Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Кафедра «Информатика»

Т. А. Трохова, Т. Л. Романькова

ВВЕДЕНИЕ В SCILAB

ПРАКТИКУМ
по курсу «Информатика»
для студентов технических специальностей
дневной и заочной форм обучения

Электронный аналог печатного издания

Гомель 2016

УДК 681.3.6(075.8)
ББК 32.81я73
Т76

Рекомендовано к изданию научно-методическим советом
факультета автоматизированных и информационных систем
ГГТУ им. П. О. Сухого
(*протокол № 8 от 28.03.2016 г.*)

Рецензент: доц., канд. физ.-мат. наук, доц. каф. «Информационные технологии»
ГГТУ им. П. О. Сухого *О. А. Кравченко*

Трохова, Т. А.

Т76 Введение в Scilab : практикум по курсу «Информатика» для студентов техн. специальностей днев. и заоч. форм обучения / Т. А. Трохова, Т. Л. Романькова. – Гомель : ГГТУ им. П. О. Сухого, 2016. – 56 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: https://elib.gstu.by. – Загл. с титул. экрана.

ISBN 978-985-535-321-9.

Практикум содержит краткие теоретические сведения, методику решения типовых задач, набор вариантов заданий для изучения системы компьютерной математики Scilab.

Для студентов технических специальностей дневной и заочной форпм обучения.

УДК 681.3.6(075.8)
ББК 32.81я73

ISBN 978-985-535-321-9

Оглавление

Глава 1. Общая характеристика системы Scilab. Интерфейс и режимы работы Scilab

Scilab – это кроссплатформенная система компьютерной математики (СКМ), которая предназначена для выполнения научно-технических расчетов, графической интерпретации полученных результатов и визуального моделирования. Эта система имеет удобный пользовательский интерфейс и развитый язык программирования.

Scilab свободно распространяется центром Scilab Consortium, с Web-узла www.scilab.org, с которого можно загрузить последнюю версию программы и комплект документации.

Разработка системы Scilab ведется сотрудниками французского Национального института информатики и автоматизации (INRIA – Institut National de Recherche en Informatique et Automatique) с 80-х годов прошлого века. Изначально это был коммерческий проект под названием Blaise, а затем Basile. С 2003 года продукт получил новое имя Scilab и стал бесплатным. Для поддержки Scilab был создан консорциум Scilab Consortium. Сейчас в него входят 25 участников, в том числе Mandriva, INRIA и ENPC (Франция). В настоящее время он распространяется по свободной лицензии CeCILL.

Сама система Scilab, как и Matlab, предназначена прежде всего для численных расчетов и работы с матрицами. Кроме того, она обладает развитыми средствами программирования, так что ее в какой-то мере можно рассматривать как систему разработки высокотехнологичных приложений.

Scilab имеет сходный с Matlab язык программирования. В состав пакета входит утилита, позволяющая конвертировать документы Matlab в Scilab.

Все возможности системы можно классифицировать так:

– математические;
– использования численных методов;
– программирование;
– графические;
– имитационное моделирование;
– сервисные.

Математические возможности перечислены ниже:

– вычисление арифметических и логических выражений;
– вычисление стандартных математических функций;
– операции с векторами и матрицами;
– матричные операции линейной алгебры и т. д.

К численным методам относятся:

– численные методы решения алгебраических уравнений и систем;

– методы работы с полиномами;

– методы решения обыкновенных дифференциальных уравнений и систем;

– методы аппроксимации и интерполяции;

– методы минимизации функций и т. д.

Система имеет несколько режимов работы, каждый из которых поддерживается собственным диалоговым окном (рис. 1.1):

– командный режим – командное окно;

– программный режим – окно создания и редактирования программных файлов (SCE-файлов);

– графический режим – окно редактирования графиков;

– режим помощи – окно помощи;

– режим демонстрации – окон демонстрационных примеров.

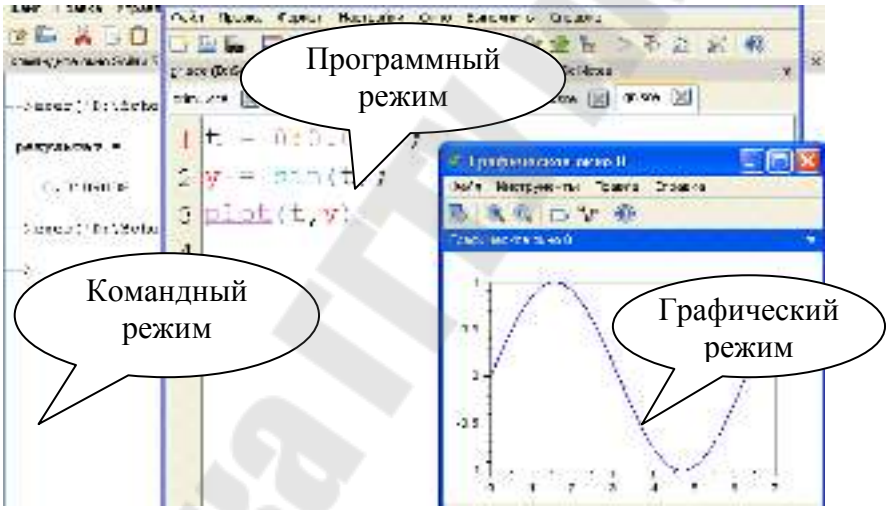


Рис. 1.1. Режимы работы Scilab

При работе в любом из перечисленных режимов могут быть использованы дополнительные информационные окна (рис. 1.2).

Окно рабочей области (Обозреватель переменных) – предназначено для просмотра и редактирования содержимого рабочей области памяти, в нем указывается имя переменной (массива или структуры), ее размерность и тип.

Окно журнала команд содержит перечень команд, введенных пользователем в командном режиме за текущий и предыдущий сеансы работы с системой.

Окно управления файлами (обозреватель файлов) служит для быстрого доступа к файлам при работе с системой.

Управлять информационными окнами можно с использованием пункта основного меню «Инструменты».

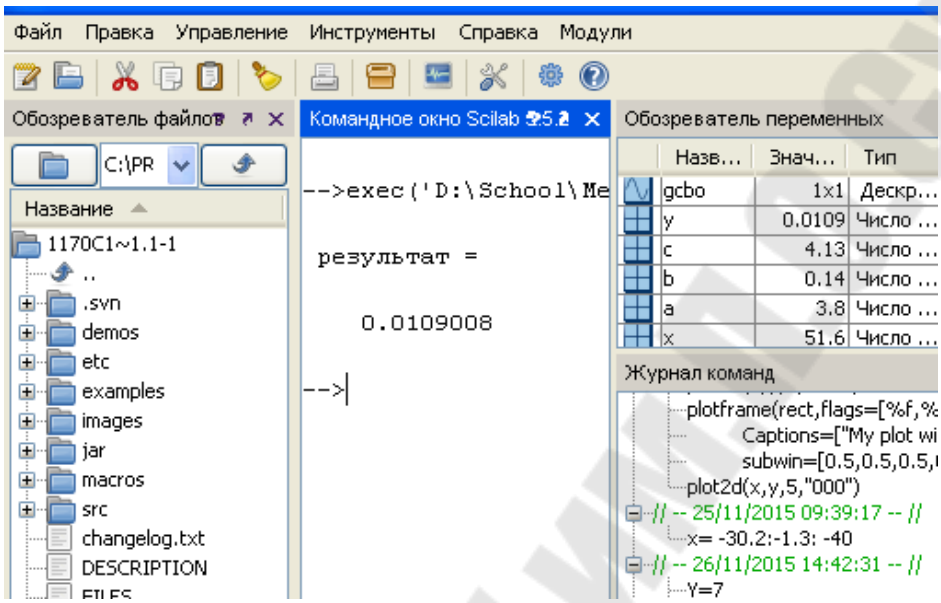


Рис. 1.2. Информационные окна Scilab

Создание документа в системе Scilab может выполняться в различных режимах, ниже рассматриваются командный и программный режимы.

Документ при работе в командном режиме представляет собой последовательность команд пользователя и ответов системы, расположенных в командном окне.

Символ --> в окне команд показывает, что система готова к диалогу с пользователем. Командная строка может содержать одну или несколько команд, она завершается нажатием клавиши Enter. Строка реакции системы называется строкой вывода, она показывает результаты выполнения команды как значение стандартной переменной ответа **ans**, либо переменной, заданной пользователем, например:

```
--> 5+3
ans=
8
--> b=5+3
b=
8
```

Переменная **ans** хранится в памяти и может использоваться в дальнейших вычислениях. Выражения или команды в строке разде-

ляются символами «,» или «;». Результат вычисления выражения или команды, за которой следует запятая, выводится на экран в строку вывода. Результат вычисления выражения или команды, за которой следует точка с запятой, на экран не выводится, но сохраняется в памяти, например:

```
--> x=5, y=x+7;
x=
5
```

Отработанная командная строка не может быть выполнена повторно путем возвращения в нее курсора мыши в командном окне. Для нового выполнения команды она должна быть вызвана из стека команд нажатием клавиш ↑ или ↓. Если выражение не помещается в одной командной строке, то его можно перенести на следующую строку, а предыдущую закончить тремя точками.

В программном режиме пользователь создает программу, которая состоит из команд и выражений системы Scilab и хранится на диске в виде файла с типом **sce**.

Последовательность создания и обработки программы следующая:

Шаг 1. Создать новый файл программы можно с помощью редактора SciNotes командой «**Инструменты – Текстовый редактор SciNotes»** или первой кнопкой на панели инструментов командного окна.

Шаг 2. Записать файл на диск с именем, содержащим тип **sce** с помощью команды «Файл – Сохранить как» программного окна.

Шаг 3. Запустить программу на выполнение с помощью меню «Выполнить» программного окна. Командой «Сохранить и выполнить», кнопкой F5 или кнопкой на панели инструментов программного окна (рис. 1.3).

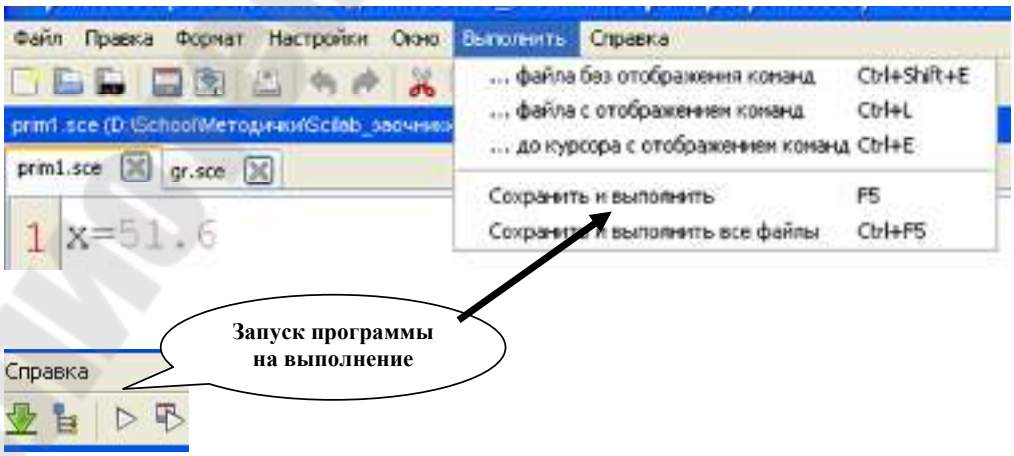


Рис. 1.3. Запуск программы на выполнение

Если компиляция программы прошла успешно, то результаты выполнения программы будут отражены в командном окне.

Если в результате компиляции были найдены ошибки в программе, то необходимо вызвать программу в окно программы, отредактировать и повторить последовательность, начиная с шага 2.

Глава 2. Базовые приемы работы с Scilab

2.1. Базовые вычисления в Scilab

Краткие теоретические сведения

Входной язык системы, т. е. набор правил и символов, на котором пользователь может запрограммировать свой алгоритм и получить результат, представлен данными, выражениями, операторами.

Данные – константы и переменные – записываются по общепринятым правилам. Они делятся на пользовательские и системные.

Основные системные переменные и константы, применяемые в Scilab:

%i или %j – мнимая единица (корень квадратный из –1);

%pi – число π=3,1415926…;

%inf – значение машинной бесконечности;

%e – число e=2.7182818;

ans – результат выполнения последней операции.

Действительные константы могут быть целыми, вещественными с фиксированной и плавающей точкой. Возможно представление чисел в научном формате с указанием мантиссы и порядка числа. Дробная часть отделяется от целой точкой.

Например:

5 -45 – целые константы.

2.301 -897.999 – вещественные константы.

125.35e-25 17.06e-12 – вещественные константы с указанием мантиссы и порядка числа.

Символьная константа представляет собой набор символов, заключенных в двойные или одинарные кавычки. Например: "Курить вредно!", "Задайте элементы массива".

Символ // используется для ввода комментариев к вычислениям.

Например:

```
L=2*pi*r // Вычисление длины окружности
//Программа табулирования функциb
```

Имя переменной (идентификатор) должно соответствовать следующим требованиям:

– должно начинаться с буквы;

– может содержать буквы латинского алфавита, цифры и символ подчеркивания;

– Scilab различает большие и малые буквы.

Примеры допустимых имен переменных:

V1, sp, prim4, sum, rez1.

Выражения в Scilab делятся на арифметические и логические. Арифметическим выражением называется совокупность констант, переменных, стандартных функций, связанных знаками арифметических операций. Арифметические выражения могут содержать круглые скобки

Ниже приведен перечень основных арифметических операций.

Название	**Знак операции**	**Пример**
Сложение	+	x+y
Вычитание	-	x-y
Умножение	*	x*y
Возведение в степень	^	x^5
Обратное (справа налево) деление матриц	\	x\y
Деление матриц слева направо	/	x/y

В Scilab приоритет возведения в степень выше приоритетов умножения и деления, приоритет умножения и деления выше приоритета сложения и вычитания. Для изменения приоритета операций в математических выражениях используются круглые скобки. Степень вложения скобок не ограничивается.

Ниже приводятся некоторые часто употребляемые математические функции.

Описание	**Имя**	**Описание**	**Имя**
Абсолютная величина	abs(x)	Синус	sin(x)
Экспонента	exp(x)	Тангенс	tan(x)
Натуральный логарифм	log(x)	Котангенс	cot(x)

Знак числа	sign(x)	Арккосинус	acos(x)
Десятичный логарифм	log10(x)	Арксинус	asin(x)
Корень квадратный	sqrt(x)	Арктангенс	atan(x)
Косинус	cos(x)	Арккотангенс	acot(x)

Приведем примеры записи арифметических выражений в системе Scilab.

Запись в математике:	Запись в Scilab:
$\frac{\cos x^2}{x+\sin^3 x}+e^{-2.1}$	**cos(x^2)/(x+sin(x)^3)+exp(-2.1)**
$\frac{1.2-2x}{\lg(x+3.2)}-\sqrt{\lvert x-5.8\rvert}$	**(1.2-2*x)/log10(x+3.2)-sqrt(abs(x-5.8))**

Одним из основных операторов в Scilab является оператор присваивания. В программе этот оператор выполняет следующие функции: присваивает переменной, стоящей слева от знака «=» значение выражения, стоящего справа.

Общий вид оператора присваивания:

Имя_переменной = Выражение

В качестве параметра **Имя_переменной** может выступать имя простой переменной, структурированной переменной (вектора, матрицы), имя функции. В качестве параметра **Выражение** применяется арифметическое, логическое или строковое выражение.

Ниже приведен пример правильной записи оператора присваивания.

A = cos(x)+c-d^2*p^2+4.92

В SciLab в качестве оператора ввода используется функция **input**, которую, в силу ее значимости при программировании, принято называть оператором.

Она имеет следующий общий вид:

ИМЯ = input(Символьная константа)

Здесь **ИМЯ** – это имя простой переменной, **Символьная константа** – любой набор символов, заключенный в двойные кавычки. Символьная константа, как правило, разъясняет смысловое назначение вводимой переменной. Например:

```
S=input("Задайте площадь")
A=input("Задайте значение A=")
```

Оператор выполняется следующим образом: в командном окне выводится набор символов, стоящий в скобках после input (символьная константа), выполнение программы приостанавливается, и компьютер переходит в режим ожидания; пользователь вводит константу и введенная константа помещается в оперативной памяти в переменную, стоящую слева в операторе **input**.

При запуске на выполнение программы, содержащей оператор ввода, следует учитывать, что пока пользователь не ввел константу в ответ на запрос своей программы, оператор ввода продолжает свою работу. Система Scilab в это время блокирует выход и закрытие окна рабочего стола.

Если необходимо вывести данные на экран дисплея в определенной последовательности, применяется функция **disp**, которую принято называть оператором вывода. Оператор имеет следующий общий вид:

disp(Выражение)

Здесь **Выражение** – это арифметическое, логическое или символьное выражение, частным случаем которого являются константы или переменные любого типа.

Каждый новый оператор **disp** выполняет вывод с новой строки командного окна, например (переменным **a**, **b**, **k**, **d** уже присвоены числовые значения):

Фрагмент программы	Командное окно
c=a-b+k*d;	**результат=**
disp ("результат="), disp(c);	**28**

Практическая часть

Задание 1. Вычисление арифметических выражений с присваиванием

Постановка задачи. Присвоить значения переменным и вычислить значение арифметического выражения с использованием оператора присваивания.

$y = \left\| \frac{\cos^2 x}{bx - abc} \right\|$	x = 51,6 a = 3,8 b = 0,14 c = 4,13

Последовательность выполнения задания следующая:

Шаг 1. Создать программу и записать ее в редакторе SciNotes.

Шаг 2. Записать программу в файл на диск с именем, содержащим тип **sce**, например, **zadanie1.sce**.

Шаг 3. Запустить программу на выполнение.

Если компиляция программы прошла успешно, то результаты выполнения программы будут отражены в командном окне.

Если в результате компиляции были найдены ошибки в программе, то необходимо вызвать программу в окно программы, отредактировать и повторить последовательность, начиная с п. 2.

Программа	Командное окно
x=51.6 **a=3.8** **b=0.14** **c=4.13** **y=abs(cos(x)^2/(b*x-a*b*c))** **disp("результат =")** **disp(y)**	**результат =** **0.0109008**

Индивидуальные задания приведены в табл. 2.1.

Таблица 2.1

№ п/п	Функция	Значения переменных
1	$y = \sqrt{\ln\left\|\frac{c+b}{c-ab}\right\|} - \cos(abc)$	a = 0,27 b = 2,326 c = 0,8
2	$z = \sqrt[3]{\left\|x+ay\right\|} + \lg\frac{x^2}{y^4}$	x =1,2 y = 0,9 a = -0,3
3	$y = \ln\left(\frac{x\sqrt{2}}{\sin ab}\right)$	x = 13 a = 0,09 b = 1,2
4	$y = \ln\left\|\frac{c+b}{c-ab}\right\| - \cos^2\left(a^3\right)$	a = 0,4 b = -0,6 c = 0,8
5	$y = \text{arctg}(x^{2,5}) - e^a b$	x = 1,26 a = 0,24 b = 7,28
6	$y = \sin^2(xg) - \ln(ag)$	x = 0,2 g = 8,3 a = 1,06
7	$y = \sin^3\frac{a^2}{b+c} - \ln\left\|ab+c^3\right\|$	a = 2,1 b = -1,3 c = 0,8

Окончание табл. 2.1

№ п/п	Функция	Значения переменных
8	$y = \sin^3(2ab) - \ln\left\|\left(c - \frac{b}{a}\right)\right\|$	a = 0,73 b = -1,27 c = 0,27
9	$y = \sin^3 2a + \sqrt{bc} - e^{2a}$	a = 0,3 b = 4,83 c = 2,385
10	$y = \frac{\sin(ab)}{\cos(2-2x)} - e^{xa}$	x = 0,83 a = 1,23 b = 0,438
11	$y = \frac{\sin c \cdot \cos 2x}{\sqrt{a+c}} - \ln(ac)$	x = 0,783 a = 2,6 c = 0,326
12	$y = \sin^3 2a - \cos^2 2b - abc$	a = 2,63 b = 3,81 c = 2,386
13	$y = x\sin 2x - \sqrt{\left\|2\ln\left\|2x\right\|\right\|} + a$	x = 0,78 a = 0,93
14	$y = x\sin(ab) - \frac{\mathrm{tg}\ x}{a^2 - b^2}$	x = 0,62 a = 3,23 b = -0,368
15	$y = \ln\left(\frac{\sqrt{2x}}{a + \ln\left\|b\right\|} - ab\right)$	x = 1,37 a = 0,84 b = -2,648

2.2. Обработка структурированных данных в Scilab

Краткие теоретические сведения

В Scilab можно использовать различные типы структурированных данных, т. е. данных, содержащих несколько элементов. К основным структурированным данным относятся массивы чисел (вектора, матрицы, многомерные массивы).

Массив – это последовательность однотипных элементов, снабженных индексами (порядковыми номерами). Вектором принято называть одномерный массив. Матрица – это двумерный массив. Векторы в Scilab делятся на векторы-строки и векторы-столбцы.

Занести числа в вектор можно несколькими способами.

1. *Непосредственный ввод.*

Чтобы задать вектор-строку, значения его элементов следует перечислить в квадратных скобках, разделяя пробелами или запятыми.

Например,

V=[2 3 8 0]

Чтобы задать вектор-столбец, значения его элементов следует перечислить в квадратных скобках, используя для разграничения строк точку с запятой.

V=[2;3;8]

Обращение к элементу вектора выполняется указанием имени вектора и номера элемента в векторе в круглых скобках.

Например, V(2) → 3

V(3) → 8

2. *Ввод с использованием диапазона*.

Общий вид диапазона:

xn : dx : xk

xn – начальное значение диапазона;

dx – шаг изменения значений диапазона;

xk – конечное значение диапазона.

Для формирования вектора следует задать:

X=xn:dx:xk

В результате будет сформирован вектор, первый элемент которого равен xn, второй – xn+dx, третий – xn+dx+dx и т. д. Последний элемент будет не больше xk для положительного шага dx, и не меньше xk – для отрицательного. Если величина шага отсутствует, то по умолчанию его значение равно 1 или -1, тогда вид диапазона таков:

X=xn : xk

Примеры правильной записи задания вектора с использованием диапазонов приведены ниже.

Фрагмент программы	Командное окно
a=5:2:15 **disp(a)**	**5 7 9 11 13 15**
v=3:5 **disp(v)**	**3 4 5**

Задачу о табулировании функции можно решить с помощью формирования векторов X и Y, например:

X=0:0.5:6.28

Y=cos(X)

В этом случае в каждый элемент вектора Y поместится значение косинуса от каждого элемента вектора X.

Чтобы задать матрицу, значения ее элементов следует перечислить в квадратных скобках, разделяя элементы в строках пробелами или запятыми, а для разграничения строк использовать точку с запятой.

Например:

Фрагмент программы	Командное окно
A=[3 1 -3; 2 0 1.5]	**3. 1. -3.** **2. 0 1.5**

Для указания отдельного элемента матрицы после имени матрицы в круглых скобках указываются два индекса: номер строки и номер столбца.

Например, M(2,3) – это второй по строке и третий по столбцу элемент матрицы M.

Система имеет обширный набор стандартных функций и операций по обработке матриц, который позволяет:

– формировать новые матрицы стандартного вида;

– выполнять матричные арифметические операции;

– вычислять матричные характеристики и математические функции.

Для формирования новых матриц стандартного вида применяются следующие системные функции:

rand(M,N) – формирует прямоугольную матрицу размерностью M×N, элементами которой являются случайные числа в интервале (0.0; 1.0), функция rand без параметров формирует одно случайное число в том же интервале.

ones(M,N) формирует матрицу размерностью M×N, состоящую из единиц.

zeros(M,N) формирует матрицу размерностью M×N, состоящую из нулей.

diag(V) создает диагональную матрицу, в которой элементы вектора V являются элементами главной диагонали.

Матричные арифметические операции представлены следующими:

A+B , A-B матричное сложение и вычитание. Оба операнда этой операции должны иметь одинаковую размерность, если они являются матрицами. Один из операндов может быть скалярной величиной.

A*B матричное умножение. Операция выполняется по правилам матричного умножения, число столбцов матрицы A должно быть равно числу строк матрицы B.

X^P возведение матрицы в степень. Эта операция при скалярном значении P возводит квадратную матрицу X в степень P. Если X – скалярная величина, а P – квадратная матрица, то X^P возводит X в матричную степень P. Эта операция является ошибочной, если оба операнда – матрицы.

В Scilab существуют матричные операции, которые выполняются над каждым элементом матрицы, это такие операции, как:

.* поэлементное матричное умножение;
. поэлементное левое деление матриц;
./ поэлементное правое деление матриц;
.^ поэлементное возведение матрицы в степень.

Оба операнда этих операций должны иметь одинаковую размерность, или один из них должен являться скалярной величиной.

Операция «апостроф» ′ вычисляет транспонированную матрицу.

Ниже приведены примеры выполнения матричных арифметических операций.

Фрагмент программы умножения двух матриц	Командное окно
x=[2 3 4; 5 6 7]	
y=[9 8; 7 6; 5 4]	**59 50**
z=x*y	**122 104**
disp(z)	

Фрагмент программы умножения матрицы на вектор	Командное окно
x=[2 3 4; 5 6 7]	**60**
t=[3; 6; 9]	**114**
z=x*t	
disp(z)	

Поэлементное умножение двух матриц	Командное окно
x=[2 3 4; 5 6 7]	
y=[9 8 7; 6 5 4]	
z=x.*y	**18 24 28**
disp(z)	**30 30 28**

Вычисление транспонированной матрицы	Командное окно
y=[9 8 7; 6 5 4]	**9 6**
w=y'	**8 5**
disp(w)	**7 4**

Система содержит стандартные функции, позволяющие вычислять различные характеристики матриц:

det(A) – вычисление определителя матрицы;

trace(A) – вычисление следа матрицы;
rank(A) – вычисление ранга матрицы;
inv(A) – вычисление обратной матрицы.

Ниже приведены примеры вычисления этих характеристик для матриц A и B.

Фрагмент программы вычисления определителя матрицы	Командное окно
x=[2 3 4; 5 6 7; 1 2 2];	
d=det(x)	**определитель =**
disp("определитель =")	**3**
disp (d)	
Фрагмент программы вычисления обратной матрицы	Командное окно
x=[2 3 4; 5 6 7; 1 2 2];	**-0.6667 0.6667 -1.**
Z=inv(x)	**-1. 0 2.**
disp(Z)	**1.3333 -0.3333 -1.**

Над массивами можно выполнять различные операции, заданные системными функциями.

max(A) – вычисление максимального элемента массива;
min(A) – вычисление минимального элемента массива;
sum(A) – вычисление суммы элементов массива;
prod(A) – вычисление произведения элементов массива;
mean(A) – вычисление среднего значения элементов массива.
[Amax, Nmax]=max(A) – вычисление максимального элемента в массиве и его номера. Порядок применения этих функций и результаты их выполнения рассмотрены на примерах.

Поиск максимального значения матрицы и номера его строки и столбца	Командное окно
y=[2 3 4; 55 6 7; 1 12 2];	**2. 3. 4.**
[ymax,nmax]=max(y)	**55. 6. 7.**
disp(y)	**1. 12. 2.**
disp(ymax)	
disp(nmax)	**55.**
	2. 1.
Фрагмент программы вычисления суммы элементов матрицы	Командное окно
x=[2 3 4; 5 6 7; 1 2 2];	
s=sum(x), disp(s)	**32**

Практическая часть

Задание 1. Создание нескольких векторов с указанными параметрами

Постановка задачи. Создать вектор ***x***, значения которого изменяются от 22 до 37 с шагом 1.

Создать вектор **d**, значения которого изменяются от 2 до 7 с шагом 0,5.

Пример программы по выполнению этого задания приведен ниже.

Программа	Результаты расчетов
x =22:37 **disp("вектор x")** **disp(x)** **d=2:0.5:7** **disp("вектор d")** **disp(d)**	**вектор x** **22. 23. 24. 25. 26. 27. 28. 29. 30. 31. 32. 33. 34. 35. 36. 37.** **вектор d** **2. 2.5 3. 3.5 4. 4.5 5. 5.5 6. 6.5 7.**

Индивидуальные задания приведены в табл. 2.2.

Таблица 2.2

№	Задание
1	Создать вектор x, значения которого изменяются от 22 до 37 с шагом 1. Создать вектор d, значения которого изменяются от 2 до 7 с шагом 0,5
2	Создать вектор x, значения которого изменяются от 32 до 43 с шагом 1. Создать вектор y, значения которого изменяются от 12 до 27 с шагом 0,2
3	Создать вектор x, значения которого изменяются от 3 до 13 с шагом 1. Создать вектор q, значения которого изменяются от 1,2 до 2,7 с шагом 0,05
4	Создать вектор x, значения которого изменяются от 15 до 26 с шагом 1. Создать вектор b, значения которого изменяются от 18,2 до 23,7 с шагом 0,5
5	Создать вектор x, значения которого изменяются от 9 до 18 с шагом 1. Создать вектор u, значения которого изменяются от 6,2 до 13,7 с шагом 0,5
6	Создать вектор x, значения которого изменяются от 42 до 57 с шагом 1. Создать вектор r, значения которого изменяются от 16,2 до 30,7 с шагом 0,5
7	Создать вектор a, значения которого изменяются от 17 до 32 с шагом 1. Создать вектор w, значения которого изменяются от 26,7 до 44,2 с шагом 0,8
8	Создать вектор x, значения которого изменяются от 32 до 47 с шагом 1. Создать вектор d, значения которого изменяются от 12 до 17 с шагом 0,5
9	Создать вектор x, значения которого изменяются от 23 до 40 с шагом 1. Создать вектор y, значения которого изменяются от 2 до 17 с шагом 1,2

Окончание табл. 2.2

№	Задание
10	Создать вектор x, значения которого изменяются от 8 до 23 с шагом 1. Создать вектор q, значения которого изменяются от 13,2 до 22,7 с шагом 0,9
11	Создать вектор x, значения которого изменяются от 25 до 36 с шагом 1. Создать вектор b, значения которого изменяются от 8,2 до 13,7 с шагом 0,5
12	Создать вектор x, значения которого изменяются от 19 до 38 с шагом 1. Создать вектор u, значения которого изменяются от 16,2 до 23,7 с шагом 0,7
13	Создать вектор x, значения которого изменяются от 22 до 37 с шагом 1. Создать вектор r, значения которого изменяются от 26,2 до 43,7 с шагом 0,5
14	Создать вектор a, значения которого изменяются от 27 до 39 с шагом 1. Создать вектор w, значения которого изменяются от 22,7 до 40,2 с шагом 0,9
15	Создать вектор a, значения которого изменяются от 17 до 41 с шагом 1. Создать вектор w, значения которого изменяются от 16,7 до 34,2 с шагом 0,6

Задание 2. Создание одномерного массива как диапазона с заданными пределами изменения

Постановка задачи. Создать одномерный массив как диапазон с заданными пределами изменения. Массив должен содержать не менее 10 чисел. Сформировать новый одномерный массив, содержащий значения функции от элементов исходного массива.

Функция	$x_н$	$x_к$
$y = x^2(4 - x^2)$	0	2

Для выполнения этого задания запрограммируем следующие действия:

Шаг 1. Очистим память с помощью оператора **clear**. Создадим вектор **x**, значения которого будут изменяться от 0 до 2 с шагом 0.2 (вектор создается с использованием диапазона).

Шаг 2. Создадим вектор **y**, каждое значение которого вычисляется с помощью заданной формулы (не забываем про поэлементные операции над векторами).

Шаг 3. Выведем в командное окно числовые значения полученных векторов.

Программа	Результаты расчетов
clear **x=0:0.2:2** **y=(4-x.^2).*x.^2** **disp("вектор x")** **disp(x)** **disp("вектор y")** **disp(y)**	**вектор x** **0. 0.2 0.4 0.6 0.8 1. 1.2 1.4 1.6 1.8 2.** **вектор y** **0. 0.1584 0.6144 1.3104 2.1504 3. 3.6864 3.9984 3.6864 2.4624 0.**

Индивидуальные задания приведены в табл. 2.3.

Таблица 2.3

Номер вари-анта	Функция	X1	X2	Номер вари-анта	Функция	X 1	X 2
1	$y=\frac{\operatorname{arctg}(x)}{1+\sin^2 x}$	2	5	2	$y=\frac{1+\sqrt{0{,}5x}}{0{,}5+\sin^2 x}$	2	4
3	$y=\ln\left(x^2+2x+2\right)$	–3	0	4	$y=e^x\sin x\cos^3 x$	–1	1
5	$y=\frac{\sin^2 x}{\sqrt{x}+x}$	2	5	6	$y=\frac{e^{0,1x}+1}{1+\cos^2 x}$	0	3
7	$y=2x^3-6x^2-18x+7$	–2	2	8	$y=2x^3-3x^2$	1	2
9	$y=x\sqrt{1+x^2}\cdot\sin x$	1	5	10	$y=-\ln\left(x^2-4x+5\right)$	1	4
11	$y=e^x\left(\sin 3x-3\cos 3x\right)$	1	4	12	$y=\left(x^2-2x\right)\ln x-\frac{3}{2}x^2+4x$	1	4
13	$y=\frac{\sin^3 x}{x+2}$	0	3	14	$y=\frac{3x^2+4x+4}{x^2+x+1}$	–1	1
15	$y=\frac{(1+2x)^2}{1.8+\cos^3 x}$	2	4	16	$y=\frac{1}{\ln\left(x^4+4x^3+30\right)}$	–1	3

Задание 3. Обработка матриц и векторов

Постановка задачи. Даны две матрицы A и B.

$$A=\begin{bmatrix} 2 & -1 & -3 \\ 8 & -7 & -6 \\ -3 & 4 & 2 \end{bmatrix} \quad B=\begin{bmatrix} 2 & -1 & -2 \\ 3 & -5 & 4 \\ 1 & 2 & 1 \end{bmatrix}$$

1. Создать вектор V1 из второго столбца матрицы A, вектор V2 из третьей строки матрицы B и вектор V3 из первого столбца матрицы B.

Порядок составления программы следующий:

Шаг 1. Создаем матрицы перечислением элементов по строкам, разделителем между строками является «;».

Шаг 2. Выводим матрицы в командное окно.

Шаг 3. Формируем вектор V1 из всех строк второго столбца матрицы A, поэтому в качестве номера строки матрицы указываем символ «:», что означает «все», а в качестве номера столбца указываем 2. Аналогично формируются вектора V2 и V3.

Шаг 4. Выводим значения векторов в командное окно.

Программа	**Результаты расчетов**
A=[2 -1 -3; 8 -7 -6; -3 4 2] **B=[2 -1 -2; 3 -5 4; 1 2 1]** **disp(A)** **disp(B)**	**2. -1. -3.** **8. -7. -6.** **-3. 4. 2.** **2. -1. - 2.** **3. -5. 4.** **1. 2. 1.**
V1=A(:,2) **V2=B(3,:)** **V3=B(:,1)** **disp(V1)** **disp(V2)** **disp(V3)**	**-1.** **-7.** **4.** **1. 2. 1.** **2.** **3.** **1.**

2. Вычислить V2·V1 и A·V1.

Порядок составления программы таков.

Шаг 1. Выбираем переменные, например, **R1** и **R2**, которые будут являться результатами выполнения операций умножения.

Шаг 2. Так как операции умножения выполняются по законам математики, то точка перед знаком умножения не ставится. Ожидаемый результат умножения двух векторов – одно число, а результат умножения матрицы на вектор – это вектор-столбец из трех элементов.

Шаг 3. Выводим результаты в командную область с помощью оператора **disp**.

Программа	Результаты расчетов
R1=V2*V1 **disp(R1)** **R2=A*V1** **disp(R2)**	**-11.** **-7.** **17.** **-17.**

3. Вычислить $A \cdot B$, A^{-1}, B^{T}.

Порядок составления программы таков.

Шаг 1. Выбираем переменные, например, **W1**, **W2**, **W3**, которые будут являться результатами выполнения операций.

Шаг 2. Так как операции умножения матриц выполняются по законам математики, то точка перед знаком умножения не ставится. Ожидаемый результат умножения двух матриц – матрица из трех строк и трех столбцов.

Шаг 3. Вычисляем обратную матрицу с помощью стандартной функции **inv**, выводим результат в командную область с помощью оператора **disp**.

Шаг 4. Вычисляем транспонированную матрицу с помощью стандартной операции «апостроф» – B', выводим результат в командную область с помощью оператора **disp**.

Программа	Результаты расчетов
W1=A*B **disp(W1)** **W2=inv(A)** **disp(W2)** **W3=B'** **disp(W3)**	**-2. -3. -11.** **-11. 15. -50.** **8. -13. 24.** **-0.6666667 0.6666667 1.** **-0.1333333 0.3333333 0.8** **-0.7333333 0.3333333 0.4** **2. 3. 1.** **-1. - 5. 2.** **-2. 4. 1.**

4. Вычислить определители A и B.

Порядок составления программы следующий:

Шаг 1. Выбираем переменные, например, **S1**, **S2**, которые будут являться результатами выполнения операций.

Шаг 2. Определители матриц вычисляем в программе с помощью стандартной функции **det**. Выводим результат в командную область с помощью оператора **disp**.

Программа	Результаты расчетов
S1=det(A) **S2=det(B)** **disp(S1)** **disp(S2)**	 **-15.** **-49.**

5. Вычислить вектор V3 поэлементным умножением векторов V1 и V3.

6. Вычислить матрицу D поэлементным умножением матриц A и B.

Порядок составления программы таков.

Шаг 1. Выбираем переменные, например, **Z1**, **Z2**, которые будут являться результатами выполнения операций.

Шаг 2. Так как операция умножения двух векторов выполняется поэлементно, то перед знаком умножения ставится точка. Ожидаемый результат поэлементного умножения двух векторов – вектор из трех элементов.

Шаг 3. Так как операция умножения двух матриц выполняется поэлементно, то перед знаком умножения ставится точка. Ожидаемый результат поэлементного умножения двух матриц – матрица из трех строк и трех столбцов.

Программа	Результаты расчетов
Z1=V1.*V3 **disp(Z1)** **Z2=A.*B** **disp(Z2)**	**-2.** **-21.** **4.** **4. 1. 6.** **24. 35. -24.** **-3. 8. 2.**

Индивидуальные варианты приведены в табл. 2.4.

Таблица 2.4

1	$A=\begin{bmatrix} 2 & -1 & -3 \\ 8 & -7 & -6 \\ -3 & 4 & 2 \end{bmatrix}$ $B=\begin{bmatrix} 2 & -1 & -2 \\ 3 & -5 & 4 \\ 1 & 2 & 1 \end{bmatrix}$	2	$A=\begin{bmatrix} 3 & 5 & -6 \\ 2 & 4 & 3 \\ -3 & 1 & 1 \end{bmatrix}$ $B=\begin{bmatrix} 2 & 8 & -5 \\ -2 & -1 & 0 \\ 4 & 5 & -3 \end{bmatrix}$
3	$A=\begin{bmatrix} 2 & 1 & -1 \\ 2 & -1 & 1 \\ 1 & 0 & 2 \end{bmatrix}$ $B=\begin{bmatrix} 3 & 6 & 0 \\ 2 & 4 & -6 \\ 1 & -2 & 3 \end{bmatrix}$	4	$A=\begin{bmatrix} -6 & 1 & 11 \\ 9 & 2 & 5 \\ 0 & 3 & 7 \end{bmatrix}$ $B=\begin{bmatrix} 3 & 0 & 1 \\ 0 & 2 & 7 \\ 1 & -3 & 2 \end{bmatrix}$
5	$A=\begin{bmatrix} 3 & 1 & 2 \\ -1 & 0 & 2 \\ 0 & 2 & 1 \end{bmatrix}$ $B=\begin{bmatrix} 0 & -1 & 2 \\ 2 & 1 & 1 \\ 3 & 7 & 1 \end{bmatrix}$	6	$A=\begin{bmatrix} 2 & 3 & 2 \\ 1 & 3 & -1 \\ 4 & 1 & 3 \end{bmatrix}$ $B=\begin{bmatrix} 3 & 2 & -1 \\ 0 & 1 & 2 \\ 5 & 3 & 0 \end{bmatrix}$
7	$A=\begin{bmatrix} 6 & 7 & 3 \\ 3 & 1 & 0 \\ 2 & 2 & 1 \end{bmatrix}$ $B=\begin{bmatrix} 2 & 0 & 5 \\ 4 & -1 & -2 \\ 4 & 3 & 7 \end{bmatrix}$	8	$A=\begin{bmatrix} -2 & 3 & 4 \\ 3 & -2 & -4 \\ -1 & 2 & 2 \end{bmatrix}$ $B=\begin{bmatrix} 3 & 3 & 1 \\ 0 & 6 & 2 \\ 1 & 9 & 2 \end{bmatrix}$
9	$A=\begin{bmatrix} 1 & 7 & 3 \\ -4 & 9 & 4 \\ 0 & 3 & 2 \end{bmatrix}$ $B=\begin{bmatrix} 6 & 5 & 2 \\ 1 & 9 & 2 \\ 4 & 5 & 2 \end{bmatrix}$	10	$A=\begin{bmatrix} 2 & 6 & 1 \\ 1 & 3 & 2 \\ 0 & 1 & 1 \end{bmatrix}$ $B=\begin{bmatrix} 4 & -3 & 0 \\ -4 & 0 & 5 \\ 3 & 2 & -3 \end{bmatrix}$
11	$A=\begin{bmatrix} 6 & 9 & 4 \\ -9 & -1 & 1 \\ 10 & 1 & 7 \end{bmatrix}$ $B=\begin{bmatrix} 1 & 1 & 1 \\ 3 & 4 & 3 \\ 0 & 5 & 2 \end{bmatrix}$	12	$A=\begin{bmatrix} 1 & 0 & 3 \\ 3 & 1 & 7 \\ 2 & 1 & 8 \end{bmatrix}$ $B=\begin{bmatrix} 3 & 5 & 4 \\ -3 & 0 & 1 \\ 5 & 6 & -4 \end{bmatrix}$
13	$A=\begin{bmatrix} 5 & 1 & -2 \\ 1 & 3 & -1 \\ 8 & 4 & -1 \end{bmatrix}$ $B=\begin{bmatrix} 3 & 5 & 5 \\ 7 & 1 & 2 \\ 1 & 3 & 0 \end{bmatrix}$	14	$A=\begin{bmatrix} 2 & 2 & 5 \\ 3 & 3 & 6 \\ 4 & 3 & 4 \end{bmatrix}$ $B=\begin{bmatrix} 1 & -1 & 1 \\ 2 & 3 & 3 \\ 1 & -2 & -1 \end{bmatrix}$
15	$A=\begin{bmatrix} 1 & -2 & 5 \\ 3 & 0 & 6 \\ 4 & 3 & 4 \end{bmatrix}$ $B=\begin{bmatrix} -1 & 1 & 1 \\ 0 & 3 & 3 \\ 1 & -3 & -1 \end{bmatrix}$	16	$A=\begin{bmatrix} 5 & 4 & 2 \\ 1 & 2 & 4 \\ 3 & 0 & 5 \end{bmatrix}$ $B=\begin{bmatrix} 5 & 4 & -5 \\ 3 & -7 & 7 \\ 1 & 2 & 2 \end{bmatrix}$

Задание 4. Вычисление суммы и произведения

Постановка задачи. Вычислить сумму и произведение элементов по заданным формулам.

$\sum_{i=1}^{8}[(i+3)\times(i-6)]$	$\prod_{i=1}^{5}(i+2i-1)$

Порядок составления программы таков.

Шаг 1. Создаем вектор, элементы которого будут изменяться от нижнего предела суммирования до верхнего, например, от 1 до 8.

Шаг 2. Создаем вектор **F**, каждый элемент которого вычисляется по формуле, стоящей под знаком суммирования.

Шаг 3. Находим сумму элементов этого вектора с помощью стандартной функции **sum**, результат помещаем в переменную **SS** и выводим ее в командную область с помощью оператора **disp**.

Шаг 4. Для вычисления произведения проделываем ту же самую последовательность шагов, только воспользуемся стандартной функцией ***prod***.

Программа	Результаты расчетов	Программа	Результаты расчетов
i=1:8 **F=(i+3).*(i-6)** **SS=sum(F)** **disp(SS)**	**-48.**	**i=1:5** **F1=(i+2*i-1)** **PP=prod(F1)** **disp(PP)**	**12320.**

Индивидуальные варианты приведены в табл. 2.5.

Таблица 2.5

1	$\sum_{i=1}^{8}[(i+3)\cdot(i-6)]$	1	$\prod_{i=1}^{5}(i+2i-1)$
2	$\sum_{i=1}^{10}[i\cdot(i-1)]$	2	$\prod_{i=1}^{6}\frac{(1)^i}{i-5.5}$
3	$\sum_{i=1}^{7}[i\cdot(3i-1)]$	3	$\prod_{i=1}^{8}\frac{i+1}{i-3.5}$
4	$\sum_{i=1}^{12}\frac{i-6}{i^2}$	4	$\prod_{i=1}^{9}\frac{i-10}{25-i}$
5	$\sum_{i=3}^{13}\frac{i+6}{i-2}$	5	$\prod_{i=1}^{7}\frac{i^2-40}{(-1)^i}$
6	$\sum_{i=5}^{16}\frac{(-1)^i}{(i-4)^2}$	6	$\prod_{i=3}^{6}[i\cdot(i-1)]$
7	$\sum_{i=1}^{10}\frac{(-1)^{-i}}{(i+1)^2}$	7	$\prod_{i=4}^{10}\frac{i^2}{i+3}$
8	$\sum_{i=1}^{12}\frac{(i+4)^2}{(i+3)^3}$	8	$\prod_{i=5}^{10}\frac{i^3}{(i-1)}$

Окончание табл. 2.5

9	$\sum_{i=1}^{11}\frac{(i-0.5)\cdot(i+1.5)}{(i+1)^2}$	9	$\prod_{i=1}^{7}\frac{i^2}{i^3-6.2}$
10	$\sum_{i=1}^{12}\frac{(i+6)\cdot(i-3)}{(i+3)^3}$	10	$\prod_{i=2}^{7}\frac{i^2-15}{i^2+1}$
11	$\sum_{i=3}^{16}\frac{(-1)^{i+1}}{(i-2)^3}$	11	$\prod_{i=3}^{7}[(i-1)\cdot(i+2)]$
12	$\sum_{i=6}^{14}\frac{(-1)^i}{i}$	12	$\prod_{i=5}^{8}\frac{(-1)^i}{i^2-i+3}$
13	$\sum_{i=8}^{17}\frac{1}{(i-1)\cdot(i-7)}$	13	$\prod_{i=1}^{5}\frac{(i^2-0,5)}{(i^2+0,5)}$
14	$\sum_{i=2}^{10}[\frac{1}{i-1}+1.5i]$	14	$\prod_{i=6}^{12}\frac{(2-i)}{25-i^2}$
15	$\sum_{i=1}^{10}[\frac{1}{4}+\frac{1}{i\cdot(i+1)}]$	15	$\prod_{i=1}^{8}\frac{i^2+16i-3}{(i+6)^2}$

Задание 5. Решение задач с матрицами

Постановка задачи. Дана матрица Y(3×4). Вычислить сумму элементов третьей строки и произведение элементов второго столбца матрицы Y. Вычислить максимальный элемент матрицы Y.

Порядок составления программы таков.

Шаг 1. Создаем матрицу, элементы которой подберем самостоятельно. Выведем матрицу в командную область.

Шаг 2. С помощью стандартной функции sum найдем сумму элементов третьей строки, для этого укажем **Y(3,:)**, что означает – третья строка и все столбцы в ней. Поместим сумму в переменную **S** и выведем результат в командную область.

Шаг 3. Для вычисления произведения проделываем ту же самую последовательность шагов, только воспользуемся стандартной функцией ***prod*** и выбираем второй столбец матрицы – **Y(:,2)**. Поместим произведение в переменную **S1** и выведем ее в командную область.

Шаг 4. Для вычисления максимума воспользуемся стандартной функцией max, и так как нужно вычислить максимум среди всех элементов матрицы, то номер строки и столбца не указываем. Поместим результат в переменную **M** и выведем его в командную область.

Программа	Результаты расчетов
Y=[2 -1 -3 8; -7 -6 -3 4; 2 5 8 2] **disp(Y)** **S=sum(Y(3,:))** **disp(S)** **S1=prod(Y(:,2))** **disp(S1)** **M=max(Y)** **disp(M)**	**2. -1. - 3. 8.** **-7. -6. - 3. 4.** **2. 5. 8. 2.** **17.** **30.** **8.**

Индивидуальные варианты приведены в табл. 2.6.

Таблица 2.6

№ п/п	Дана матрица	Вычислить
1	Y(3×4)	Вычислить сумму элементов третьей строки и произведение элементов второго столбца матрицы Y. Вычислить максимальный элемент матрицы Y
2	P(4×3)	Вычислить сумму элементов второй строки и произведение элементов третьего столбца матрицы P. Вычислить максимальный элемент матрицы P
3	M(4×4)	Вычислить сумму элементов последней строки и произведение элементов первого столбца матрицы M. Вычислить минимальный элемент матрицы M
4	A(3×3)	Вычислить сумму элементов второй строки и произведение элементов третьего столбца матрицы A. Вычислить минимальный элемент матрицы A
5	Y(3×4)	Вычислить сумму элементов второй строки и второго столбца матрицы. Вычислить минимальный элемент матрицы Y
6	P(4×3)	Вычислить произведение элементов третьей строки и третьего столбца матрицы P. Вычислить максимальный элемент матрицы P
7	M(4×4)	Вычислить сумму элементов последней строки и первого столбца матрицы M. Вычислить максимальный элемент матрицы M
8	A(3×3)	Вычислить произведение элементов второй строки и третьего столбца матрицы A. Вычислить максимальный элемент матрицы A

Окончание табл. 2.6

№ п/п	Дана матрица	Вычислить
9	Z(4×3)	Вычислить среднее арифметическое элементов второго столбца матрицы Z. Вычислить минимальный элемент матрицы Z
10	P(4×5)	Вычислить среднее арифметическое элементов второй строки матрицы P. Вычислить минимальный элемент матрицы P
11	M(4×4)	Вычислить сумму элементов последней строки и произведение элементов первого столбца матрицы M. Вычислить сумму элементов матрицы M
12	A(3×3)	Вычислить сумму элементов второй строки и минимальный элемент третьего столбца матрицы A. Вычислить произведение элементов матрицы A
13	Y(3×4)	Вычислить максимальный элемент второй строки и произведение элементов второго столбца матрицы Y. Вычислить сумму элементов матрицы Y
14	P(4×3)	Вычислить сумму элементов третьей строки и максимальный элемент третьего столбца матрицы P. Вычислить произведение элементов матрицы P
15	M(4×4)	Вычислить сумму элементов последней строки и минимальный элемент первого столбца матрицы M. Вычислить произведение элементов матрицы M

2.3. Программирование в Scilab

Краткие теоретические сведения

Для программирования разветвляющихся алгоритмов в Scilab существуют оператор условия и оператор выбора.

Условный оператор представлен в нескольких формах и имеет следующий общий вид.

Полная форма 1

```
if   ЛВ then
     ОПЕРАТОР1
          else   ОПЕРАТОР2,
end
```

Полная форма 2

```
if ЛВ then ОПЕРАТОР1,
elseif  ЛВ2 then ОПЕРАТОР2,
```

else ОПЕРАТОР,
end

Краткая форма

if ЛВ then ОПЕРАТОР1
end

Здесь

ЛВ, ЛВ1, ЛВ2, ЛВ3 – логические выражения;

ОПЕРАТОР1, ОПЕРАТОР2, ОПЕРАТОР– любые операторы или группы операторов языка Scilab.

Разделителями в операторах могут быть запятая или точка с запятой. Если ОПЕРАТОР расположен в следующей строке, то разделитель перед ним ставить не обязательно.

Порядок выполнения оператора условия следующий.

Полная форма 1

• Вычисляется значение логического выражения.

• Если логическое выражение истинно, то выполняется оператор, стоящий после слова **then**, а затем следующий за оператором **if** оператор.

• Если логическое выражение ложно, то выполняется оператор, стоящий после слова **else**, а затем следующий за оператором **if** оператор.

Краткая форма

• Вычисляется значение логического выражения.

• Если логическое выражение истинно, то выполняется оператор, стоящий после слова **then**, а затем следующий за оператором **if** оператор.

• Если логическое выражение ложно, то выполняется следующий за оператором if оператор.

При программировании логических выражений используются следующие операции отношения и логические операции.

Название	**Знак операции**	**Пример**
Равно	==	x==y
Не равно	~=	x~=y
Меньше	<	x<y
Больше	>	x>y
Меньше или равно	<=	x<=y
Больше или равно	>=	x>=y
Логическое И	&	x&y
Логическое ИЛИ	|	x|y
Логическое НЕ	~	~x

В простых логических выражениях используется знак операции отношения, например: **a > b** или **c <= 5.8**. В сложных логических выражениях используются знаки логических операций, например:

a>b | b<0
(a > 0) & (b>0)

Ниже приведен фрагмент программы, в котором с помощью оператора **if** в командную область выводится либо символьная константа **"корни мнимые"** либо **"корни вещественные"** в зависимости от того отрицателен ли результат арифметического выражения **b^2-4*a*c.**

```
if   b^2-4*a*c<0 then
                disp("корни мнимые")
                        else
               disp("корни вещественные")
end
```

Операторы цикла предназначены для программирования циклических алгоритмов, они изменяют естественный ход выполнения программы и относятся к операторам управления. Операторы являются взаимозаменяемыми, выбор для применения того или иного оператора зависит от программиста.

Операторы цикла в Scilab можно классифицировать следующим образом:

– оператор цикла с параметрами for;

– оператор цикла с предусловием while.

Оператор цикла **for** предназначен для программирования циклических алгоритмов, когда переменная цикла явно выражена и изменяется от начального значения до конечного значения с постоянным шагом.

Общий вид оператора цикла for:

```
   for   x= xn : dx : xk
                    ОПЕРАТОР,
                   end
```

Здесь:

x – переменная цикла;

xn – начальное значение переменной цикла;

dx – шаг изменения переменной цикла;

xk – конечное значение переменной цикла;

ОПЕРАТОР – любой оператор или группа операторов рабочей части цикла.

Порядок выполнения оператора цикла:

– проверяется условие **xn<=xk;**

– если условие не выполняется, то оператор цикла прекращает свою работу, и выполняется следующий за ним оператор программы;

– если условие выполняется, то переменной цикла присваивается ее начальное значение **x=xn;**

– выполняется оператор рабочей части цикла;

– до тех пор, пока **x< = xk**, переменная цикла увеличивается на шаг **dx** и выполняется рабочая часть цикла.

Количество повторений цикла вычисляется по формуле

$$|xk - xn|/dx + 1.$$

Примеры правильной записи оператора for:

Фрагмент программы	Командное окно
for k=1:5 **disp(k)** **end**	**1** **2** **3** **4** **5**

Оператор while предназначен для программирования любых циклов, где проверка условия повторения цикла выполняется перед выполнением рабочей части цикла.

Общий вид:

while ЛВ,
ОПЕРАТОР,
end

Здесь:

ЛВ – логическое выражение;

ОПЕРАТОР – любой оператор или группа операторов рабочей части цикла.

Порядок выполнения оператора следующий:

– проверяется истинность логического выражения;

– до тех пор, пока оно истинно, выполняется оператор рабочей части цикла;

– если логическое выражение стало ложным, то выполняется следующий за оператором цикла оператор программы.

Примеры правильной записи оператора цикла while приведены ниже.

Вывести на экран дисплея четные целые числа от 2 до 12.

```
k=2
while k<=20
      disp(k)
      k=k+2
end
```

Практическая часть

Задание 1. Программирование разветвляющихся алгоритмов

Постановка задачи. Разработать программу для формирования функции **y(x)**. Подобрать набор тестов для проверки правильности работы программного фрагмента. Вычислить значения функции **y(x)** для тестовых значений ***x***.

Вид функции
$y = \begin{cases} \sqrt{x}, & \text{если} \quad x > 20 \\ x, & \text{если} \quad 1 \le x \le 20 \\ 4x^2 & \text{в остальных случаях} \end{cases}$

Порядок составления программы следующий:

Шаг 1. Программируем ввод исходного данного – переменной **x** с использованием оператора ввода **input**.

Шаг 2. Проверяем, выполняется ли условие x > 20, и если оно выполнится, то программируем вычисление **y=sqrt(x)**.

Шаг 3. Продолжаем оператор **if**, описывая, что будет в случае, если первое условие не выполняется. Для этого воспользуемся полной формой 2 и после **elseif** формируем второе условие в виде сложного логического выражения.

Шаг 4. Завершаем оператор **if**, указывая после **else**, чему равен результат в остальных случаях. Завершаем оператор **if** словом **end**.

Шаг 5. Выводим полученный результат в командную область.

Программа	Результаты расчетов
x=input("Задайте x=") **if x>20 then y=sqrt(x),** **elseif (x>=1) & (x<=20) then y=x** **else y=4*x^2** **end** **disp('y=')** **disp(y)**	**Задайте x=25** **y=** **5.** **Задайте x=15** **y=** **15.** **Задайте x=-3** **y=** **36.**

Индивидуальные варианты приведены в табл. 2.7.

Таблица 2.7

Номер варианта	Вид функции	Номер варианта	Вид функции
1	$y=\begin{cases}\sqrt{x}, & \text{если } x>20\\ x, & \text{если } 1\le x\le 20\\ 4x^2 & \text{в остальных случаях}\end{cases}$	2	$y=\begin{cases}2x^2, & \text{если } x>10\\ -20x, & \text{если } x\le 2\\ 4x & \text{в остальных случаях}\end{cases}$
3	$y=\begin{cases}\sqrt{x}, & \text{если } x>16\\ \dfrac{10}{x}, & \text{если } 1\le x\le 16\\ x+2 & \text{в остальных случаях}\end{cases}$	4	$y=\begin{cases}\sin 2x, & \text{если } x>20\\ \sqrt{x}, & \text{если } 1\le x\le 20\\ \cos x & \text{в остальных случаях}\end{cases}$
5	$y=\begin{cases}\dfrac{1}{x}, & \text{если } x>10\\ \sqrt[3]{x}, & \text{если } 1\le x\le 8\\ \dfrac{x}{5} & \text{в остальных случаях}\end{cases}$	6	$y=\begin{cases}\sqrt{x}+1, & \text{если } 0\le x\le 3\\ \dfrac{1}{x}, & \text{если } x\ge 5\\ 3x^2-2 & \text{в остальныхслучаях}\end{cases}$
7	$y=\begin{cases}\sqrt{x}, & \text{если } x>2\\ \dfrac{x^2}{2}, & \text{если } -10\le x\le -3\\ 2x & \text{в остальных случаях}\end{cases}$	8	$y=\begin{cases}15\sin(x), & \text{если } x>3\\ 50\cos(x), & \text{если } x\le 0\\ x^2+2 & \text{в остальных случаях}\end{cases}$

Окончание табл. 2.7

9	$y=\begin{cases}5\sqrt{x}, & \text{если } x>10\\ 10\sqrt[3]{x}, & \text{если } 1\le x\le 9\\ \dfrac{x^2}{2} & \text{в остальных случаях}\end{cases}$	10	$y=\begin{cases}\dfrac{x^2}{50}, & \text{если } x>8\\ -2x, & \text{если } x\le 0\\ \sqrt{x} & \text{в остальных случаях}\end{cases}$
11	$y=\begin{cases}8x, & \text{если } x>5\\ x^2, & \text{если } -5\le x\le 5\\ \sin(x) & \text{в остальныхслучаях}\end{cases}$	12	$y=\begin{cases}10\cos(x), & \text{если } x>8\\ \dfrac{\sin(x)}{0{,}5}, & \text{если } 1\le x\le 8\\ \dfrac{x^2}{100} & \text{в остальных случаях}\end{cases}$
13	$y=\begin{cases}1-3x, & \text{если } x>5\\ x-5\sin(x), & \text{если} -5\le x\le 5\\ \dfrac{x^2}{10} & \text{в остальных случаях}\end{cases}$	14	$y=\begin{cases}\dfrac{x^2}{10}, & \text{если } x<10\\ \sqrt[3]{\lvert(x)\rvert}, & \text{если } x>15\\ 5 & \text{в остальных случаях}\end{cases}$
15	$y=\begin{cases}\sqrt{x}, & \text{если } x>5\\ \dfrac{1}{x}, & \text{если } -10\le x\le -1\\ \cos(x) & \text{в остальных случаях}\end{cases}$	16	$y=\begin{cases}\sqrt{x+4}, & \text{если } x>20\\ \lg(x), & \text{если } 1\le x\le 20\\ x & \text{в остальных случаях}\end{cases}$

Задание 2. Программирование циклических алгоритмов

Постановка задачи. Задан вектор произвольной длины **N**. Разработать программу для вычисления результата по индивидуальному заданию. Проверить работу программы на тестовом примере (числовые значения вектора подобрать самостоятельно).

Вычислить произведение отрицательных элементов вектора с четными номерами

Порядок составления программы следующий:

Шаг 1. Программируем ввод исходного данного – вектора **X** с использованием оператора присваивания. Числа в векторе подбираем таким образом, чтобы встречались и отрицательные и положительные числа. Программа должна выделить отрицательные числа и найти их произведение.

Шаг 2. Для вычисления произведения выбираем переменную **P**, первоначально присваиваем ей значение 1.

Шаг 3. Для того чтобы просмотреть элементы, имеющие в векторе четные номера, например, 2, 4, 6 и т. д., организуем цикл с переменной цикла **i**. Этой переменной первоначально присвоим значение 2, т. е. будем первоначально рассматривать второй элемент вектора.

Шаг 4. Цикл должен выполняться до тех пор, пока текущий номер элемента в векторе (переменная цикла **i**) не больше общего количества элементов. В данном векторе 10 элементов, поэтому заголовок цикла имеет вид: **while i<=10**.

Шаг 5. В цикле используется оператор **if**, чтобы проверить, является ли элемент вектора **X** отрицательным, т. е. меньшим нуля. Если это условие выполняется, то переменную **P** умножаем на этот элемент и результат помещаем в ту же самую переменную **P.** Завершаем оператор **if** словом **end**.

Шаг 6. В цикле изменяем переменную цикла на 2 (i=i+2) и заканчиваем цикл словом **end**.

Шаг 7. Выводим элементы исходного вектора и полученный результат в командную область.

Программа	**Результаты расчетов**
X=[-2 -3 -6 5 2 -4 8 9 -10 -7] **P=1; i=2** **while i<=10** **if X(i)<0 then P=P*X(i)** **end** **i=i+2** **end** **disp(X); disp(P)**	**-2. -3. -6. 5. 2. -4. 8. 9. -10. -7.** **-84.**

Индивидуальные варианты приведены в табл. 2.8.

Таблица 2.8

Номер варианта	**Условие задачи**
1	Подсчитать количество элементов вектора, по значению принадлежащих интервалу [3,1…5,4]
2	Подсчитать количество отрицательных элементов вектора
3	Подсчитать сумму чисел, по значению меньших последнего элемента вектора
4	Подсчитать произведение чисел, по значению больших 7

Окончание табл. 2.8

5	Подсчитать среднее арифметическое отрицательных элементов вектора
6	Подсчитать количество элементов вектора, по значению больших 5 и стоящих на нечетных местах
7	Подсчитать сумму квадратов положительных элементов вектора, стоящих на четных местах
8	Подсчитать количество элементов вектора, больших первого элемента
9	Подсчитать сумму элементов вектора, по значению принадлежащих интервалу [2,5…8,4]
10	Подсчитать количество чисел, больших последнего элемента вектора
11	Вычислить произведение положительных элементов вектора
12	Подсчитать сумму элементов вектора, больших 12 и стоящих на четных местах
13	Подсчитать произведение элементов вектора, меньших 4 и стоящих на нечетных местах
14	Подсчитать сумму отрицательных и стоящих на четных местах элементов вектора
15	Подсчитать среднее арифметическое положительных элементов вектора

Глава 3. Графика и численные методы в Scilab

3.1. Построение графиков

Краткие теоретические сведения

Графические объекты в Scilab строятся в специальном графическом окне (figure). Одновременно может быть открыто несколько таких окон, каждому из которых присваивается номер. Для перехода к имеющемуся окну с номером N или открытия нового графического окна необходимо ввести команду **figure(N)**. Кроме того, первое обращение к графической команде автоматически вызывает появление графического окна, которому присваивается номер 0.

Для построения графиков функций одной переменной в декартовой системе координат используются различные формы команды **plot**, которая рисует графики функций по ряду точек, соединяя их отрезками прямых.

Команда **plot(X,Y)** – строит график функции, координаты точек которой берутся из векторов одинаковой размерности X и Y. Если Y – матрица, то строится семейство графиков по данным, содержащимся в столбцах матрицы.

Например, для построения графика функции **y=sin(t)** нужно задать следующий фрагмент программы.

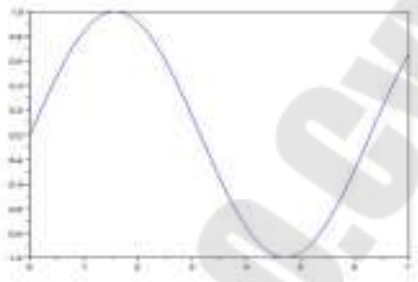
Фрагмент программы

```
t=0:0.01:7;
y=sin(t);
plot(t,y)
```

или

```
t=0:0.01:7;
plot(t,sin(t))
```

Графическое окно

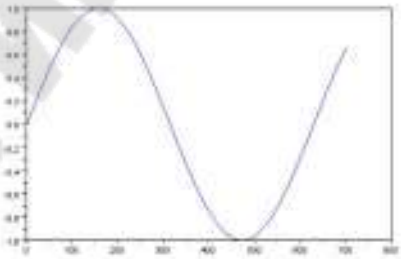


Команда **plot(Y)** – строит график зависимости, значения ординат которой берутся из вектора Y, а значения абсцисс представляют собой индексы соответствующих элементов вектора.

Фрагмент программы

```
t=0:0.01:7;
y=sin(t);
plot(y)
```

Графическое окно



Для построения графиков двух функций – **sin(x)** и **cos(x)**, значения функции которых содержатся в векторах y1 и y2, а значения аргумента **x** хранятся в векторе **x**, нужно задать:

Фрагмент программы

```
x=0:0.01:6
y1=sin(x)
y2=cos(x)
plot(x,y1,x,y2)
```

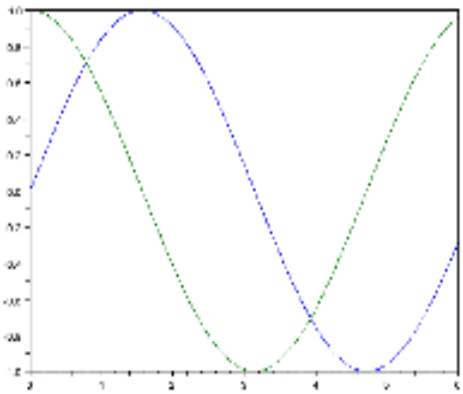
или

```
x=0:0.01:6
plot(x,sin(x),x,cos(x))
```

Можно так:

```
x=0:0.01:6
plot(x,sin(x))
plot(x,cos(x))
```

Графическое окно



Команда **plot(X,Y,S)** аналогична команде **plot(X,Y)**, в которой формат линии графика можно задавать с помощью строковой константы S. Символы, которые могут использоваться в параметре S, приведены в табл. 3.1.

Таблица 3.1

Тип линии		Тип точки		Цвет линии	
-	Сплошная	**.**	Точка	**y**	Желтый
:	Двойной пунктир	**o**	Окружность	**m**	Фиолетовый
-.	Штрих-пунктир	**x**	Крест	**c**	Голубой
--	Штриховая	**+**	Плюс	**r**	Красный
		*****	Звездочка	**g**	Зеленый
		s	Квадрат	**b**	Синий
		d	Ромб	**w**	Белый
		v	Треугольник	**k**	Черный

Например, для отображения графика функции $y=0.02x^3$ штриховой линией красного цвета с узловыми точками в виде ромбов можно выполнить следующую последовательность команд:

```
x=-5:0.5:2;
plot(x,0.02*x.^3,'--dr')
```

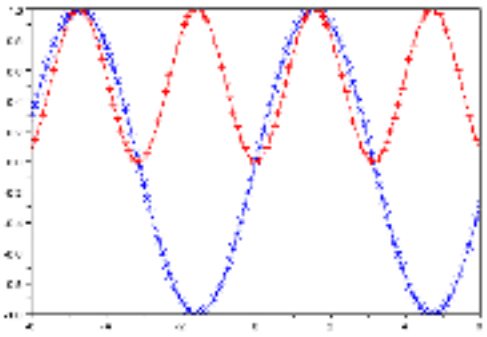
Команда **plot(X1,Y1,S1,X2,Y2,S2,…)** – строит графики нескольких функций на одном поле графика, где X1,Y1 – абсциссы и ординаты 1-й кривой, X2, Y2 – абсциссы и ординаты 2-й кривой и т. д. Дополнительные параметры S1, S2 и т. д. позволяют задать стиль линий графиков.

Рассмотрим пример построения графиков двух функций с различным стилем представления каждой из них:

Фрагмент программы

```
x=-6:0.1:6
y1=sin(x)
y2=sin(x).^2
plot(x,y1,'-xb',x,y2,'-+r')
```

Графическое окно



Команды **xgrid ()** позволяют задавать построение сетки на поле графика.

Заголовок графика и надписи осей графика можно вывести с помощью команды

xtitle(title,xstr,ystr),

где **title** – символьная константа, содержащая название графика; **xstr** – символьная константа, содержащая название оси X; **ystr** – символьная константа, содержащая название оси Y.

Идентификацию кривых графика (создание легенды) можно выполнить с использованием команды **legend**, например,

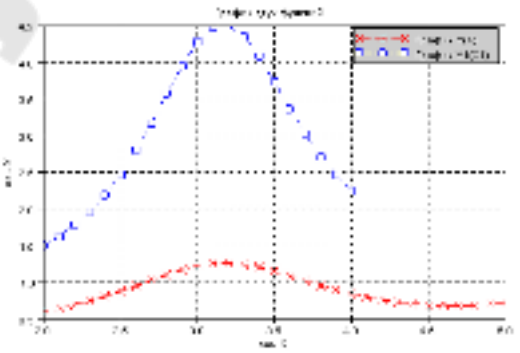
legend("График Y(X)","График Y1(X1)")

Ниже приведен пример построения графиков двух функций, у каждой из которых задан тип линии, маркер и цвет линии. На график нанесена координатная сетка, подписаны оси, выведена надпись самого графика и указана легенда в правом верхнем углу графика.

Фрагмент программы

```
figure(1)
X=2:0.1:5
Y=atan(X)./(1+sin(X).^2)
X1=2:0.1:4
Y1=(1+sqrt(0.5*X1))./(0.5+sin(X1).^2)
plot(X,Y,'--rx',X1,Y1,'-.во')
xgrid ()
xtitle("График двух функций",…
       "ось X","ось У")
legend("График Y(X)",…
       "График Y1(X1)")
```

Графическое окно



Для создания в графическом окне нескольких графических областей для вывода графиков применяется команда

subplot(m, n, p),

которая разбивает графическое окно на m×n областей, где m – число областей по вертикали, n – число областей по горизонтали, p – номер области, в которую будет выводиться текущий график (области отсчитываются последовательно **по строкам**).

Следующий пример иллюстрирует применение команды subplot:

```
x=-5:0.01:5;
subplot(2,2,1),plot(x,sin(x))
subplot(2,2,2),plot(sin(5*x),cos(2*x+0.2))
subplot(2,2,3), plot(x,sin(x)^2)
subplot(2,2,4), plot(x,sin(x)^3)
```

Результат приведен на рис. 3.1.

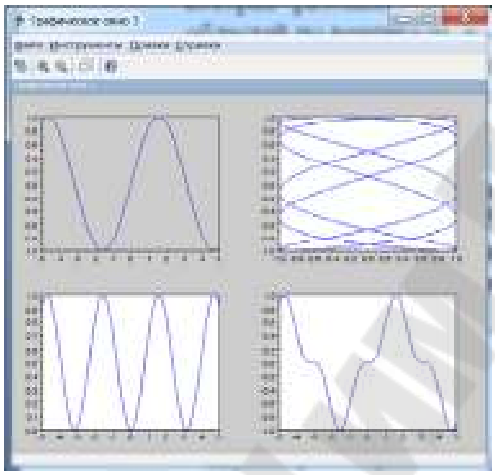


Рис. 3.1. Результат выполнения команды subplot

Практическая часть

Задание 1. Построение двумерных графиков

Постановка задачи. Построить график функции в заданных пределах изменения ее аргумента

Функция	x_n	x_k
$y = \dfrac{\operatorname{arctg}(x)}{1+\sin^2 x}$	2	5

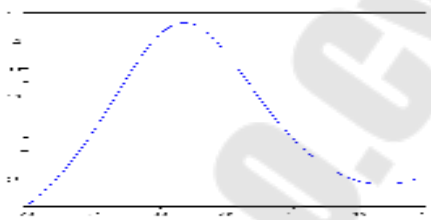
Шаг 1. Откроем графическое окно командой **figure(1)** и очистим его с помощью **newaxec** оператора.

Шаг 2. Создадим вектор **X**, значения которого будут изменяться от 2 до 5 с шагом 0.01 (вектор создается с использованием диапазона).

Шаг 3. Создадим вектор **Y**, каждое значение которого вычисляется с помощью заданной формулы (не забываем про поэлементные операции над векторами).

Шаг 4. Применим команду **plot** для построения графика функции, используя в качестве аргументов созданные вектора.

Пример программы по выполнению этого задания приведен ниже.

Программа	Результат выполнения
figure(1) newaxes **X=2:0.01:5** **Y=atan(X)./(1+sin(X).^2)** **plot(X,Y)**	

Индивидуальные задания приведены в табл. 3.2.

Таблица 3.2

Номер варианта	Функция	x_n	x_k	Номер варианта	Функция	x_n	x_k
1	$y=\frac{\operatorname{arctg}(x)}{1+\sin^2 x}$	2	5	2	$y=\frac{1+\sqrt{0{,}5x}}{0{,}5+\sin^2 x}$	2	4
3	$y=\ln\left(x^2+2x+2\right)$	-3	0	4	$y=e^x \sin x \cos^3 x$	-1	1
5	$y=\frac{\sin^2 x}{\sqrt{x}+x}$	2	5	6	$y=\frac{e^{0,1x}+1}{1+\cos^2 x}$	0	3
7	$y=2x^3-6x^2-18x+7$	-2	2	8	$y=2x^3-3x^2$	1	2
9	$y=x\sqrt{1+x^2}\cdot\sin x$	1	5	10	$y=-\ln\left(x^2-4x+5\right)$	1	4
11	$y=e^x\left(\sin 3x-3\cos 3x\right)$	1	4	12	$y=\left(x^2-2x\right)\ln x-\frac{3}{2}x^2+4x$	1	4
13	$y=\frac{\sin^3 x}{x+2}$	0	3	14	$y=\frac{3x^2+4x+4}{x^2+x+1}$	-1	1
15	$y=\frac{(1+2x)^2}{1.8+\cos^3 x}$	2	4	16	$y=\frac{1}{\ln\left(x^4+4x^3+30\right)}$	-1	3
17	$y=x\sin x+\cos x-\frac{1}{4}x^2$	-1,5	1,5	18	$y=3\cos^2 x-\cos^3 x$	1	5
19	$y=\frac{1-x+x}{1+x+x^2}$	-2	0	20	$y=\frac{0{,}5+\sin^2 x}{-0{,}5}$	0	4

Задание 2. Форматирование двумерных графиков

Постановка задачи. Построить на одном поле графики двух функций, промаркировать точки графиков, задать типы линий, подписать оси и весь график, создать легенду, нанести координатную сетку.

Функция № 1	x_n	x_k	Функция № 2	x_n	x_k
$y = \frac{\cos(x)}{1+\sin^2 x}$	2	5	$y = \frac{1+\sqrt{2{,}5x}}{0{,}2+\cos^2 x}$	2	4

Шаг 1. Откроем графическое окно командой **figure(1)**.

Шаг 2. Создадим вектор **X**, значения которого будут изменяться от 2 до 5 с шагом 0.1 (вектор создается с использованием диапазона).

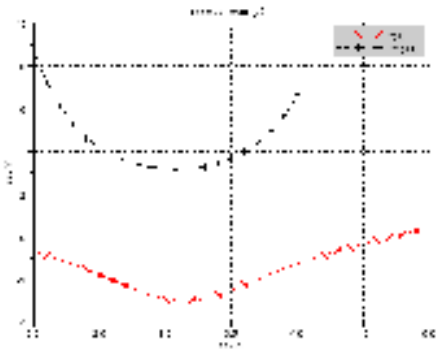
Шаг 3. Создадим вектор **Y**, каждое значение которого вычисляется по первой формуле (не забываем про поэлементные операции над векторами).

Шаг 4. Создадим вектор **X1**, значения которого будут изменяться от 2 до 4 с шагом 0.1 (вектор создается с использованием диапазона).

Шаг 5. Создадим вектор **Y1**, каждое значение которого вычисляется по второй формуле.

Шаг 6. Применим команду **plot** для построения графика функции, используя в качестве аргументов созданные вектора и указывая в форматной строке тип и цвет линии, а также символ для маркеров точек (например, **'--rx'** – красная штриховая линия с маркерами в виде креста).

Шаг 7. Добавим на график сетку, легенду, заголовки графика и осей.

Программа	Результат выполнения
figure(1) **X=2:0.1:5** **Y=3*cos(X)./(1+sin(X).^2)** **X1=2:0.1:4** **Y1=(1+sqrt(2.5*X1))./(0.2+cos(X1).^2)** **plot(X,Y,'--rx',X1,Y1,'-.k+')** **xgrid ()** **xtitle("График функций","ось X","ось У")** **legend("Y(X)"," Y1(X1)")**	

Индивидуальные задания приведены в табл. 3.3.

Таблица 3.3

№	Функция № 1	x_n	x_k	Функция № 2	x_n	x_k
1	$y=\frac{\sin x+3,7}{\lvert\cos x^3-2\rvert}$	2	5	$y=\frac{x^3}{x+3\cos^2 x^3}$	2	4
2	$y=\ln\left(x^2+2x+2\right)$	-3	0	$y=e^x\sin x\cos^3 x$	-1	1
3	$y=\frac{\sin^2 x}{\sqrt{x}+x}$	2	5	$y=\frac{e^{0,1x}+1}{1+\cos^2 x}$	0	3
4	$y=2x^3-6x^2-18x+7$	-2	2	$y=2x^3-3x^2$	1	2
5	$y=x\sqrt{1+x^2}\cdot\sin x$	1	5	$y=-\ln\left(x^2-4x+5\right)$	1	4
6	$y=e^x\left(\sin 3x-3\cos 3x\right)$	1	4	$y=\left(x^2-2x\right)\ln x-\frac{3}{2}x^2+4x$	1	4
7	$y=\frac{\sin^3 x}{x+2}$	0	3	$y=\frac{3x^2+4x+4}{x^2+x+1}$	-1	1
8	$y=\frac{(1+2x)^2}{1.8+\cos^3 x}$	2	4	$y=\frac{1000}{\ln\left(x^4+4x^3+30\right)}$	-1	3
9	$y=x\sin x+\cos x-\frac{1}{4}x^2$	-1,5	1,5	$y=3\cos^2 x-\cos^3 x$	1	5
10	$y=\frac{1-x+x}{1+x+x^2}$	-2	0	$y=\frac{0,5+\sin^2 x}{-0,5}$	0	4
11	$y=\cos^3 x+\frac{2x+1,09}{\sqrt{\lvert x-2\rvert}}$	3	6	$y=e^{\lvert x-4\rvert}\left(\sin^2 x+1\right)^x$	2	5
12	$y=3x^3-2x^2-3,5x+3$	-2	2	$y=0,5x^3-2x^2$	1	2
13	$y=2\left(\sin 2x-1,5\cos x\right)$	-2	0	$y=7\sin x+\cos^2 x-0,2x^2$	-1	3
14	$y=\frac{\cos^2 x}{x+1}$	2	4	$y=\frac{0,3+\sin^3 x}{x+1,5}$	-1	5
15	$y=\frac{2x^2+3x+1}{x^2+2x+1}$	0	2	$y=\frac{5\cos x+2,7}{0,5}$	-3	3

Задание 3. Построение графиков кусочно-непрерывных функций

Постановка задачи. Построить график кусочно-непрерывной функции, задав пределы изменения ее аргумента таким образом, чтобы в расчете значений функции участвовали все три формулы.

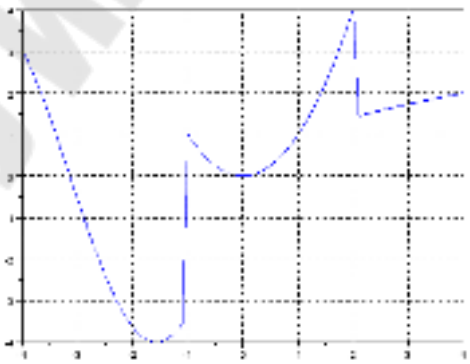
$$y = \begin{cases} \sqrt{x}, & \text{если} \quad x > 2 \\ x^2, & \text{если} \quad -1 \le x \le 2 \\ 4\sin x & \text{в остальных случаях} \end{cases}$$

Шаг 1. Откроем графическое окно командой **figure(1)**.

Шаг 2. Создадим вектор **X**, значения которого будут изменяться от -4 до 4 с шагом 0.1 (чтобы в расчете участвовали все три формулы).

Шаг 3. Для того чтобы сформировать вектор значений функции, организуем цикл с переменной цикла ***i.*** Чтобы определить, по какой формуле для каждого элемента вектора вычислять значение функции, будем использовать оператор **if**.

Шаг 4. Применим команду **plot** для построения графика функции, используя в качестве аргументов созданные вектора.

Программа	Результат выполнения
figure(1) **X=-4:0.1:4, i=1** **while i<=length(X)** **if X(i)>2 then y=sqrt(X(i)),** **elseif (X(i)>=-1) & (X(i)<=2) then** **y=X(i)^2** **else y=4*sin(X(i))** **end** **Y(i)=y** **i=i+1** **end** **plot(X,Y), xgrid ()**	

Индивидуальные задания приведены в табл. 2.7.

3.2. Вычисление интегралов, решение уравнений и систем

Краткие теоретические сведения

В Scilab вычисление определенного интеграла методом трапеций реализовано функцией

inttrap(x,y),

где **x** – вектор значений аргумента подынтегральной функции на отрезке интегрирования; **y** – вектор значений, полученных при вычислении подынтегральной функции для элементов вектора **x**.

Например, для вычисления $\int_2^{5.3} \frac{2xdx}{\sin x + 1.5}$ нужно выполнить следующий набор команд:

Фрагмент программы	Результат
x =2:0.01:5.3 **y =2*x./(sin(x)+1.5)** **integral = inttrap(x,y)** **disp(integral)**	**30.436962**

Для вычисления определенного интеграла с использованием алгоритма квадратурных формул предназначена функция

integrate(fun, x, a, b, ,er1,er2),

где **fun** – подынтегральная функция в символьном виде; **x** – переменная интегрирования в символьном виде; **a, b** – пределы интегрирования; **er1,er2** – абсолютная и относительная погрешности интегрирования (необязательные параметры).

Например, для вычисления $\int_2^{5.3} \frac{2xdx}{\sin x + 1.5}$ нужно задать:

Фрагмент программы	Результат
z=integrate('2*x./(sin(x)+1.5)', 'x', 2, 5.3) **disp(z)**	**30.437056**

Универсальная команда интегрирования:

[integral,err]=intg(a, b, name ,er1,er2),

где **a**, **b** – пределы интегрирования; **name** – имя подынтегральной функции (может быть задана с помощью внешней функции); **er1,er2** – абсолютная и относительная погрешности интегрирования (необязательные параметры). Функция **intg** возвращает значение интеграла (integral) и погрешность вычислений (err).

Внешнюю функцию можно задать командой

deff('*переменная=имя функции*(*параметр*)', '*символьное представление функции*')

Например, deff('y=F(x)', 'y=2*x./(sin(x)+1.5)')

или

function ***переменная = имя функции*(*параметр-аргумент функции*)**

операторы, вычисляющие значение функции

endfunction

Например,

```
function    y=f(t)
      y=t^2/sqrt(3+sin(t))
endfunction
```

или

```
function y=f(t),y=t^2/sqrt(3+sin(t)),endfunction
```

Ниже приводится пример вычисления интеграла $\int_{2}^{5.3} \frac{2xdx}{\sin x + 1.5}$.

Фрагмент программы	Результат
function y=f(x) **y=2*x/(sin(x)+1.5)** **endfunction** **z=intg(2,5.3,f), disp(z)**	**30.437056**

Для решения нелинейных уравнений в Scilab используется функция

fsolve(x0,f),

где **x0** – начальное приближение корня; **f** – функция, описывающая левую часть уравнения **f(x)=0**.

Например, для решения уравнения $\sin(2x) = \cos(3x^2) + \sin(3x)$ для начального приближения $x \approx 7$ нужно выполнить следующие команды:

Фрагмент программы	Результат
deff('y=F(x)', 'y=sin(2*x)-cos(3*x^2)-sin(3*x)') **root=fsolve(7,F)** **disp(root)**	**6.9755674**

Для решения полиномиальных уравнений вида $v_n \cdot x^n + v_{n-1} \cdot x^{n-1} + ... + v_1 \cdot x + v_0 = 0$ используется функция

roots(a),

где **a** – вектор коэффициентов перед неизвестными полинома размерностью n+1(n – порядок полинома).

Результатом работы этой функции будет вектор корней полинома размерностью n.

Пример решения полиномиального уравнения $3x^3 + x^2 - 10x - 8 = 0$ приведен ниже.

Фрагмент программы	Результат
v=[3 1 -10 -8]	**2.**
R=roots(v)	**-1.3333333**
disp(R)	**-1.**

Для уравнения $3x^3 + x^2 - 10x + 8 = 0$ два корня – комплексные.

Фрагмент программы	Результат
v=[3 1 -10 8]	**- 2.2935835**
R=roots(v)	**0.9801251+0.4494650i**
disp(R)	**0.9801251-0.4494650i**

Функция **roots** может также принимать в качестве параметра полином, созданный функцией **poly** и представляющий собой левую часть уравнения $v_n \cdot x^n + v_{n-1} \cdot x^{n-1} + ... + v_1 \cdot x + v_0 = 0$:

poly(a, x, f1),

где **a** – вектор коэффициентов полинома, записанных в обратном порядке; **x** – символьная переменная; **f1** – символьная переменная, принимающая значения **'r'** или **'c'** (roots или coeff).

Например, чтобы создать полином $4x^4 + x^2 - 2x + 3$, нужно использовать команду:

Фрагмент программы	Результат
p=poly([3 -2 1 0 4],'x','c')	2 4
disp(p)	3 – 2x + x + 4x

Для решения уравнения $3x^3 + x^2 - 10x - 8 = 0$ можно выполнить следующие команды:

Фрагмент программы	Результат
p=poly([-8 -10 1 3],'x','c')	**2.**
R=roots(p)	**-1.3333333**
disp(R)	**-1.**

Для решения систем линейных уравнений в Scilab есть следующие способы:

– применение операции левого матричного деления;

– использование обратной матрицы.

Если задана система линейных алгебраических уравнений вида:

AX=B,

где **A** – матрица коэффициентов перед неизвестными системы; **B** – вектор свободных членов, то решение системы может быть найдено в виде:

X=A \ B

То же самое решение может быть получено с помощью обратной матрицы, например:

X=inv(A)*B

Например, решить систему уравнений $\begin{cases} 3x_1 + x_2 = -4 \\ -3x_1 + 5x_2 = 36 \end{cases}$ можно так:

Фрагмент программы	Результат
A=[3 1;-3 5]; B=[-4 ;36];	
X=inv(A)*B, disp(X)	**-3.1111111**
или	**5.3333333**
X1=A\B , disp(X1)	

Для решения систем нелинейных уравнений можно использовать функцию

fsolve(x0,f),

где **x0** – вектор начальных приближений для неизвестных; **f** – функция, определяющая систему уравнений вида f(x)=0.

Например, решение системы $\begin{cases} 2 \cdot x + y = 6 \\ x^2 + y^2 = 14 \end{cases}$ можно выполнить следующим образом:

Фрагмент программы	Результат
function [y]=fun(x)	
y(1)=2*x(1)+x(2)-6	**1.2338096**
y(2)=x(1)^2+x(2)^2-14	**3.5323808**
endfunction	
X0=[1;1]	
R=fsolve(X0,fun)	
disp(R)	

Практическая часть

Задание 1. Вычисление определенного интеграла

Постановка задачи. Вычислить числовое значение интеграла от этой функции в заданных пределах интегрирования методом трапеций, методом квадратурных формул и с помощью функции **intg.**

$$\int_{2.1}^{4.3} \frac{\sin x dx}{1.5}$$

Шаг 1. Создадим вектор **X**, значения которого будут изменяться от 2,1 до 4,3 с шагом 0.01.

Шаг 2. Создадим вектор **Y**, каждое значение которого вычисляется по формуле $\frac{\sin x}{1.5}$.

Шаг 3. Применим команду **inttrap(X, Y)**.

Шаг 4. Используем функцию **integrate**, задав подынтегральную функцию в символьном виде.

Шаг 5. Определим внешнюю функцию с помощью команды **deff** или конструкции **function**.

Шаг 6. Выведем результаты, используя команду **disp**.

Программа	Результат выполнения
X =2.1:0.01:4.3 **Y =sin(X)/1.5** **integral_1 = inttrap(X,Y)** **integral_2=integrate('sin(x)/1.5', 'x', 2.1, 4.3)** **disp(integral_1)** **disp(integral_2)**	**-0.0693640** **-0.0693646**
deff('y=F(x)','y=sin(x)/1.5'); **integral_3=intg(2.1,4.3,F)** **function y=f(x)** **y=sin(x)/1.5** **endfunction** **integral_4=intg(2.1,4.3,f)** **disp(integral_3)** **disp(integral_4)**	**-0.0693646** **-0.0693646**

Индивидуальные задания приведены в табл. 3.4.

Таблица 3.4

1	$\int_{5.1}^{8.3} \frac{xdx}{x+2.5}$	2	$\int_{0.5}^{\pi/2} \frac{\cos^2 x}{\sin x} dx$	3	$\int_{3}^{6.1} \frac{x+6.25}{(x+1.5)^2} dx$
4	$\int_{0.5}^{2} (5-x^2)^{0.5} dx$	5	$\int_{0.1}^{1.5} \frac{dx}{\sin x \cos x}$	6	$\int_{3.1}^{4.6} \frac{xdx}{\sqrt{x^4-x}}$
7	$\int_{0.5}^{6.1} \frac{dx}{x^2(x+1.3)}$	8	$\int_{0.1}^{1.6} \frac{2-\sin x}{1+\cos x} dx$	9	$\int_{-1.3}^{3} \frac{x^3 dx}{(9+x^2)^{3/2}}$
10	$\int_{1.1}^{3.6} \frac{dx}{(x+1)\sqrt{x+x^4}}$	11	$\int_{0.1}^{4.2} \frac{dx}{x^2+3x}$	12	$\int_{2}^{4.5} \frac{dx}{\sqrt{(x+2)(x+0.5)}}$
13	$\int_{2.5}^{7.5} \frac{dx}{x(x+1.6)^2}$	14	$\int_{2.6}^{7.1} \frac{(x^2+3)^{3/2}}{x^3} dx$	15	$\int_{0.56}^{1.5} \frac{dx}{\cos^3 x \sin^3 x}$

Задание 2. Поиск корней уравнения, графическая интерпретация

Постановка задачи. Найти корень уравнения для заданного начального приближения. Выполнить графическую интерпретацию результата.

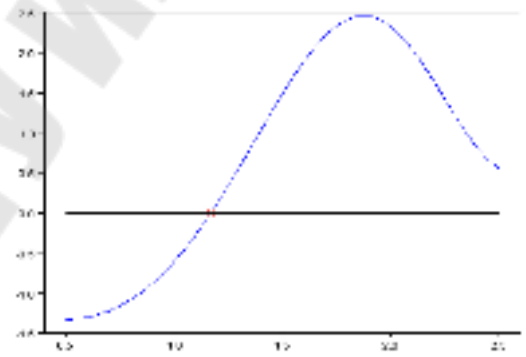
$\sin(x) = \cos(x^2) + \sin(2x)$	$x \approx 1$

Шаг 1. Определим внешнюю функцию с помощью команды **deff** или конструкции **function**.

Шаг 2. Найдем корень уравнения с помощью функции **fsolve**, подставив в качестве первого параметра заданное начальное приближение. Результат будет храниться в переменной **k**.

Шаг 3. Выведем результат, используя команду **disp**.

Шаг 4. Выполним графическую интерпретацию результата. Для этого зададим аргумент функции из левой части уравнения таким образом, чтобы найденный корень попадал в диапазон между первым и последним элементом вектора. Построим график функции из левой части уравнения с помощью **plot**. Построим также линию **y=0** и отметим точку с абсциссой, равной корню, и ординатой, равной значению функции для корня.

Программа	Результат выполнения
deff('y=F(x)', … **'y=sin(x)-cos(x^2)-sin(2*x)')** **k=fsolve(1,F)** **disp(k)** **x=0.5:0.01:2.5** **plot(x,F(x),'-b',k,F(k),'xr',x,0,'-k')**	**1.1695683** 

Индивидуальные задания приведены в табл. 3.5.

Таблица 3.5

Вариант	Уравнение	Начальное приближение
1	$\cos x - 0.1x = 0$	$x \approx 1$
2	$0.5\sin x - x^2 = 0$	$x \approx 2$
3	$2\sin x - \cos^2 x = 0$	$x \approx 0.5$
4	$x^2 - 4x - 6 = \sqrt{2x^2 - 8x + 12}$	$x \approx 0$
5	$\sin(x)\cos(x-2) - \sin(x) = \sin(x+2)\cos(x)$	$x \approx 7$
6	$\sin(\frac{x}{2})\cos(\frac{3x}{2})) = 0$	$x \approx 5$

Окончание табл. 3.5

Вариант	Уравнение	Начальное приближение
7	$\sin 3x^2 = 1 + 2\cos 0.5x$	$x \approx 1$
8	$\sqrt{3x^2+1} + \sqrt{x^2+3} = \sqrt{6x^2+10}$	$x \approx 2$
9	$5(1+\cos(x)) = 2 + \sin^4 x - \cos^4 x$	$x \approx 2$
10	$2.5\sin x - 0.8x^2 = 0$	$x \approx 2$
11	$\sqrt{x^2+x-1} + \sqrt{x^2+x+3} = \sqrt{2x^2+8}$	$x \approx 5$
12	$1 + \sin(2x) = (\cos(3x) + \sin(3x))^2$	$x \approx 3$
13	$2\sin 3 - 1 = \cos^2 x - 0.7$	$x \approx 3$
14	$\sqrt{x+2\sqrt{x-1}} + \sqrt{x-2\sqrt{x-1}} = x - 1$	$x \approx 3$
15	$\arcsin(x) - 0.2x - 0.1 = 0$	$x \approx 0$

Задание 3. Поиск корней полиномиального уравнения, графическая интерпретация

Постановка задачи. Найти все корни полиномиального уравнения. Выполнить графическую интерпретацию для одного из найденных действительных корней.

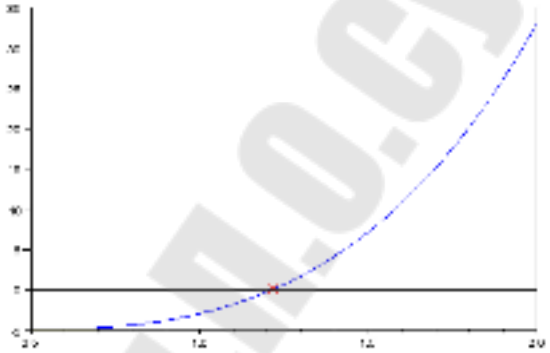
$$2x^4 + 4x^2 - 6x - 3 = 0$$

Шаг 1. Создадим вектор коэффициентов полинома в левой части уравнения (или полином с помощью **poly**).

Шаг 2. Найдем корни уравнения с помощью функции **roots**.

Шаг 3. Выведем результат, используя команду **disp**.

Шаг 4. Выполним графическую интерпретацию результата. Для этого зададим аргумент функции из левой части уравнения таким образом, чтобы выбранный действительный корень попадал в диапазон между первым и последним элементом вектора. Построим график функции из левой части уравнения с помощью **plot**. Построим также линию **y=0** и отметим точку с абсциссой, равной корню, и ординатой, равной значению функции для корня.

Программа	Результат выполнения
v=[2 0 4 -6 -3] **R=roots(v)** **disp(R)** **k=R(3)** **x=0.5:0.01:2** **y=2*x.^4+4*x.^2-6*x-3** **Fk=2*k^4+4*k^2-6*k-3** **plot(x,y,'-b',k, Fk, 'xr',x,0,'-k')**	**- 0.4129576+1.7282075i** **- 0.4129576-1.7282075i** **1.2164706** **- 0.3905555** 

Индивидуальные задания приведены в табл. 3.6.

Таблица 3.6

Вариант	Уравнение
1	$x^4 - 5.67x^3 - 1.906x^2 + 15.81x + 3.282 = 0$
2	$x^4 - 5.8x^3 - 4.2x^2 - 18.6x - 3.6 = 0$
3	$x^4 - 2.5x^3 - 2.54x^2 + 6.3x + 1.38 = 0$
4	$x^4 + 9.65x^3 - 4.97x^2 - 30.15x - 5.91 = 0$
5	$x^4 - 1.2x^3 - 2.8x^2 + 2.4x + 0.6 = 0$
6	$x^4 - 0.4x^3 - 7.07x^2 + 8.73x - 1.98 = 0$
7	$x^4 + 4.65x^3 - 7.575x^2 - 27.125x + 31.36 = 0$
8	$x^4 - 10.2x^3 - 6.09x^2 + 78.31x - 66.66 = 0$
9	$x^4 + 11.33x^3 - 8.243x^2 - 74.553x + 75.438 = 0$
10	$x^4 - 1.1x^3 - 7x^2 + 13.7x - 6.6 = 0$
11	$x^4 - 12.8x^3 + 2.15x^2 - 2.8x + 1.15 = 0$
12	$x^4 - 17.6x^3 + 36.213x^2 - 17.61x + 35.213 = 0$
13	$x^4 + 17.87x^3 - 45.391x^2 + 17.87x - 46.391 = 0$
14	$x^4 - 1.3x^3 - 1.3x^2 - 1.3x - 2.3 = 0$
15	$x^3 + 4x^2 - 6x + 2 = 0$

Задание 4. Решение системы линейных уравнений

Постановка задачи. Решить систему линейных уравнений.

$$\begin{cases} 3x_1 + x_2 + x_3 = -4 \\ -3x_1 + 5x_2 + 6x_3 = 36 \\ x_1 - 4x_2 - 2x_3 = -19 \end{cases}$$

Шаг 1. Создадим матрицу коэффициентов при неизвестных

Шаг 2. Создадим вектор свободных членов.

Шаг 3. Умножим матрицу, обратную к матрице коэффициентов, на вектор свободных членов (или применим операцию левого матричного деления).

Шаг 4. Выведем результат, используя команду **disp**.

Программа	Результат выполнения
A=[3 1 1;-3 5 6;1 -4 -2] **B=[-4;36;-19]** **X=inv(A)*B** **disp(X)**	**-3.** **3.** **2.**

Индивидуальные задания приведены в табл. 3.7.

Таблица 3.7

Номер варианта	Система уравнений	Номер варианта	Система уравнений
1	$\begin{cases} 2x_1 + x_2 + 3x_3 = 7 \\ 2x_1 + 3x_2 + x_3 = 1 \\ 3x_1 + 2x_2 + x_3 = 6 \end{cases}$	2	$\begin{cases} 2x_1 - x_2 + 2x_3 = 3 \\ x_1 + x_2 + 2x_3 = -4 \\ 4x_1 + x_2 + 4x_3 = -3 \end{cases}$
3	$\begin{cases} 3x_1 - x_2 + x_3 = 12 \\ x_1 + 2x_2 + 4x_3 = 6 \\ 5x_1 + x_2 + 2x_3 = 3 \end{cases}$	4	$\begin{cases} 2x_1 - x_2 + 3x_3 = -4 \\ x_1 + 3x_2 - x_3 = 11 \\ x_1 - 2x_2 + 2x_3 = -7 \end{cases}$
5	$\begin{cases} 3x_1 - 2x_2 + 4x_3 = 12 \\ 3x_1 + 4x_2 - 2x_3 = 6 \\ 2x_1 - x_2 - x_3 = -9 \end{cases}$	6	$\begin{cases} 8x_1 + 3x_2 - 6x_3 = -4 \\ x_1 + x_2 - x_3 = 2 \\ 4x_1 + x_2 - 3x_3 = -5 \end{cases}$
7	$\begin{cases} 4x_1 + x_2 - 3x_3 = 9 \\ x_1 + x_2 - x_3 = -2 \\ 8x_1 + 3x_2 - 6x_3 = 12 \end{cases}$	8	$\begin{cases} 3x_1 - 2x_2 + 4x_3 = 21 \\ 3x_1 + 4x_2 - 2x_3 = 9 \\ 2x_1 - x_2 - x_3 = 10 \end{cases}$

Окончание табл. 3.7

Номер варианта	**Система уравнений**	**Номер варианта**	**Система уравнений**
9	$\begin{cases} 3x_1 - 2x_2 - 5x_3 = 5 \\ 2x_1 + 3x_2 - 4x_3 = 12 \\ x_1 - 2x_2 + 3x_3 = -1 \end{cases}$	10	$\begin{cases} 4x_1 + x_2 + 4x_3 = 19 \\ 2x_1 - x_2 + 2x_3 = 11 \\ x_1 + x_2 + 2x_3 = 8 \end{cases}$
11	$\begin{cases} 2x_1 - x_2 + 2x_3 = 0 \\ 4x_1 + x_2 + 4x_3 = 6 \\ x_1 + x_2 + 2x_3 = 4 \end{cases}$	12	$\begin{cases} 2x_1 - x_2 - 3x_3 = 0 \\ 3x_1 + 4x_2 + 2x_3 = 1 \\ x_1 + 5x_2 + x_3 = -3 \end{cases}$
13	$\begin{cases} 2x_1 - x_2 + 2x_3 = 8 \\ x_1 + x_2 + 2x_3 = 11 \\ 4x_1 + x_2 + 4x_3 = 22 \end{cases}$	14	$\begin{cases} 2x_1 - x_2 - 3x_3 = -9 \\ x_1 + 5x_2 + x_3 = 20 \\ 3x_1 + 4x_2 + 2x_3 = 15 \end{cases}$
15.	$\begin{cases} -3x_1 + 5x_2 + 6x_3 = -8 \\ 3x_1 + x_2 + x_3 = -4 \\ x_1 - 4x_2 - 2x_3 = -9 \end{cases}$	16	$\begin{cases} 3x_1 + x_2 + x_3 = -4 \\ -3x_1 + 5x_2 + 6x_3 = 36 \\ x_1 - 4x_2 - 2x_3 = -19 \end{cases}$

Учебное электронное издание комбинированного распространения

Учебное издание

Трохова Татьяна Анатольевна
Романькова Татьяна Леонидовна

ВВЕДЕНИЕ В SCILAB

Практикум
по курсу «Информатика»
для студентов технических специальностей
дневной и заочной форм обучения

Электронный аналог печатного издания

Редактор *Н. Г. Мансурова*
Компьютерная верстка *Н. Б. Козловская*

Подписано в печать 14.11.16.
Формат 60x84/16. Бумага офсетная. Гарнитура «Таймс».
Цифровая печать. Усл. печ. л. 3,25. Уч.-изд. л. 3,45.
Изд. № 68.
http://www.gstu.by

Издатель и полиграфическое исполнение
Гомельский государственный
технический университет имени П. О. Сухого.
Свидетельство о гос. регистрации в качестве издателя
печатных изданий за № 1/273 от 04.04.2014 г.
246746, г. Гомель, пр. Октября, 48