3;j++)
{ a[i][j]=random(10);
StringGrid1->Cells[i][j]=IntToStr(a[i][j]);
sum=sum+a[i][j];}
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
  Label1->Caption="Сумма всех элементов=" +IntToStr(sum);
}
//---------------------------------------------------------------------------
```

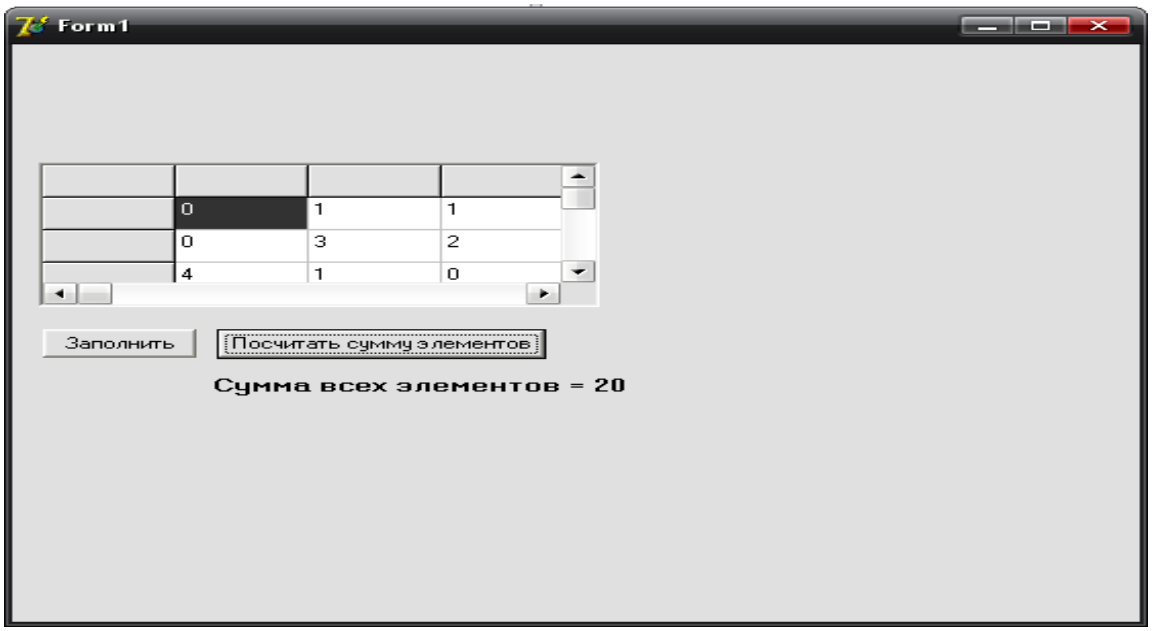


Рис.1

Варианты заданий для выполнения лабораторной работы:

1.	Грузовое судно имеет грузоподъёмность 30т. Погрузка производится по списку в порядке очереди. Фиксируется фамилия отправителя и вес. Вывести на печать список: фамилия, вес не погруженных грузов.
2.	В соревнованиях по прыжкам в высоту участвуют n спортсменов. Определить лучший результат и вывести список: фамилия, результат в порядке занятых мест.
3.	В киоске продаются аудиокассеты с детской, эстрадной и классической музыкой. Время звучания и цена – разные. Вычислить вырученную за день сумму и вывести список: время, цена, вид музыки.
4.	Результаты переписи населения хранятся в памяти ЭВМ. Напечатать фамилии и подсчитать общее число жителей, родившихся в 1958 году.

5.	Из корзины вынимают цветы и фиксируют окраску (белая, красная, желтая) и вид цветка (роза, гвоздика). Определить и вывести на печать количество белых роз и красных гвоздик.
6.	Абитуриенты сдают пробные экзамены и получают сумму очков до 100. Вывести на печать фамилии и результаты семи лучших абитуриентов. Определить средний балл по всем абитуриентам и по семи лучшим.
7.	В баскетбольном матче фиксируются: шифр команды, номер и фамилия оштрафованного игрока, штрафное время (если игрок не удален, то 0). Вывести на печать по каждой команде фамилии с нулевым временем.
8.	Ученикам 1-го класса назначается дополнительно стакан молока (200мл), если их вес составляет меньше 30 кг. Определить, сколько литров молока потребуется ежедневно для одного класса, состоящего из n учеников. После взвешивания вес каждого ученика вводится в ЭВМ.
9.	Составить программу для контроля знаний. В программе задается один вопрос, ответ на который включает несколько наименований (например, назовите все элементы периодической системы, представляющие группу галогенов, и т.д.). В памяти ЭВМ хранится список наименований, являющихся полным ответом на вопрос. Введенный ответ необходимо сравнить с правильным.
10.	Составить программу успеваемости студентов по 4-м предметам: математике, физике, информатике, и химии. В программе требуется предусмотреть редактирование данных, поиск студентов и их данных по известному параметру (по фамилиям, названию факультета, по номеру группы). Программа должна быть оформлена в виде меню "ввод данных, поиск, корректировка, выход".{ESC}

Контрольные вопросы:

1. Дайте определение комбинированному типу. Что такое запись, элементы записи, поля записи?
2. Описание записи в программе (на примерах).
3. Правила корректного представления элементов записи в программе.
4. Какие операции выполняются над полями записи?
5. Объясните оператор присоединения.
6. Комбинированный тип с вложенной структурой. Правила доступа к полям записи вложенной структуры.

Лабораторная работа №7
Графические возможности Borland C++ Bulder6
Построение графиков функций и изображений.

Цель работы: Освоить методику создания графических изображений с помощью инструментов Borland C++ Builder6 и изучить некоторые возможности построения графиков функций с помощью компонент *Chart* и *Image*; научиться работать с графическими объектами; написать и отладить программу с использованием функций отображения графической информации.

Задания:

1. Изучить теоретическую часть.
2. Составить программу для построения графического изображения.
3. Реализовать программу в визуальном режиме.
4. Подготовить отчет о проделанной работе.

Теоретическая часть

Как и в любой системе программирования в системе Borland C++ Builder6 есть возможность работы с графиком. Для работы с графиком существуют два специальных класса *TGraphic* и *TPicture*. Класс *TGraphic* обеспечивает создание 3-х типов файлов: пиктограммы, метафайлы и растровые изображения. Имеются специальные классы *TFont*-стили и размеры пера, *TPen*-перо, стили пера, *TBrush*-стили кисти.

Значение свойства Color определяет цвет линии(таб.3)

Таблица3.

	Константа	**Цвет**	**Константа**	**Цвет**	
	clBlack	Черный	clSilver	Серебристый	
	clMaroon	Каштановый	clRed	Красный	
	clGreen	Зеленый	clLime	Салатный	
	clOlive	Оливковый	clBlue	Синий	
	clNavy	Темно-синий	clFuchsia	Ярко-розовый	
	clPurple	Розовый	clAqua	Бирюзовый	
	clTeal	Зелено-голубой	clWhite	Белый	
	clGray	Серый			

Свойство ***style*** определяет вид (стиль) линии, которая может быть непрерывной или прерывистой, состоящей из штрихов различной длины. В табл.4 перечислены именованные константы, позволяющие задать стиль закрашивания.

Значения свойства Brush, style определяют тип закрашивания
Таблица 4.

Константа	Тип заполнения (заливки) области
bsSolid	Сплошная заливка
bsClear	Область не закрашивается
bsHorizontal	Горизонтальная штриховка
bsVertical	Вертикальная штриховка
bsFDiagonal	Диагональная штриховка с наклоном линий вперед
bsBDiagonal	Диагональная штриховка с наклоном линий назад
bsCross	Горизонтально-вертикальная штриховка, в клетку
bsDiagCross	Диагональная штриховка, в клетку

Построение графиков с помощью компоненты Chart

Обычно результаты расчетов представляются в виде графиков и диаграмм. Система *Builder* имеет мощный пакет стандартных программ вывода на экран и редактирования графической информации, который реализуется с помощью компоненты *Chart*, находящейся на панеле компонет *Additional* -  .

Построение графика (диаграммы) производится по вычисленным значениям координат точек x и $y = f(x)$, которые с помощью метода *AddXY* передаются в специальный двухмерный массив *Series*[***k***] компоненты *Chart* (***k*** = 0,1,2,... – номер используемого графика).

Компонента *Chart* строит и размечает оси, рисует координатную сетку, подписывает название осей и самого графика, отображает переданные точки в виде графиков или диаграмм.

Установив компоненту *Chart*1 на форму, для изменения ее параметров двойным щелчком кнопкой мыши вызываем окно редактирования *EditingChat*1 (рис. 47). Для создания *Series*1 нажимаем кнопку *Add* на странице *Series*.

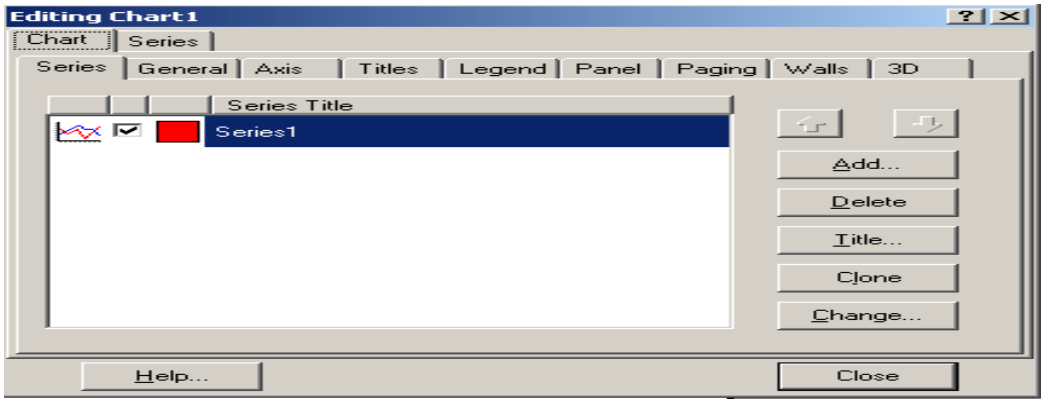


Рис.1.

В появившемся после этого окне *TeeChart Gallery* выбираем пиктограмму с надписью *Line* (график выводится в виде линий). Если нет необходимости представления графика в трехмерном виде, отключается независимый переключатель *3D*. Для изменения названия нажимаем кнопку *Title*. Название графика вводится на странице *Titles*.

Данные по оси *X* автоматически сортируются, поэтому, если необходимо нарисовать, например, окружность, сортировку отключают функцией *Order*: Chart1->Series[0]->XValues->Order = loNone.

Нажимая различные кнопки меню, познакомьтесь с другими возможностями редактора *EditingChat*.

Использование класса Canvas

Для рисования используется класс типа *TCanvas*, который является не самостоятельной компонентой, а свойством многих компонент, таких как *Image, PaintBox*, и представляет собой холст (контекст *GDI* в *Windows*) с набором инструментов для рисования. Каждая точка холста имеет свои координаты. Начало осей координат располагается в верхнем левом углу холста. Данные по оси *X* увеличиваются слева направо, а по оси *Y* сверху вниз. Компонента *Image* находится на странице *Additional*, а *PaintBox* – *System*.

Основные свойства класса *Canvas*:

Pen – перо (определяет параметры линий),

Brush – кисть (определяет фон и заполнение замкнутых фигур),

Font – шрифт (определяет параметры шрифта).

Некоторые методы класса *Canvas*:

Ellipse(*x*1,*y*1, *x*2,*y*2) – чертит эллипс в охватывающем прямоугольнике (*x*1, *y*1), (*x*2, *y*2) и заполняет внутреннее пространство эллипса текущей кистью;

MoveTo(*x*,*y*) – перемещает карандаш в положение (*x*,*y*);

LineTo(*x*,*y*) – чертит линию от текущего положения пера до точки (*x*,*y*);

Rectangle(*x*1,*y*1, *x*2,*y*2) – вычерчивает и заполняет прямоугольник (*x*1,*y*1), (*x*2, *y*2). Для вычерчивания без заполнения используйте *FrameRect* или *Polyline*;

Polygon(*const TPoint* Points, const int Points_Size*) – вычерчивает многоугольник по точкам, заданным в массиве *Points* размера *Points_Size*. Конечная точка соединяется с начальной и многоугольник заполняется текущей кистью. Для вычерчивания без заполнения используется метод *Polyline*.

TextOut(*x, y, const AnsiString Text*) – выводит строку *Text* так, чтобы левый верхний угол прямоугольника, охватывающего текст, располагался в точке (*x, y*).

Пример создания оконного приложения

Задача1.Написать программу отображения графика выбранной функции с помощью компонент *Chart* и *Image*.

Настройка формы Панель диалога программы с полученными результатами представлена на рис.48.

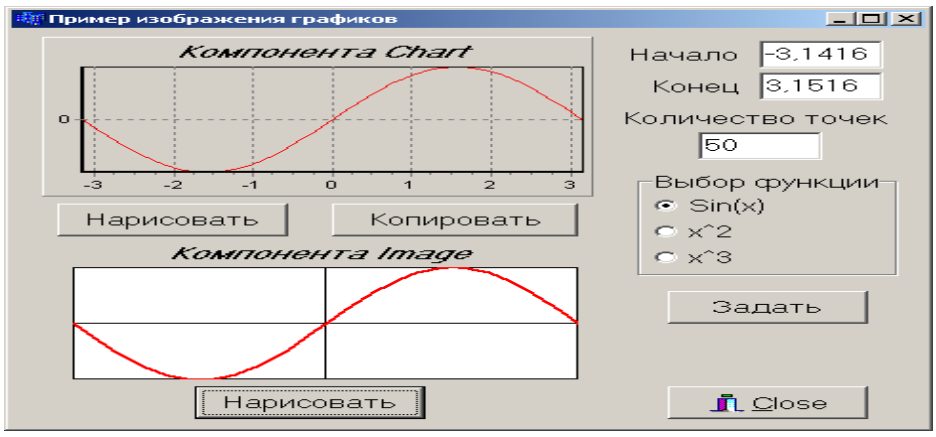


Рис.2.

Текст программы, реализующий поставленную задачу, может иметь следующий вид:

```
//---------------------------------------------------------------------------
  double a,b,h,y_min,y_max;
  int n;
  typedef double (*Tfun)(double);
  Tfun f;
  double fun0(double);
  double fun1(double);
  double fun2(double);
//---------------------------------------------------------------------------
void __fastcall TForm1::FormCreate(TObject *Sender)
{
        Edit1->Text="-3,1416";    // a
        Edit2->Text="3,1416";     // b
        Edit3->Text="50";         // n
        RadioGroup1->ItemIndex = 0;   }
//----------------- Задать начальные значения  ----------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
double x, r;
a=StrToFloat(Edit1->Text);
b=StrToFloat(Edit2->Text);
n=StrToInt(Edit3->Text);
h = (b-a)/n;
switch(RadioGroup1->ItemIndex) {
        case 0: f = fun0; break;
        case 1: f = fun1; break;
        case 2: f = fun2; break;   }
y_min = y_max = f(a);
for (x = a+h; x<=b; x+=h)
{               r = f(x);
              if(y_min>r) y_min = r;
```

```
            if(y_max<r) y_max = r;
}     }
//----------------------- Построить график в Chart ---------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Chart1->Series[0]->Clear();                    // Очистка графика
for(double x=a; x<=b; x+=h)
     Chart1->Series[0]->AddXY(x,f(x));
}
//--------------------- Копировать в буфер -------------------------------------
void __fastcall TForm1::Button3Click(TObject *Sender)
{
 Chart1->CopyToClipboardMetafile(True);
}
//--------------------- Построить график в Image --------------------------
void __fastcall TForm1::Button4Click(TObject *Sender)
{
int xmax, ymax, xt, yt, y0, x0;
double hx,hy,x;
  Image1->Canvas->Pen->Color=clBlack;    // Установка цвета пера
// Поиск координат правого нижнего угла холста Image
 xmax = Image1->Width;                    ymax = Image1->Height;
// Закрашивание холста Image текущей белой кистью
  Image1->Canvas->Rectangle(0,0,xmax,ymax);
// Поиск середины холста
  y0=ymax/2;          x0=xmax/2;
// Вычерчивание координатных линий
  Image1->Canvas->MoveTo(0,y0);
  Image1->Canvas->LineTo(xmax,y0);
  Image1->Canvas->MoveTo(x0,0);
  Image1->Canvas->LineTo(x0,ymax);
  Image1->Canvas->Pen->Color=clRed;              // Установка цвета пера
  Image1->Canvas->Pen->Width=2;                  // Установка ширины пера
// Поиск шагов по x и y с масштабированием
hx=(b-a)/xmax;       hy=(y_max-y_min)/ymax;
Image1->Canvas->MoveTo(ceil(x0+a/hx),ceil(y0-f(a)/hy));
for(x=a; x<=b; x+=h)
   Image1->Canvas->LineTo(ceil(x0+x/hx),ceil(y0-f(x)/hy));
}
//-------------------------------------------------------------
  double fun0(double r) {
       return sin(r);
  }
  double fun1(double r) {
```

```
    return r*r;
 }
 double fun2(double r) {
    return r*r*r;
 }
```

Задача 2: Построить домик с использованием стилей заполнения областей. На форму установим только одну компоненту Button и переименуем её в «Построить» (рис.49) и код программы для этой кнопки запишем в следующем виде:

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
    : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
Canvas->TextOut(135,70,"WELCOME");
Canvas->MoveTo(80,90);
Canvas->LineTo(250,90);
Canvas->LineTo(150,40);
Canvas->LineTo(80,90);
Canvas->MoveTo(220,75);
Canvas->LineTo(220,50);
Canvas->LineTo(240,50);
Canvas->LineTo(240,81);
Canvas->Brush->Color=clMaroon;
Canvas->Brush->Style=bsCross;
Canvas->TextOut(240,65,"GOOD BYE");
Canvas->Rectangle(80,90,250,300);
Canvas->Brush->Color=clGreen;
Canvas->Rectangle(120,140,200,200);
}
//---------------------------------------------------------
```

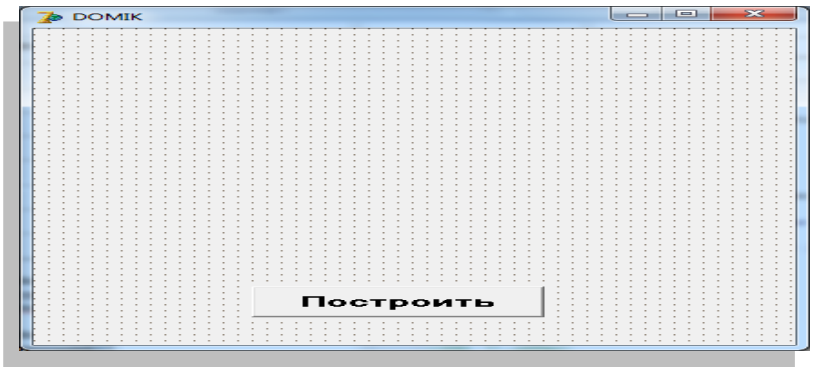


Рис.3**.** Вид Формы для построения изображения

В программе использованы следующие методы:

1. Вычерчивание прямой линии осуществляет метод LineTo, инструкция вызова которого в общем виде выглядит следующим образом:

Canvas->LineTo(x,y)

Метод LineTo вычерчивает прямую линию от текущей позиции карандаша в точку с координатами, указанными при вызове метода. Начальную точку линии можно задать, переместив карандаш в нужную точку графической поверхности. Сделать это можно при помощи метода ***MoveTo***, указав в качестве параметров координаты нового положения карандаша.

2.Вывод текста на Форму осуществляется методом ***Textout*** в нужную точку. При этом можно использовать различные стили и размеры шрифтов. Например,

Font->Size = 20;

Font->Style = [fsltalic, fsBold]

3. Метод ***Rectangle*** вычеркивает прямоугольник с указанными координатами.

В результате получим следующее изображение (рис.50).

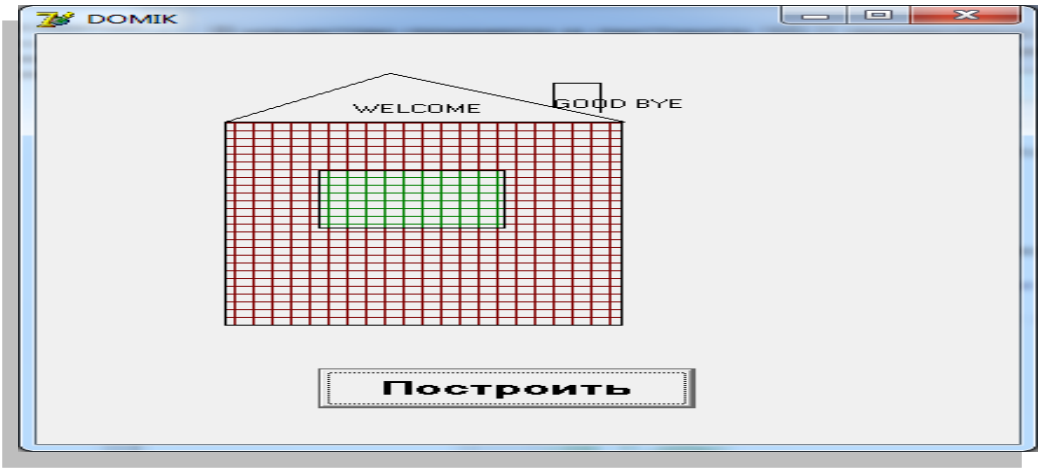


Рис.4 Полученное изображение при выполнении программы

Задача 3. Нарисовать государственный флаг Республики Узбекистан.
На форму установим одну компоненту Button1 и переименуем её в «Нарисовать» и вторую кнопку Button2 и переименуем её в «Выход».
Код программы для этих кнопок запишем в следующем виде:

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package (smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
Canvas->Pen->Color = clBlue;
Canvas->Brush->Color = clBlue;
Canvas->Rectangle(70,50,600,150);
// нарисовать   луну
Canvas->Pen->Color = clBlue;
Canvas->Brush->Color = clWhite;
Canvas->Ellipse(80,55,170,145);
Canvas->Pen->Color = clBlue;
Canvas->Brush->Color = clBlue;
Canvas->Ellipse(105,55,195,145);
 //нарисовать   звезду
Canvas->Font->Name = "Monotype Corsiva";
Canvas->Font->Color = clWhite;
Canvas->Font->Size = 28;
Canvas->TextOut(180,60," ***");
Canvas->TextOut(180,85,"****");
Canvas->TextOut(180,110,"*****");
Canvas->Pen->Color = clRed;
Canvas->Brush->Color = clRed;
Canvas->Rectangle(70,150,600,160);
Canvas->Pen->Color = clWhite;
Canvas->Brush->Color = clWhite;
Canvas->Rectangle(70,160,600,260);
Canvas->Pen->Color = clRed;
Canvas->Brush->Color = clRed;
Canvas->Rectangle(70,260,600,270);
Canvas->Pen->Color = clGreen;
Canvas->Brush->Color = clGreen;
Canvas->Rectangle(70,270,600,370);
```

```
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Form1->Close();
 }
```

В результате получим следующее изображение (рис.27):

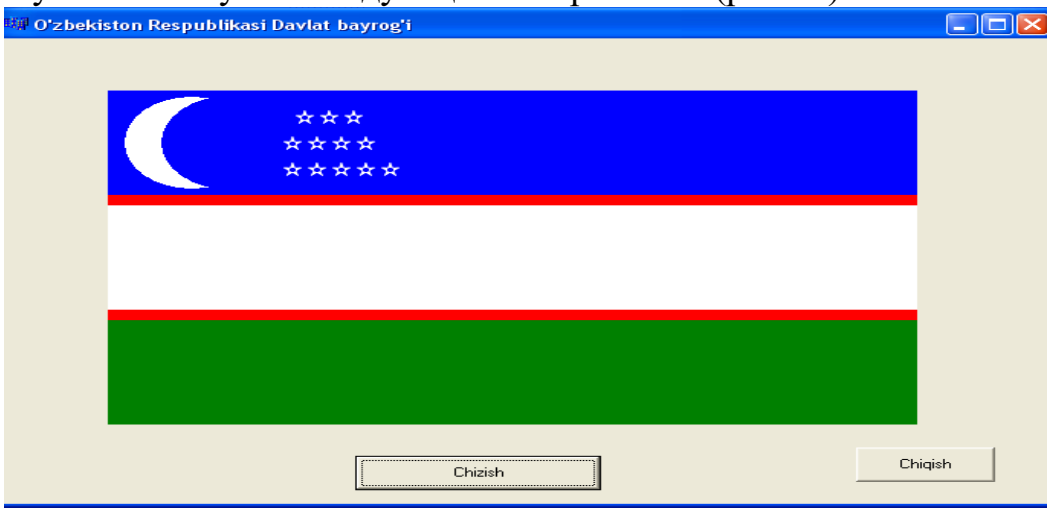


Рис.5 Полученное изображение при выполнении программы.

Контрольные вопросы:

1.Какие классы используются для работы с изображением в Borland C++?
2.Какие компоненты применяются для рисования на канве класс Canvas.
3.Какие методы рисования класса Canvas использовались в программе?
4.Как выбирается и устанавливается цвет объектов в Canvas?

Литература

Основная учебная литература

1. 1.Информатика. Базовый курс. 2-е издание./ Под ред. Симонович С.В.-СПб.,Питер,2005-640с.ил. Учебник для вузов.
2. Кадиров М.М. Ахборот технологиялари. Ўқув қўлланма, 1-қисм. - Т.:Сано-стандарт, 2018. - 320 б.
3. 1.Павловская Т.А. С/С++. Программирование на языке высокого уровня. Учебник для вузов.- СПб.,Питер,2003-464 с.ил
4. Кадиров М.М. Техник тизимларда ахборот технологиялари. Дарслик, 2-қисм. -Т.:Ўзбекистон файласуфлари миллий жамияти, 2019. -306 б.
5. Дадабаева Р.А., Насридинова Ш.Т., Шоахмедова Н.Х., Ибрагимова Л.Т., Ерматов Ш.Т. Ахборот-коммуникация технологиялари ва тизимлари. Ўқув қлланма. -Т.:Сано-стандарт, 2017, - 552 б
6. Кенжабайев А.Т., Икромов М.М., Алланазаров А.Ш. Ахборот-коммуникация технологиялари. Ўқув қўлланма. – Т.: Ўзбекистон файласуфлари миллий жамияти, 2017. - 408 б.
7. Сангин В.Ф., «Комплексная зашита информацсии в корпоративных системах», Учебное пособие. М.: ИД. «ФОРУМ» - ИНФРА М. 2019, 591с.

Дополнительная учебная литература

1. Назиров Ш.А., Обулов Р.В., Бобожонов М.Р., Рахманов .С. С ва С++ тили. Т.:- Ворис-нашриёт, 2013. - 488 б.
2. Кеннет С. Лаудон, Жане. П. Лаудон. Management Information Systems: Managing the Digital Firm, 13th Edition, Pearson Education, USA 2014. p 621.
3. Kunwoo Jee. Principles of CAD/CAM/CAE: The Computer Aided Engineering Design Series. 5th Edition Аддисон Weслей Лонгман, USA, 2015..
4. Алех Alain. Jumping into C++. USA, 2014. p 340.
5. Азимджанова М.Т., Мурадова М.Т., Пазилов М.С. Информатика ва Ахборот технологиялари. Ўқув қўлланма. –Т.: Ўзбекистон файласуфлари миллий жамияти, 2013. -176 б.
6. Арипов М., Доттойев С., Файзийева М. Web технологиялари. Ўқув қўлланма. –Т.: Ўзбекистон файласуфлари миллий жамияти, 2013. - 280 б.
7. Ганиев С.К., Каримов М.М., Ташев К.А. Ахборот хавфсизлиги. Дарслик. – Т.:Фан ва технология, 2017. - 372 б.

Интернет сайты:

1. www.ziyonet.uz – таълим портали.
2. www.lex.uz – Ўзбекистон Республикаси қонун ҳужжатлари маълумотлари миллий базаси.

СОДЕРЖАНИЕ

Редактор Ахметжанова Г.М.