Поволжский Государственный Университет
Телекоммуникаций и Информатики

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
к лабораторным работам

"Программирование в системе Scilab"
Часть 1. Использование Scilab и Scicos

Автор-составитель:	Акчурин Э.А.	д.т.н., профессор
Редактор:	Акчурин Э.А.	д.т.н., профессор
Рецензент:	Тарасов В.Н.	д.т.н., профессор

Самара

2009

Факультет информационных систем и технологий

Кафедра «Информатика и вычислительная техника»

Автор - д.т.н., профессор Акчурин Э.А.

Другие материалы по дисциплине Вы найдете на сайте

www.ivt.psati.ru

Содержание

Введение

Лабораторный цикл содержит 11 работ по изучению программирования с использованием математической системы Scilab и входящей в него программы моделирования Scicos.

Цикл может быть использован в учебных курсах для аспирантов, а также в дисциплине "Компьютерные технологии в науке и образовании" для магистров направлений 210400.

Рекомендуемая литература:

1. Алексеев Е. Scilab. Решение инженерных и математических задач. М.: ALT Linux; БИНОМ. Лаборатория знаний. 2008, 260с.
2. Андриевский Б., Фрадков А. "Элементы математического моделирования в программных средах MATLAB 5 и Scilab" СПб.: Наука, 2001. 286с.
3. Chandler G. Stephen R. Introduction to Scilab. 2002, 27с.
4. Domanie de Voluceau. Introduction to Scilab. Scilab Groupe. 125с. http://www./Intro.pdf.
5. Domanie de Voluceau. Scilab Reference Mahual. Scilab Groupe. 700с. http://www./manual.pdf.
6. Gomez C. Communication Toolbox. 12с. http://www./comm.pdf.
7. Domanie de Voluceau. Guide for Developers. Scilab Groupe. 29с. http://www./Internals.pdf.
8. Domanie de Voluceau. InterSci. A scilab interfacing tool. Scilab Groupe. 14с. http://www./Intersci.pdf.
9. Nikoukbah K. LMI Tool: a Package for LMI Optimisation in Scilab. 16с. http://www./lmi.pdf.
10. Gomez C., Goursat M. Metanet User's Guide and Tutorial. 19с. http://www./metanet.pdf.
11. Nikoukbah K. A Dynamic System Builder and Simulator. User's Guife. 15с. http://www./scicos.pdf.
12. Domanie de Voluceau. Signal Processing with Scilab. Scilab Groupe. 205с. http://www./signal.pdf.

Содержание отчета по каждой работе:

- Название работы, задание в соответствии с вариантом.
- Программа.
- Результаты выполнения программы на ПК.
- Выводы.

1. Основы Scilab

Предметная область

Знакомство с СКМ Scilab Освоение интерфейса, просмотр демосов, работа в режиме калькулятора.

Контрольные вопросы

1. Структура окна системы Scilab.
2. Команды пункта "File" системного меню.
3. Команды пункта "Edit" системного меню.
4. Команды пункта "Preference" системного меню.
5. Команды пункта "Control" системного меню.
6. Команды пункта "Editor" системного меню.
7. Команды пункта "Applications" системного меню.
8. Команды пункта "?" системного меню.
9. Правила ввода команд.
10. Правила ввода функций и операндов.
11. Правила ввода выражений.
12. Правила ввода комментариев.
13. Правила просмотра результатов операций.

Задания к работе

Задание 1. Изучить интерфейс Scilab.

Задание 2. Ознакомиться с демонстрационными примерами Scilab.

Задание 3. Выполнить в режиме калькулятора следующие действия:

- Ввод исходных операндов.
- Выполнить над операндами 1 и 2 операцию 1.
- Выполнить над результатом и операндом 1 операцию 2.
- Выполнить над результатом и операндом 2 операцию 3.
- Возвести операнд 1 почленно в степень 3.

Варианты заданий

№	Операнд 1	Операнд 2	Операторы		
			1	2	3
1.	V=[12 34 61 45]	v = 34	*	./	+
2.	V=[80 67 34 11]	v = 43	/	.*	-
3.	V=[19 77 45 11]	v = -5	+	.\	/
4.	V=[11 98 67 45]	v = 7	-	.*	/
5.	V=[67 34 67 45]	v = -12	+	.\	*
6.	V=[18 36 45 45]	v = 10	/	./	-
7.	V=[55 43 18 45]	v = 44	/	.*	/
8.	V=[32 28 55 45]	v = 87	*	-	/

Методические указания

В Scilab все данные рассматриваются, как матрицы. Тип результата определяется автоматически по виду выражения.

В идентификаторах высота буквы имеет значение. Рекомендуется для имен простых переменных выбирать строчные буквы, а для структурированных (векторы и массивы) - прописные.

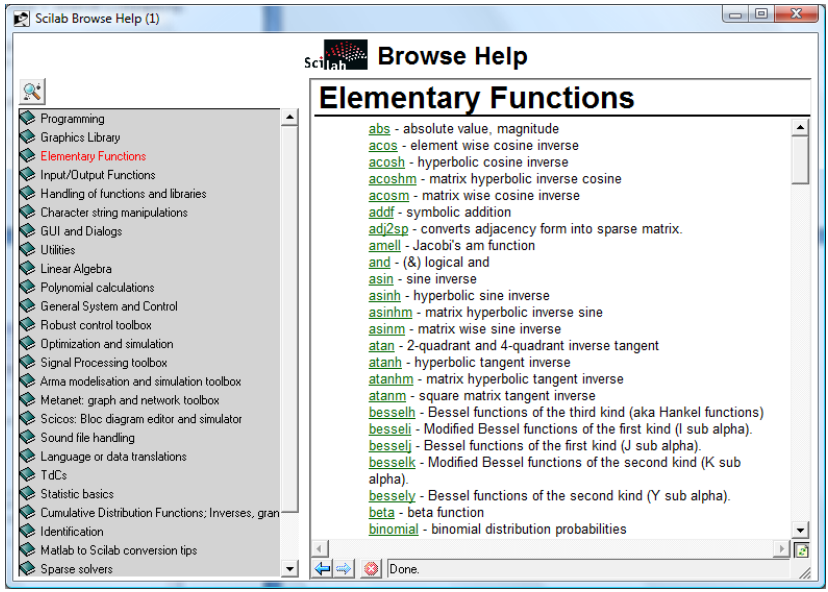
Векторы вводятся в квадратных скобках, компоненты вектора разделяются запятыми (или пробелами). Например, V=[1,2,3].

Матрицы вводятся в квадратных скобках, внутри размещаются векторы строк, разделенные знаком точка с запятой (;). Например, V=[1,2,3;4,5,6;7,8,9].

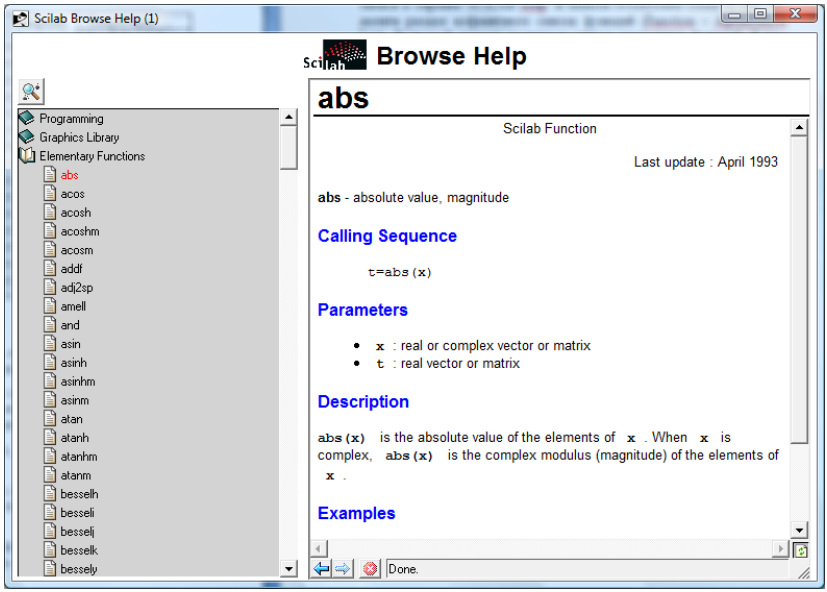
Если данные не умещаются в строке, строку можно отобразить в нескольких строках, используя разделитель в виде многоточия (не менее трех точек).

Значение π задается системной константой с именем %pi.

Для правильного вызова встроенных функций рекомендуется обратиться к справке Scilab Help. В панели оглавления слева нужно выделить раздел, в примере список элементарных функций (Elementary Functions). В правой панели отображается список доступных функций.



При открытии раздела отображается список доступных функций. В нем нужно найти и выделить нужную функцию. В правой панели отображается описание выбранной функции с форматом обращения и примерами использования. Ниже в качестве примера показано обращение к справке по функции взятия модуля abs(X). Тот же результат достигается при щелчке по функции в правом поле.



В Scilab возможны два режима работы:

- В командном окне, как с калькулятором. В этом случае каждое действие сразу же исполняется.
- В редакторе программ. В этом случае программа вводится, как обычно, а исполняется по команде встроенного отладчика.

При работе в режиме калькулятора выражения могут вводиться:

- В прямой форме, тогда после завершения ввода ответ будет выведен под встроенным системным именем ans. Переменная с этим именем всегда хранит результат последнего вычисления.
- В форме оператора присвоения, когда переменной с выбранным именем присваивается значение выражения. Ответ в этом случае выводится под именем этой переменной.

Если вычисляется значение переменной с выбранным именем по заданному выражению, результат выводится под именем этой переменной в следующей строке. Векторы выводятся в строке с пробелами, матрицы - построчно, каждая содержит вектор строки.

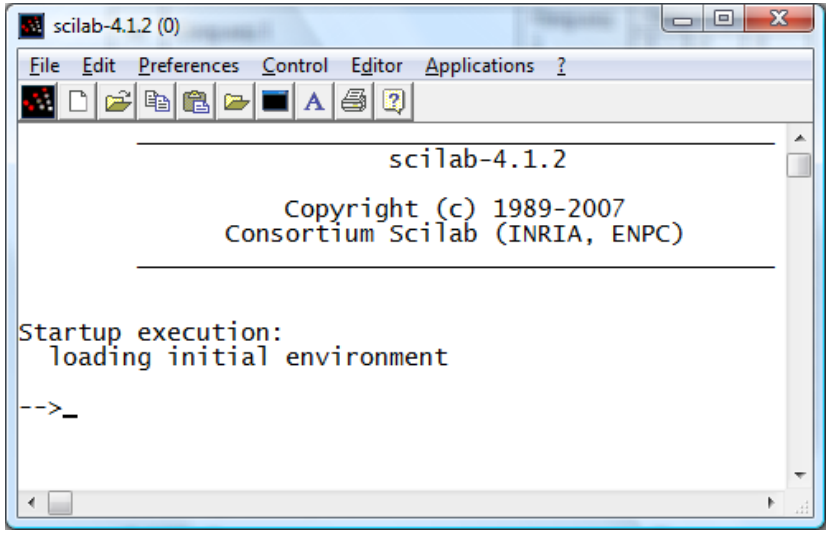
При работе с программой неграфические результаты выводятся в окно командной строки.

Вывод результата можно заблокировать, если в конце строки ввода ввести знак точка с запятой (;). Значение переменной, которой результат присваивается, хранится в рабочей области.

При работе с массивами определены операторы почленного выполнения. В них перед символом операции вводится точка (.).

Символ присвоения - знак равенства (=). Равенство, как оператор отношения в условиях, вводится, как двойное равенство (==).

Пример 1

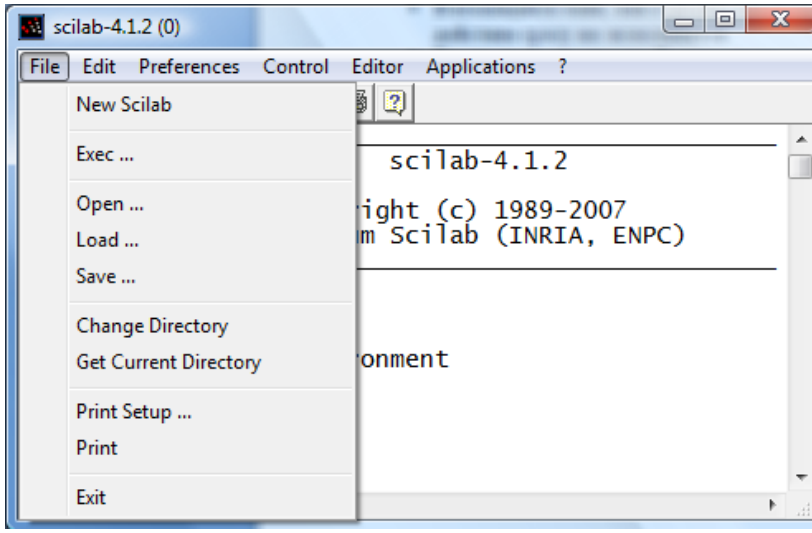


Пункты панели инструментов в порядке слева направо:

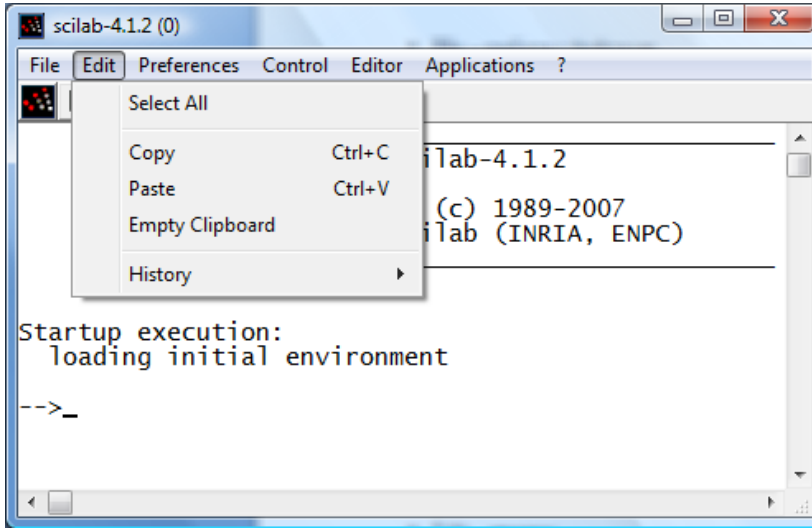
- New Scilab - новая Scilab.
- Open Scipad - открыть редактор Scipad.
- Open File - открыть файл.
- Copy - копировать.
- Paste - вставить.
- Change directory - изменить каталог.
- Scilab Output – Консоль Scilab в отдельном окне.
- Print –печать.
- Scilab Help – справка.

Пункты главного меню:

File – работа с файлами. Определены средства работы с файлами.

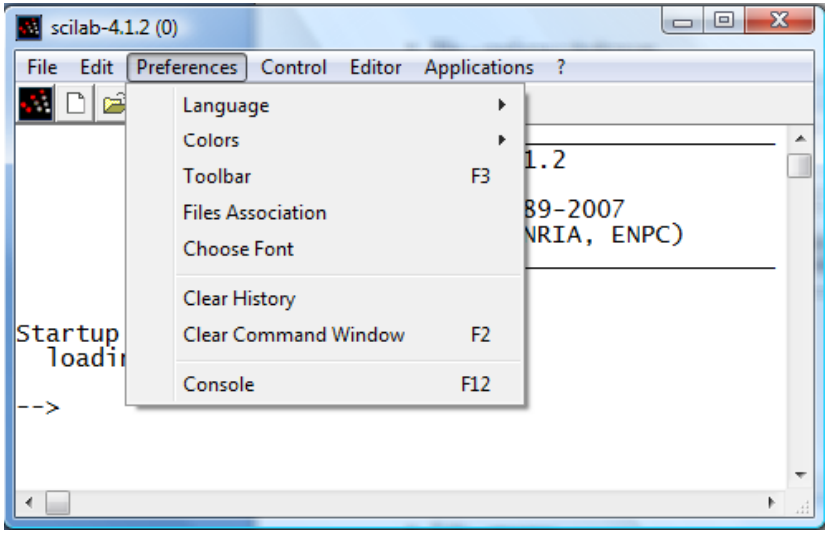


Edit – правка. Определены стандартные средства редактирования.

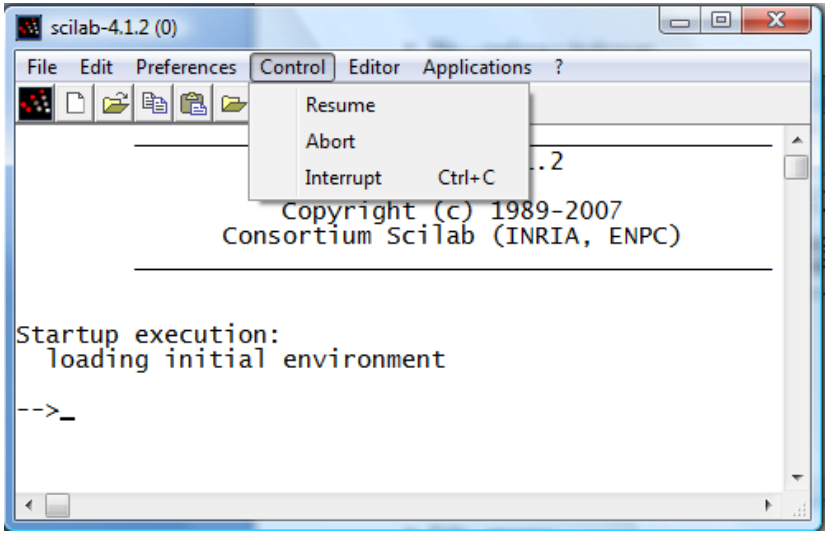


Preferences – предпочтения. Содержит команды: Language (выбрать язык, английский или французский), Colors (цвета текста и фона), Toolbar (отображать ли панели инструментов), Files Association (ассоциированные файлы по расширениям), Choose font (выбрать шрифт), Clear History (очистить историю

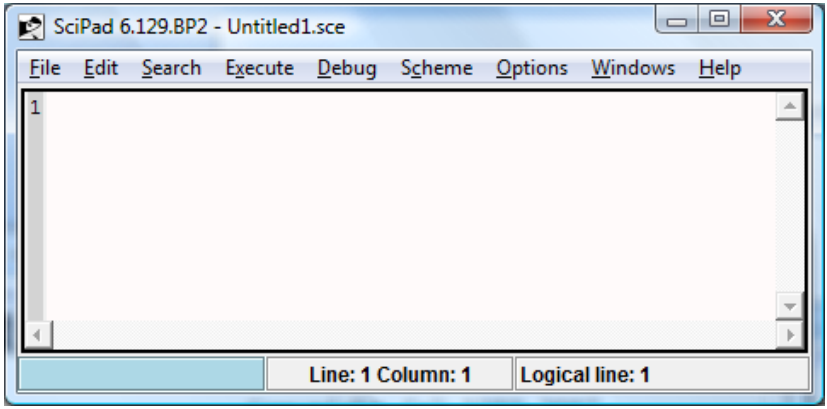
команд), Clear Command Window (очистить командное окно), Console (консоль).



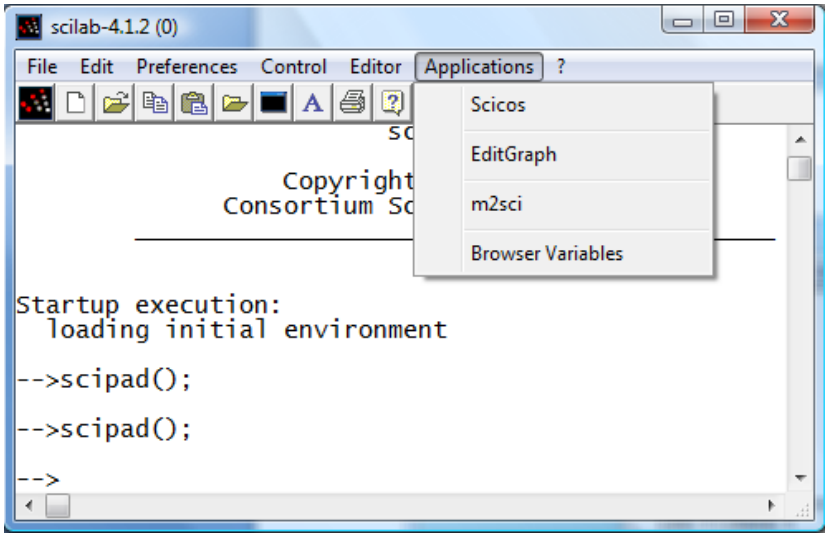
Control – управление. Содержит команды: Resume (повторить), Abort (отменить), Interrupt (прервать).



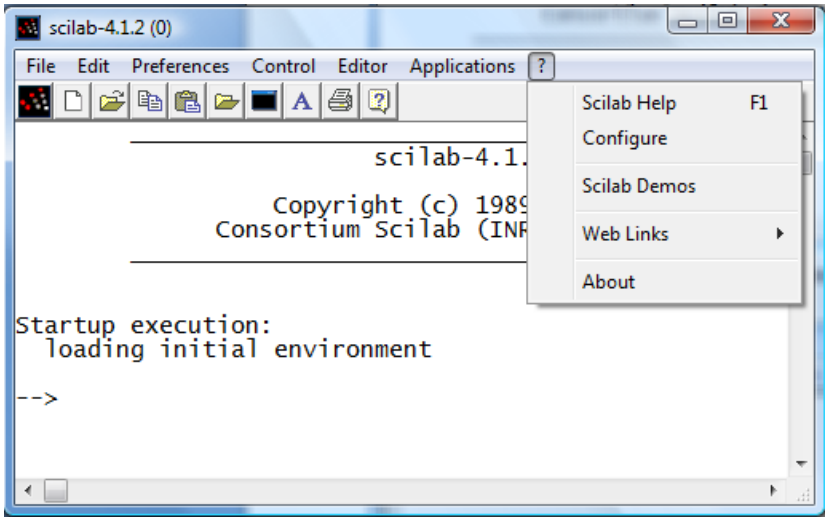
Editor – редактор. Открывается окно создания исполняемого файла с расширением .sce.



Applications – приложения. Выбираемые средства: Scicos (программа имитационного моделирования), EditGraph (графический редактор), m2sci (преобразователь m-файла MATLAB в sce-файл Scilab), Browser Variables (браузер переменных, новый или старый).

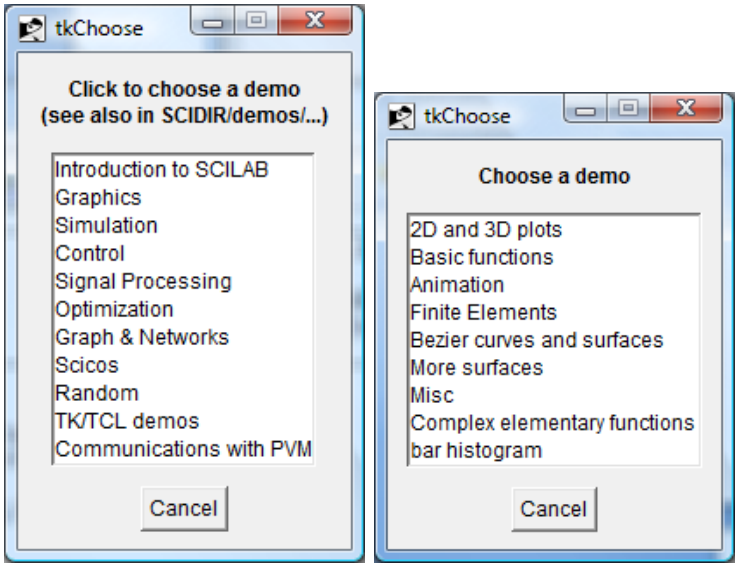


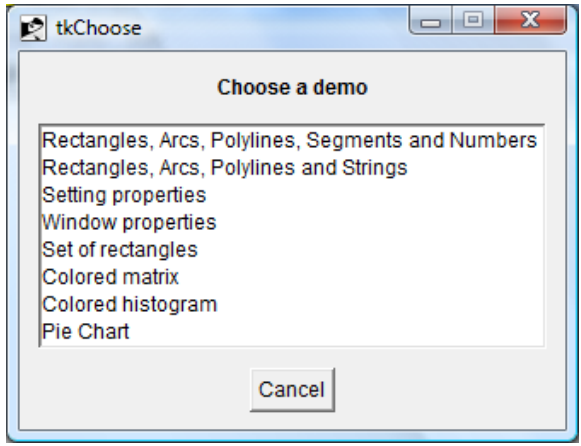
? – справка. Доступные команды: Scilab Help (Справка Scilab), Configure (Конфигурация), Scilab Demos (демонстрационные программы), Web Links (ссылки Интернета), About (о программе).



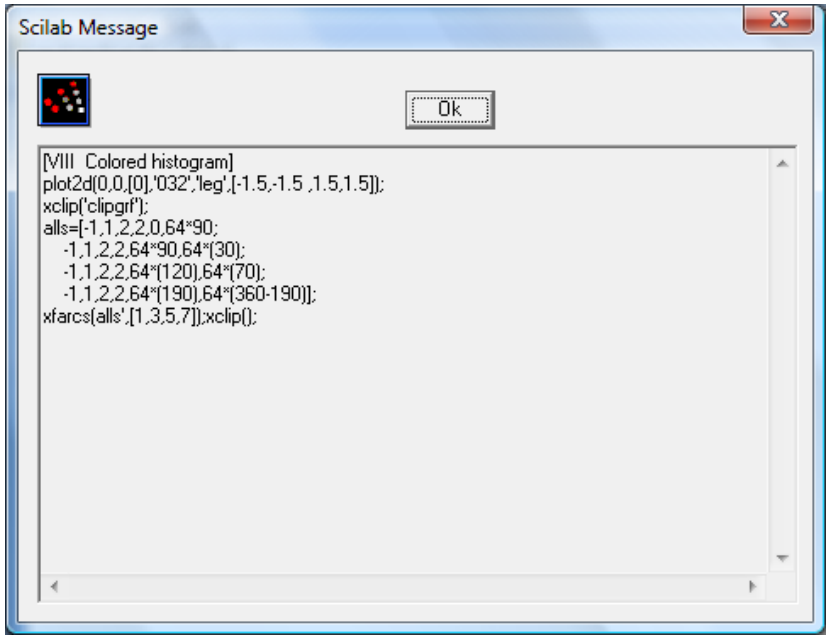
Пример 2

Для обращения к демонстрационным программам Scilab нужно использовать команду ?=>Scilab Demos. Открываются иерархически связанные окна выбора программы. В примере последовательно выбираются Graphics (графика), Basic funcions (основные функции), Pie Chart (Нарезанный торт).

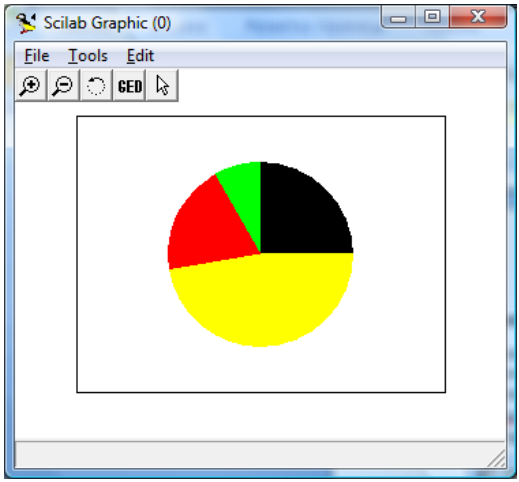




После выбора отображается окно сообщений Scilab с кодом исполняемой программы.



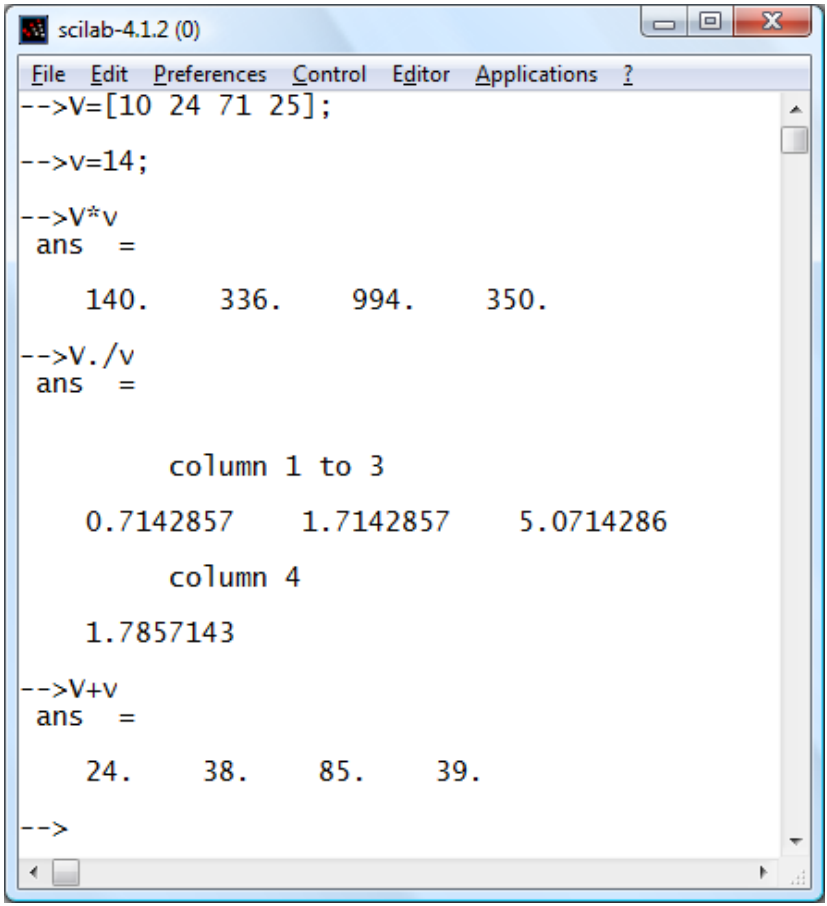
Нажатие кнопки Ok приводит к отображению графика, рисуемого по этой программе:



Пример 3

Операнд 1	Операнд 2	Операторы		
		1	2	3
V=[10 24 71 25]	v = 14	*	./	+

Командное окно Scilab после выполнения команд:


scilab-4.1.2 (0)
File Edit Preferences Control Editor Applications ?
-->V=[10 24 71 25];
-->v=14;
-->V*v
ans =
140. 336. 994. 350.
-->V./v
ans =
column 1 to 3
0.7142857 1.7142857 5.0714286
column 4
1.7857143
-->V+v
ans =
24. 38. 85. 39.
-->

2. Простые вычисления в Scilab

Предметная область

Для программирования сценариев в Scilab можно использовать комадное окно или редактор Scipad. Редактор предпочтителнее, так как он содержит встроенный отладчик.

В работе программируются вычисления двух функций. Результаты выводятся в виде двумерных графиков с использование графических функций высокого уровня.

Контрольные вопросы

1. Структура окна редактора Scilab.
2. Правила ввода команд.
3. Правила ввода функций и операндов.
4. Правила ввода выражений.
5. Организация циклов.
6. Правила ввода комментариев.
7. Правила просмотра результатов операций.
8. Правила создания двумерных графиков.
9. Запуск и отладка программ.

Задания к работе

Задание 1.

- Ввести в коде программы текст в виде комментария, как заглавие программы.
- Ввести исходные данные.
- Задать изменение аргумента.
- Вычислить значения функций 1 и 2 для аргумента в заданном интервале.
- Вывести графики функций одновременно на одном графике в декартовых координатах. Для разных графиков использовать разный тип линий.

Задание 2 . Повторить задание1, но графики функций вывести в двух подокнах на одном графике. Графики в столбиковом формате.

Задание 3 . Повторить задание1, но графики функций вывести в четырех подокнах с разными стилями линий на одном графике. Использовать функции plot2d, plot2d2, plot2d3, plot2d4.

Варианты заданий

№	Функция 1	Функция 2	a	b	h
1.	y = sin(x)	z= exp(x+3)/5000 - 1	-2pi	2pi	pi/20
2.	y = cos(x)	z = 0.00025e3-x - 0.6	-2pi	2pi	pi/20
3.	y = |tg(x)| + 0.1	z = (1+x)6	-2pi	2pi	pi/20
4.	y = (x2-1)/15	z = 1+sin(x)	-2pi	2pi	pi/20
5.	y = (x3-2)/15	z = 5cos(x)	-2pi	2pi	pi/20
6.	y = x2 - 10	z = 0.025exp(-1.2x)	-5	5	1
7.	y = 3sin(x)	z=0.015x3	-5	5	1
8.	y = 4sin(x)	z = 0.05x2	1	10	1

Методические указания

Текстовые пояснения в программу вводятся, как комментарий. Он начинается с символов //, которые располагается в первой позиции строки. Комментарий - это текст! В него не надо включать символы операций.

Для формирования XY графика необходимо:

- Задать аргумент в формате x=<начало>:<шаг>:<конец>.
- Вычислить функцию, например, y=f(x).

Вывести график процедурой plot(x,y,s). Процедура рисует график прямыми линиями между вычисленными точками. Здесь s - строковая константа, задающая параметры линии (цвет, тип точки, тип линии), ее можно пропускать (тогда параметры выбираются по умолчанию). Определены следующие значения s:

Цвет линии		Тип точки		Тип линии	
y	желтый	.	точка	-	сплошная
m	фиолетовый	o	кружок	:	двойной пунктир
c	голубой	x	крест	-.	штрих пунктир
r	красный	+	плюс	--	штрих
g	зеленый	*	звездочка		
b	синий	s	квадрат		
w	белый	d	ромб		
k	черный	^	треугольник вверх		
		v	треугольник вниз		
		<	треугольник влево		
		>	треугольник вправо		

Если на одном графике нужно отобразить несколько функций, например, (например, y1=f(x) и y2=f(x)) то они вначале вычисляются, а затем выводятся процедурой plot(x,y1,'s1',x,y2,'s2'...). В качестве параметров для каждой функции следуют строки символов 's', каждая из которых может содержать без раз-

делителей символы типа линии, типа точки, цвета линии в произвольной последовательности.

Для создания в графическом окне нескольких подокон для вывода графиков используется процедура subplot(m,n,p), где m - число подокон в окне по горизонтали, n – число подокон по вертикали, p - номер используемого подокна (нумерация с 1).

Для формирования графика в столбиковой форме (каждому вычисленному отсчету соответствует достаточно широкая вертикальная полоска) нужно использовать процедуру bar(x,y). При выводе такого графика в коде программы должны быть subplot(m,n,p) и bar(x,y).

Для формирования графика в форме с разными типами линий (сплошная, ступенчатая, вертикальные полоски, со стрелками) нужно использовать функции plot2d, plot2d2, plot2d3, plot2d4.

Пример 1

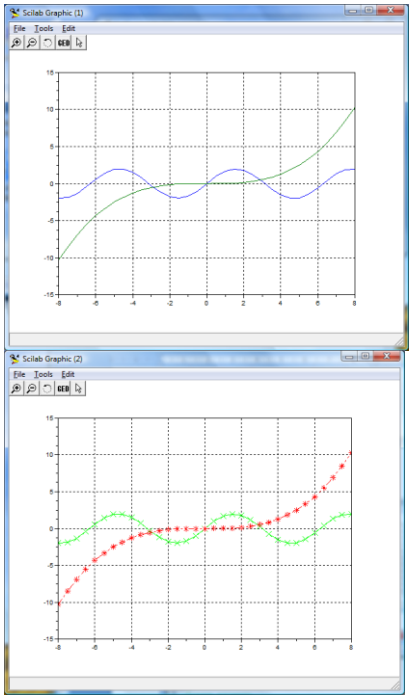
Задание

- Функция 1 y = 2*sin(x)
- Функция 2 z = 0.02*x3
- Начальное значение аргумента a = -8
- Конечное значение аргумента b = 8
- Шаг изменения аргумента h = 0.5

Листинг программы

```
// Диапазон и шаг
a=-8;
b=8;
h=0.5;
// Задание аргумента
X=a:h:b;
// Расчет функций
Y=2*sin(X);
Z=0.02*X.^3;
// Вывод с типами по умолчанию в окно 1
scf(1);                                  //Создать окно 1
plot(X,Y,X,Z);
xgrid()                                  //Включить координатную сетку
// Вывод с выбираемыми типами в окно 2
scf(2);                                  //Создать окно 2
plot(X,Y,'-gx',X,Z,':*r');
xgrid()                                  //Включить координатную сетку
```

Запуск программы командой Execute => Load Into Scilab. Результат – графики в отдельных окнах.



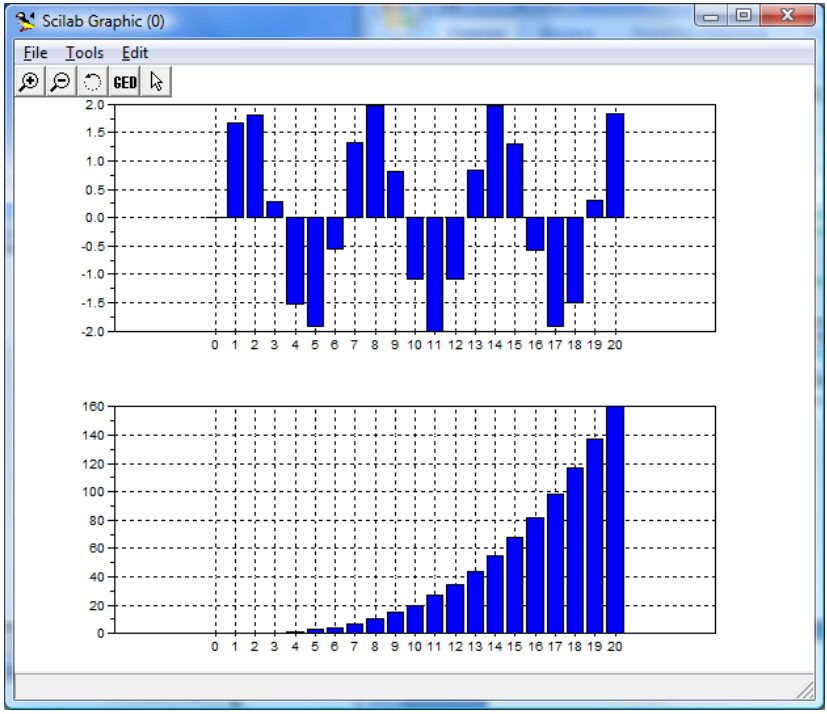
Пример 2

Листинг программы

```
// Подокна и функция bar
a=0;
b=20;
h=1;
// Задание аргумента
X=a:h:b;
// Расчет функций
Y=2*sin(X);
Z=0.02*X.^3;
// Вывод Y столбиками в подокно 1
subplot(2,1,1);
```

```
bar(X,Y);
xgrid()                                  //Включить координатную сетку
// Вывод Z столбиками в подокно 2
subplot(2,1,2);
bar(X,Z);
xgrid()                                  //Включить координатную сетку
```

Запуск программы командой Execute => Load Into Scilab. Результат – графики в отдельных подокнах общего окна.



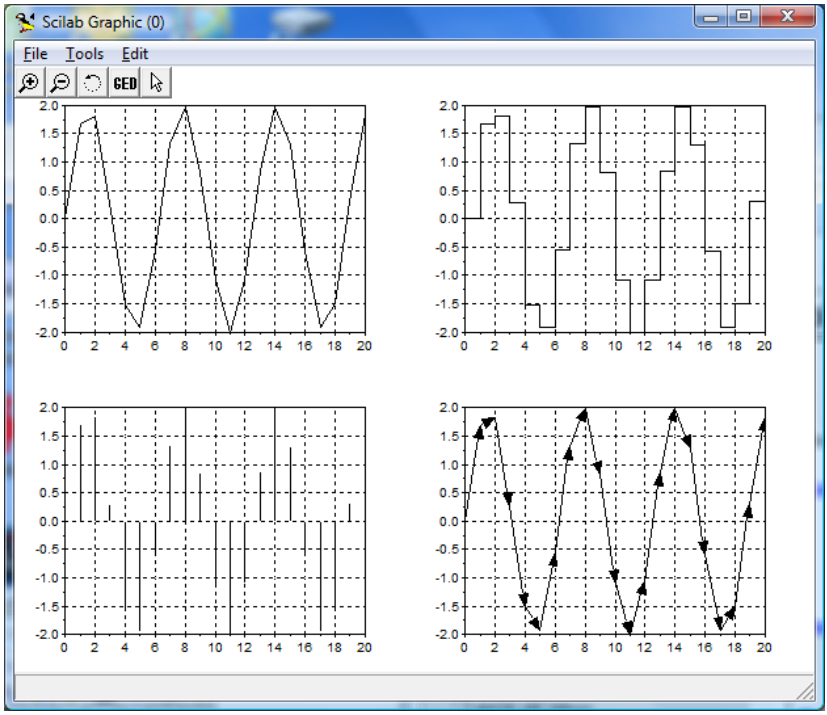
Пример 3

Листинг программы

```
// Подокна и функции со стилями линий
a=0;
b=20;
h=1;
```

```
// Задание аргумента
X=a:h:b;
// Расчет функций
Y=2*sin(X);
//Вывод Y в подокно 1
subplot(2,2,1);
plot2d(X,Y);
xgrid()          //Включить координатную сетку
// Вывод Y ступенькой в подокно 2
subplot(2,2,2);
plot2d2(X,Y);
xgrid()          //Включить координатную сетку
// Вывод Y вертикальыми полосками в подокно 3
subplot(2,2,3);
plot2d3(X,Y);
xgrid()          //Включить координатную сетку
// Вывод Y со стрелками в подокно 4
subplot(2,2,4);
plot2d4(X,Y);
xgrid()          //Включить координатную сетку
```

Запуск программы командой Execute => Load Into Scilab. Результат – графики в отдельных подокнах общего окна.


Scilab Graphic (0)
File Tools Edit
GED
2.0 1.5 1.0 0.5 0.0 -0.5 -1.0 -1.5 -2.0
0 2 4 6 8 10 12 14 16 18 20

3. Пространственные кривые в Scilab

Предметная область

Для программирования сценариев в Scilab можно использовать комадное окно или редактор Scipad. Редактор предпочтителнее, так как он содержит встроенный отладчик.

В работе программируются вычисления функций для пространственных кривых. Результаты выводятся в виде трехмерных графиков с использованием графических функций высокого уровня param3d и param3d1.

Контрольные вопросы

1. Организация вложенных циклов.
2. Правила задания многомерных функций.
3. Связь двумерной функции с матрицей для вывода графиков.
4. Трехмерная графика в аксонометрии.
5. Функция param3d.
6. Функция param3d1.

Задания к работе

Задание 1. Функции пространственных кривых (функция param3d).

- Ввести исходные данные.
- Вычислить координаты пространственной кривой.
- Вывести кривую в виде трехмерного графика.

Задание 2. Функции пространственных кривых (функция param3d1).

- Ввести исходные данные.
- Вычислить координаты двух пространственных кривых.
- Вывести кривые в виде трехмерного графика.

Варианты заданий. Заданы две функции расчета координат X, Y. В задании 1 использовать одну.

№	X	Y	t
7.	sin(t), sin(2t)	cos(t), cos(2t)	0:01:2*%pi
8.	sin(1,5t), sin(2t)	cos(1,5t), cos(2t)	0:01:3*%pi
9.	sin(2t), sin(3t)	cos(2t), cos(3t)	0:01:4*%pi
10.	sin(2,5t), sin(4t)	cos(2,5t), cos(4t)	0:01:5*%pi
11.	sin(t), sin(2t)	cos(t), cos(2t)	0:01:2,5*%pi
12.	sin(1,5t), sin(2t)	cos(1,5t), cos(2t)	0:01:3,5*%pi
13.	sin(2t), sin(3t)	cos(2t), cos(3t)	0:01:4,5*%pi

14.	sin(2,5t), sin(4t)	cos(2,5t), cos(4t)	0:01:5,5*%pi

Методические указания

Формирование задач. В работе предусмотрены 2 задачи, в каждой из которых вычисляется функции, описывающие пространственные кривые, и строятся поверхностные графики с использованием различных графических функций. В первой задаче рисуется олна кривая, во второй две.

Для формирования пространственного графика необходимо:

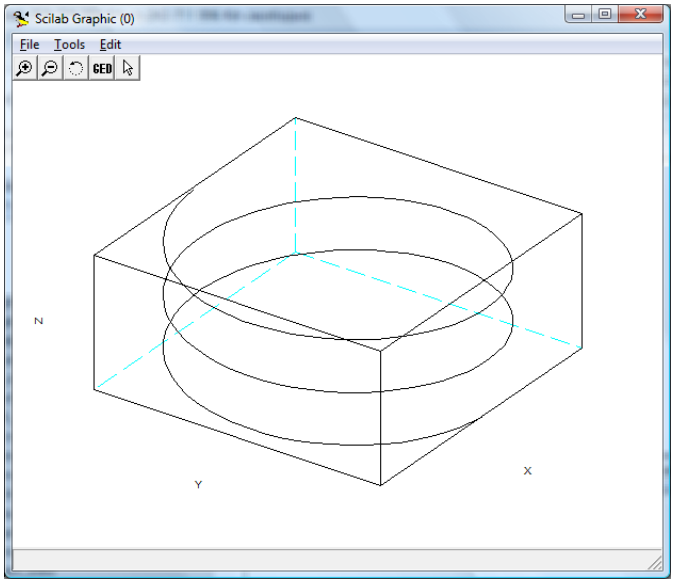
- задать число точек по координатам X и Y,
- вычислить векторв координонат X, Y.
- создать графическое окно и вывести туда график выбранного типа.

Пример 1

Кривая 1: x=sin(t), y=cos(t),
Используется функция param3d.

Листинг программы

```
// Функция param3d, пространственная кривая
t=0:0.1:5*%pi;
param3d(sin(t),cos(t),t/10,35,45,"X@Y@Z",[2,3])
```



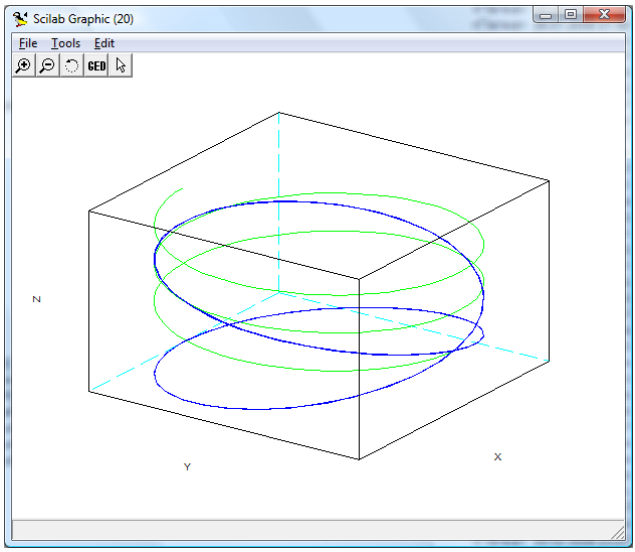
Пример 2

Кривая 1: sin(t), cos(t). Кривая 2: sin(2*t), cos(2*t).

Функция param3d1 рисует несколько пространственных кривых.

Листинг программы

```
// Функция param3d1, пространственные кривые
t=[0:0.1:5*%pi]';
param3d1([sin(t),sin(2*t)],[cos(t),cos(2*t)],..
  list([t/10,sin(t)],[3,2]),35,45,"X@Y@Z",[2,3])
```



4. Графика поверхностей в Scilab

Предметная область

Для программирования сценариев в Scilab можно использовать комадное окно или редактор Scipad. Редактор предпочтителнее, так как он содержит встроенный отладчик.

В работе программируются вычисления функций для поверхностей. Результаты выводятся в виде трехмерных графиков с использованием графических функций высокого уровня plot3d, mesh, surf, contour.

Контрольные вопросы

1. Правила задания многомерных функций.
2. Связь двумерной функции с матрицей для вывода графиков.
3. Трехмерная графика в аксонометрии.
4. Трехмерная графика с функциональной раскраской.
5. Контурная графика.

Задания к работе

Задание 1. Трехмерная графика (функции plot3d, mesh, surf, contour).

- Ввести исходные данные.
- Вычислить функцию.
- Вывести функцию в виде трехмерных графиков разного типа.

Задание 2. Повторить задание 1 с отображением графиков в подокных

Варианты заданий

№	Функция	Пределы изменения	
		x	y
1.	z=sin(x)cos(y)	от -2pi до 2pi	от -2pi до 2pi
2.	z=sin(x/2)cos(y)	от -2pi до 2pi	от -2pi до 2pi
3.	z=sin(2x)cos(y)	от -2pi до 2pi	от -2pi до 2pi
4.	z = sin(x)cos(y/2)	от -2pi до 2pi	от -2pi до 2pi
5.	z = sin(x/2)cos(2y)	от -2pi до 2pi	от -2pi до 2pi
6.	z = sin(2x)cos(2y)	от -2pi до 2pi	от -2pi до 2pi
7.	z = (1+sin(x)/x)(sin(y)/y)	от -2pi до 2pi	от -2pi до 2pi
8.	z = (sin(x)/x)cos(y)	от -2pi до 2pi	от -2pi до 2pi

Методические указания

Формирование задач. В работе предусмотрены 2 задачи, в каждой из которых вычисляется двумерная функция, описывающая объемную фигуру, и строятся поверхностные и контурные графики с использованием различных графических функций. В первой задаче каждый график выводится в свое окно, во второй в подокна общего окна.

Поверхностный и контурный графики. Для формирования поверхностного или контурного графика необходимо:

- задать число точек по координатам X иY,
- создать вложенные циклы по X и Y, вычислить функцию Z=f(X,Y),
- ввести номер графического окна, вывести туда график выбранного типа.

Следует использовать графики:

- трехмерный с аксонометрией, функция plot3(X,Y,Z),
- трехмерный с функциональной окраской, функция mesh(X,Y,Z),
- трехмерный с функциональной окраской и проекцией, функция surf(X,Y,Z),
- контурный, функция contour(X,Y,Z),

Пример 1

Задание

Функция $z = \frac{\sin(x)}{x} \cdot \frac{\sin(y)}{y}$.

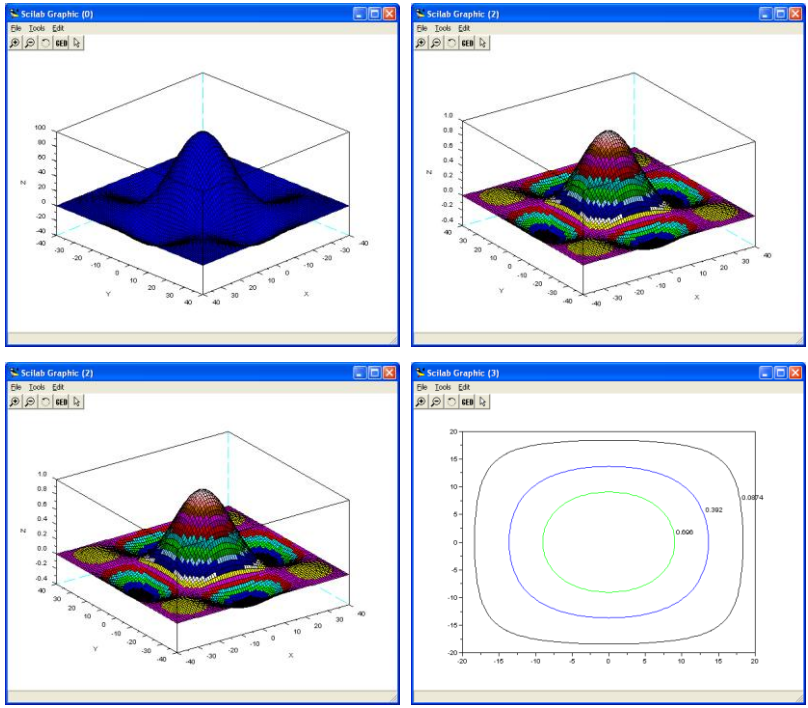
Пределы изменения аргументов -2pi...2pi

Листинг программы

```
// Функции plot3d, mesh, surf, contour
N=40;
h=%pi/20;
// Расчет матрицы
for n=1:2*N+1
 if n==N+1 A(n)=1; else A(n)=sin(h*(n-N-1))/(h*(n-N-1)); end;
end;
for n=1:2*N+1
   for m=1:2*N+1
      Z(n,m)=A(n)*A(m);
   end;
end;
// Задание площадки
X=-N:1:N;
```

```
Y=-N:1:N;
scf();
plot3d(X,Y,Z*100);  // Окно 1. 3d график с монотонной окраской
scf();
mesh(X,Y,Z);        // Окно 2. 3d график, каркас
scf();
surf(X,Y,Z);        // Окно 3. 3d график с функциональной окраской
scf();
contour(X,Y,Z,3);   // Окно 4. 3d график, контуры
```

Запуск программы командой Execute => Load Into Scilab. Результат – графики в отдельных окнах.



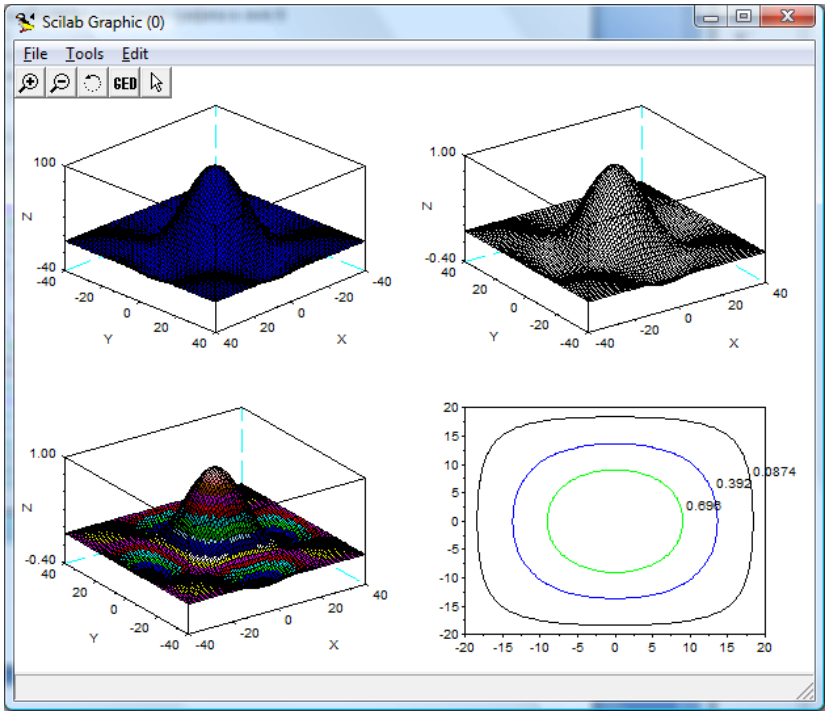
Пример 2

Листинг программы

```
// Функции plot3d, mesh, surf, contour. Используются подокна.
N=40;
h=%pi/20;
```

```
// Расчет матрицы
for n=1:2*N+1
 if n==N+1 A(n)=1; else A(n)=sin(h*(n-N-1))/(h*(n-N-1)); end;
end;
for n=1:2*N+1
   for m=1:2*N+1
      Z(n,m)=A(n)*A(m);
   end;
end;
// Задание площадки
X=-N:1:N;
Y=-N:1:N;
subplot(2,2,1); // Подокно 1. 3d график с монотонной окраской
plot3d(X,Y,Z*100);
subplot(2,2,2); // Подокно 2. 3d график, каркас
mesh(X,Y,Z);
subplot(2,2,3); // Подокно 3. 3d график с функциональной окраской
surf(X,Y,Z);
subplot(2,2,4); // Подокно 4. 3d график, контуры
contour(X,Y,Z,3);
```

Запуск программы командой Execute => Load Into Scilab. Результат – графики в отдельных подокнах общего окна.


Scilab Graphic (0)
File Tools Edit
GED
100
Z
-40
-40
-20
0
20
40
Y
40
20
0
-20
-40
X
1.00
Z
-0.40
40
20
0
-20
-40
Y
-40
-20
0
20
40
X
1.00
Z
-0.40
40
20
0
-20
-40
Y
-40
-20
0
20
40
X
20
15
10
5
0
-5
-10
-15
-20
-20 -15 -10 -5 0 5 10 15 20
0.0874
0.392
0.698

5. Решение системы линейных уравнений в Scilab

Предметная область

Решение системы линейных уравнений в Scilab осуществляется с использованием матричного деления. Результаты выводятся в командное окно.

Контрольные вопросы

1. Решение системы линейных уравнений с использованием матричного деления.
2. Ввод системы уравнений.
3. Вывод полученного решения
4. Проверка решения..

Задание к работе

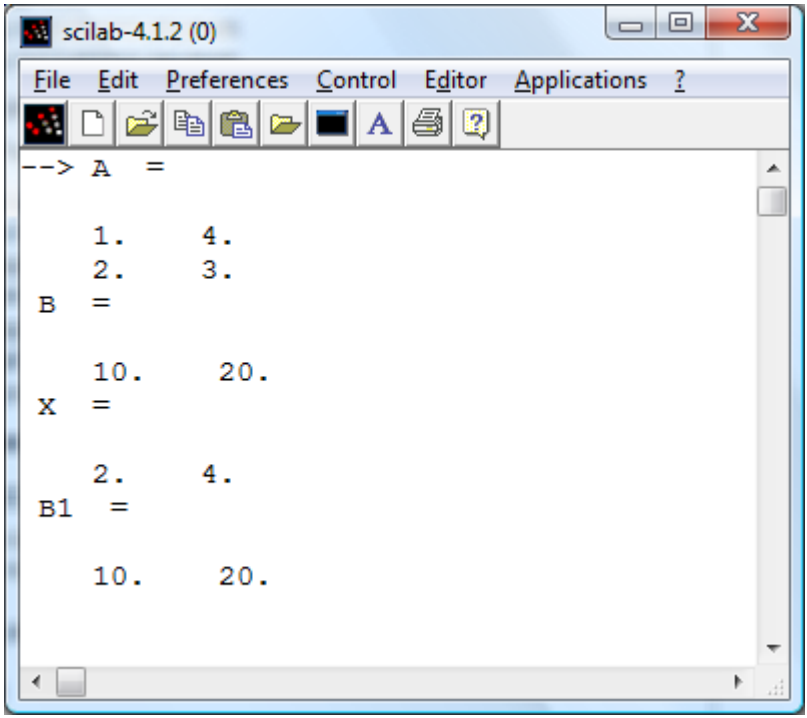
Задача 1. Решение системы линейных уравнениий.

- Создать программу решения системы из N линейных уравнений в редакторе scipad.
- Задать матрицу A коэффициентов. Коэффициенты уравнений размещаются по столбцам матрицы. Число строк в матрице равно N.
- Задать вектор правой части B размером N.
- Найти результат по формуле X=B/A.
- Проверить ответ по формуле B1=X*A. Должно получиться B1=B.

Пример 1

Листинг программы

```
// Решение системы линейных уравнений
// Матрица коэффициентов
A=[1,4;2,3]
// Вектор правой части
B=[10,20]
// Рещение
X=B/A
// Проверка
B1=X*A
```


scilab-4.1.2 (0)
File Edit Preferences Control Editor Applications ?
--> A =
1. 4.
2. 3.
B =
10. 20.
X =
2. 4.
B1 =
10. 20.

6. Решение нелинейных уравнений в Scilab

Предметная область

Решение нелинейного уравнений f(x)=0 в Scilab осуществляется с использованием функции fsolve. Решение ищется в окрестности предполагаемого значения x0. Для его определения проводится локализация решений по предварительно построенному графику f(x). Результаты выводятся в командное окно или в строку заголовка графика.

Контрольные вопросы

1. Задание функции пользователя.
2. Локализация решений уравнения.
3. Решение нелинейного уравнения с использованием функции fsolve.
4. Вывод полученных решений уравнения.
5. Локализация решений системы из двух уравнений.
6. Решение системы из двух уравнений.
7. Вывод полученных решений системы уравнений.

Задание к работе

Задача 1. Решение нелинейного уравнения.

- Создать программу решения нелинейного уравнений в редакторе scipad.
- В программе определить функцию f1(x).
- Вывести y1=f1(x) в виде XY графика. По нему определить приближенно корни уравнения y1(x)=0. Если корни на графике не просматриваются, то изменить пределы изменения аргумента и повторить операции.
- Для каждого корня найти точное значение, используя функцию fsolve. Перед расчетами задать приближенное значение корня x0.
- Сформировать строку с результатами и вывести ее в заголовок окна графика.

Задача 2. Решение системы из двух нелинейных уравнений.

- Создать программу решения нелинейных уравнений в редакторе scipad..
- В программе определить функции f1(x), f2(x), f3(x)=f2(x)-f1(x).
- Вывести y3=f1(x) в виде XY графика. По нему определить приближенно корни уравнения y3(x)=0. Если корни на графике не просматриваются, то изменить пределы изменения аргумента и повторить операции.

- Для каждого корня найти точное значение, используя функцию fsolve. Перед расчетами задать приближенное значение корня x0.
- Сформировать строку с результатами и вывести ее в заголовок окна графика.

Варианты заданий

№	f1(x)- полином 3-ей степени с коэффициентами a				f2(x)
	a3	a2	a1	a0	
1	0	-1	4	-1	0.2exp(x)-20
2	0	2	-2	-15	40|cos(x)|
3	0	1	4	-1	10ln(x+5.5)
4	0	9	-8	-70	100|sin(x)|
5	0	-4	4	50	70cos(x)
6	.1	-5	4	40	60exp(|0.1*x|)-100
7	.2	-3	2	30	20sin(2x)
8	.3	-6	1	50	exp(|x|)sin(2x)

Методические указания

При решении нелинейного уравнения оно формируется из функций задания, как f1(x)=0. При решении системы из двух нелинейных уравнений из функций задания формируется уравнение f3(x) = f1(x) - f2(x) = 0.

Локализация корней. Уравнение или система уравнений может иметь несколько корней, каждый из которых ищется отдельно. При этом для каждого корня надо задать диапазон аргумента, в котором он находится (только один!).

Это делается путем локализации корня. Для этого надо просчитать значения функций в заданном интервале и построить их графики. Начальное значение для решения одного уравнения - точка пересечения графиком функции оси X. График выводится процедурой, в которой аргументы - переменная x и анализируемая функция. С помощью grid on график делается с координатной сеткой:

plot(x,f1(x)) ;xgrid();

Начальное значение для решения системы из двух уравнений - точка взаимного пересечения графиков функций. Графики выводятся процедурой, в которой для каждого графика следует группа параметров:

plot(x,f(x),x,f2(x)); xgrid();

Функция fsolve. Используется для нахождения корня нелинейного уравнения. Формат этой функции:

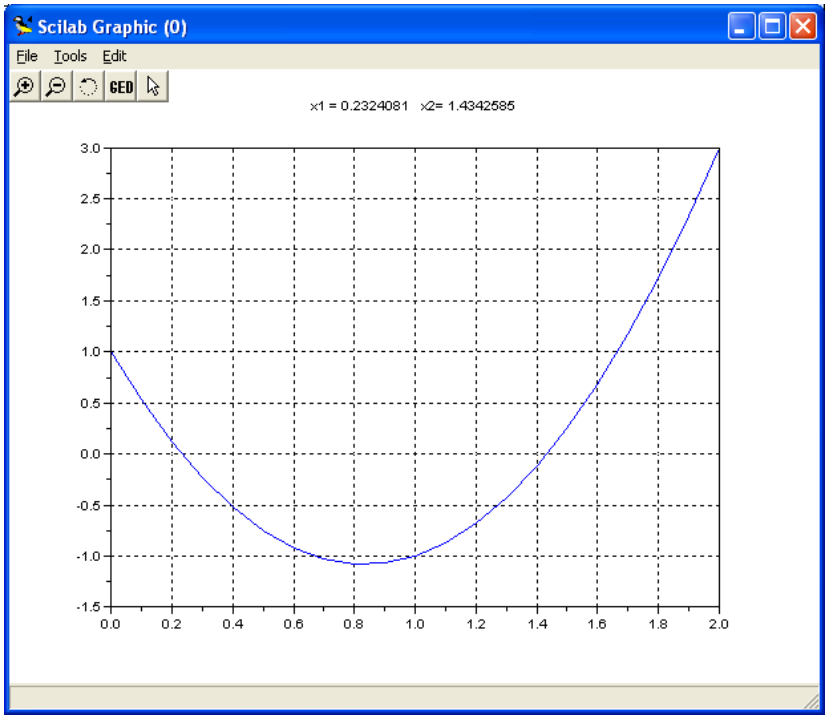
<имя результата>=fsolve(x0, f1)

Пример 1

Листинг программы

```
// Решение нелинейного уравнения
function y1=f1(x);      // Функция f1
      y1=x+3*(x-1)^2-2;
endfunction
x=0:0.1:2;
plot(x,f1(x)); xgrid;   // Графики
x0=0.2;
x1=fsolve(x0,f1)        // Корень 1
x0=1.4;
x2=fsolve(x0,f1)        // Корень 2
rezult='x1 = '+string(x1)+'   x2= '+string(x2);
title(rezult);
```

В программе ищутся 2 корня. Их приближенные значения 0.2 и 1.4 определены при пробном прогоне программы. Окончательный результат в окне графики.



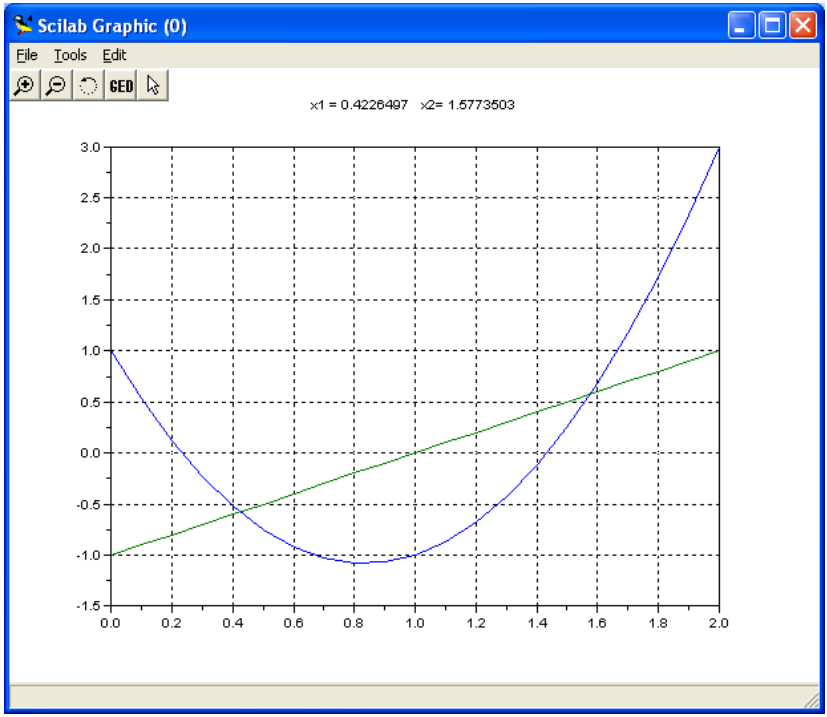
Пример 2

Листинг программы

```
// Решение системы нелинейных уравнений
function y1=f1(x);              // Функция f1
      y1=x+3*(x-1)^2-2;
endfunction
function y2=f2(x);              // Функция f2
      y2=x-1;
endfunction
function y3=f3(x);               // Функция f3
      y3=f1(x)-f2(x);
endfunction
x=0:0.1:2;
plot(x,f1(x),x,f2(x)); xgrid;    // Графики
x0=0.4;
x1=fsolve(x0,f3)                 // Корень 1
x0=1.5;
```

```
x2=fsolve(x0,f3)                    // Корень 2
rezult='x1 = '+string(x1)+'   x2= '+string(x2);
title(rezult);
```

В программе ищутся 2 корня. Их приближенные значения 0.4 и 1.5 определены при пробном прогоне программы. Окончательный результат в окне графики.



7. Работа с полиномами

Предметная область

В Scilab определены удобные средства работы с полиномами:

- Функция poly(V,'x','c'). Признак режима ‘c’ задает интерпертацию вектора V, как набора коэффициентов полинома. Функция формирует полином от переменной 'x', коэффициенты которого определены вектором V. В векторе V позиции слева направо соотвествуют степеням полинома, начиная с 0.
- Функция poly(V,'x','r'). Признак режима ‘r’ задает интерпертацию вектора V, как набора корней полинома. Крни полинома это корни уравнения f(x)=0, где f(x) – полином.. Функция формирует полином от переменной 'x', корни которого определены вектором V.
- Функция roots(p), возвращает корни полинома p.

Контрольные вопросы

1. Задание коэффициентов полинома.
2. Формирование полинома по коэффициентам степеней.
3. Корни полинома.
4. Задание корней полинома.
5. Вывод полученных результатов.
6. Вычисление еорней полинома.

Задание к работе

Задача 1. формирование полинома по коэффициентам степеней и корням полинома.

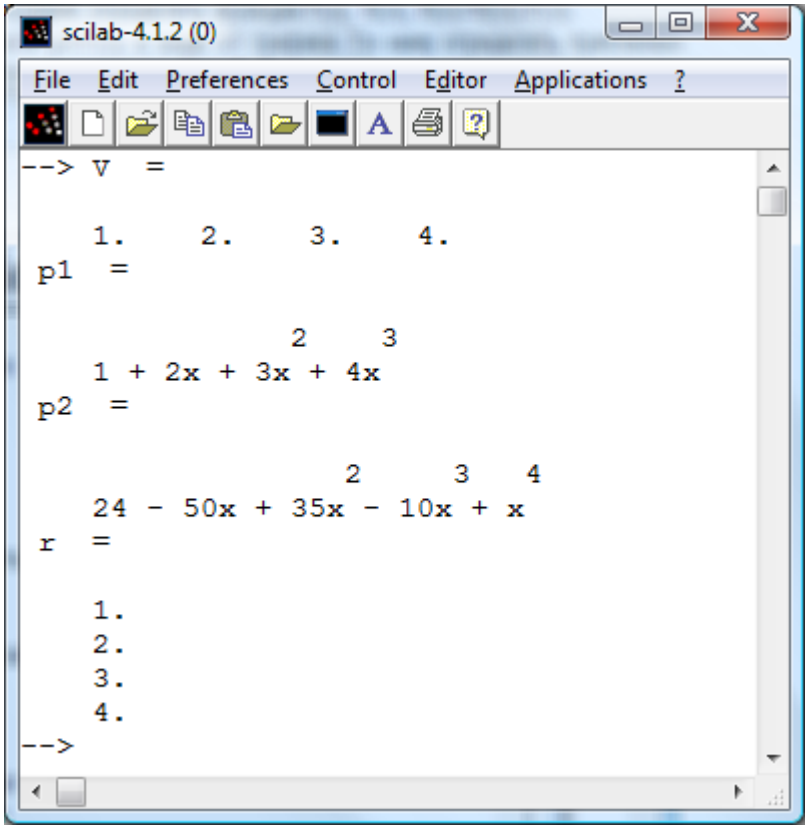
- Создать в редакторе scipad программу работы с полиномами.
- Задать вектор V.
- Добавить формирование полинома p1 по коэффициентам степеней.
- Добавить формирование полинома p2 по его корням.
- Для проверки p2 вычислить корни, используя функцию roots(p2).

Пример 1

Листинг программы

```
// Работа с полиномами
V=[1,2,3,4]
// формирование по коэффициентам степеней
p1=poly(V,'x','c')
// Формирование по корням полинома
```

```
p2=poly(V,'x','r')
// Вычисление корней полинома
r=roots(p2)
```



8. Моделирование устройства с помощью Scicos

Предметная область

Пакет моделирования объединенных объектов Scicos (Scilab Connected Object Simulator) служит для имитационного моделирования систем, состоящих из блоков с заданными свойствами (параметрами). Это функциональный аналог пакета Simulink в СКМ MATLAB.

В блоках Scicos используются сигналы:

- Регулярные. Это сигналы данных.
- Активизации. Они управляют работой блоков.

Блоки Scicos могут работать в режимах непрерывной или дискретной активизации. Первые не имеют входов активизации, вторые имеют их.

При дискретной активизации регулярный сигнал обновляется в моменты активизации и сохраняет значение между моментами активизации. При непрерывной активизации регулярный сигнал обновляется в каждый текущий момент времени.

Контрольные вопросы

1. Назначение Scicos.
2. Правила построения моделей в Scicos.
3. Правила моделирования в Scicos.
4. Структура иерархической библиотеки Scicos.
5. Блоки Sources -Источники.
6. Блоки Sinks - Получатели.
7. Блоки Linear - Линейные.
8. Блоки Non_Linear - Нелинейные.
9. Блоки Matrix -Матрицы.
10. Блоки Integer -Целые.
11. Блоки Events - События.
12. Блоки Threshold - Пороги.
13. Блоки Others - Другие.
14. Блоки Branching - Ветвления.
15. Блоки Electrical - Электрика.
16. Другие наборы блоков.

Задание к работе

Задача 1. Простая модель устройства.

Создать модель. В ней сигнал от источника поступает на функциональный блок. Регистратор с двумя входами позволяет наблюдать сигналы на входе и выходе функционального блока. Провести ее моделирование.

Задача 2. Расширенная модель устройства.

Создать модель. В ней к модели задачи 1 добавляется параллельная ветвь с вторым функциональным блоком. Регистратор с тремя входами позволяет наблюдать сигналы на входах и выходах функциональных блоков. Провести ее моделирование.

Варианты заданий

№	Источник сигнала	Блоки	
		Первый	Дополнительный
1.	Sinusoid Generator Синус	GainBlk Усиление	Deriv Дифференциатор
2.	Square Wave Generator Прямоугольные импульсы	Saturation Ограничитель	Integral Интегратор
3.	Sawtooth Generator Пила	Quant Квантизатор	GainBlk Усиление
4.	RMP Линейно нарастающий	Derivate Дифференциатор	Saturation Ограничитель
5.	Sinusoid Generator Синус	Integral Интегратор	Quant Квантизатор
6.	Square Wave Generator Прямоугольные импульсы	Time delay Задержка	Deriv Дифференциатор
7.	Sawtooth Generator Пила	DeadBand Мертвая зона	Integral Интегратор
8.	RMP Линейно нарастающий	Gain Усиление	Time delay Задержка

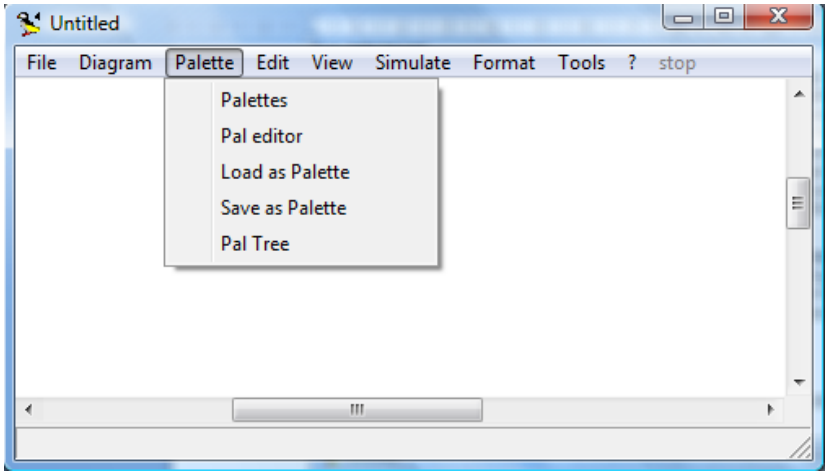
В таблице названия функциональных блоков даны на русском и английском языках (так, как они названы в браузере библиотеки блоков Scicos).

Методические указания

Модель устройства содержит источник сигнала, функциональные блоки и средства наблюдения за поведением системы (дисплей, численный индикатор и др.).

Во всех вариантах задания нужно использовать дисплей с одним входом в задаче 1 и с двумя входами в задаче 2.

Для создания модели используется редактор, который вызывается из Scilab командой Scicos() в консоли или командой Applications=>Scicos из главного меню. Библиотека Scicos содержит подбиблиотеки блоков разных предметных областей и назначения. Доступ к подбиблиоткам через команду из пункта Palette главного меню редактора.



Команда Palette=>Palette отображает список подбиблиотек для выбора:

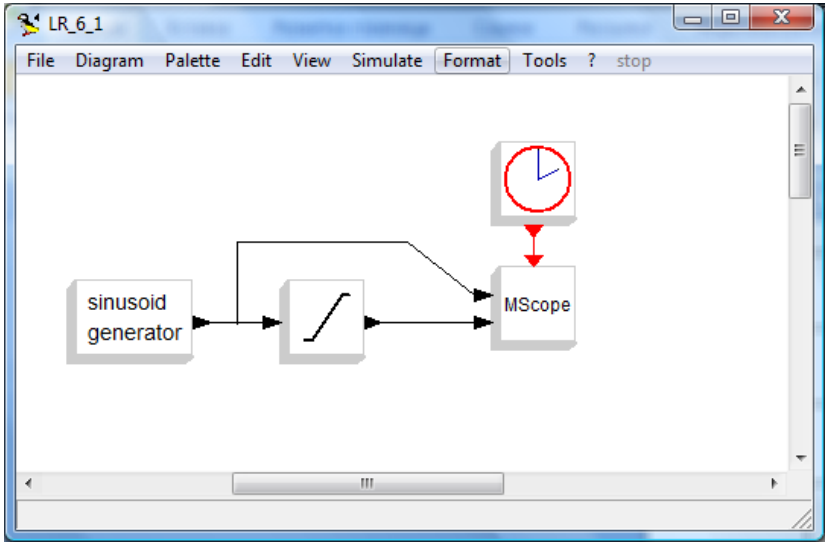
Разместите окна браузера и модели таким образом, чтобы они не перекрывали друг друга. Теперь можно формировать модель визуальным методом. Скопируйте мышью из браузера в окно модели нужные блоки и удобно разместите их. При переносе блока в модель там создается экземпляр блока с именем, совпадающим с надписью под блоком (при необходимости, когда однотипных блоков в модели несколько, в имя блока добавляется номер).
Соедините блоки коннекторами. Для этого нужно протаскивать мышь от одной соединяемой точки к другой. При отпускании кнопки мыши в модели отображается коннектор со стрелкой.
Установите для каждого блока свойства. Для этого нужно на блоке сделать двойной щелчок мышью, что приведет к появлению окна со свойствами блока. Установите нужные свойства в полях окна.

Пример 1

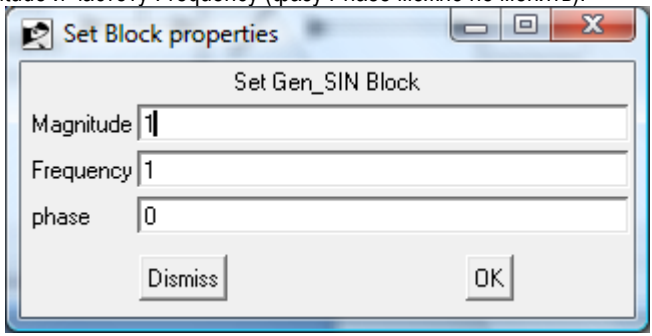
Задача 1. Двусторонний ограничитель синусоидального сигнала.

- Создать модель.
- Провести ее моделирование.

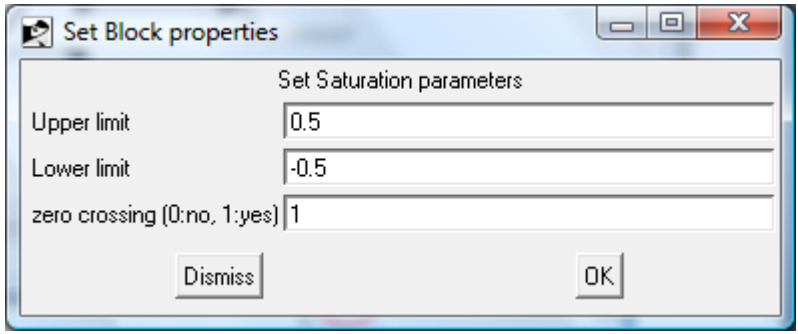
Создать на экране дисплея пустое окно модели и вызвать браузер библиотеки блоков. Из нее перенести в модель нужные блоки. Затем коннекторами соединяем блоки.



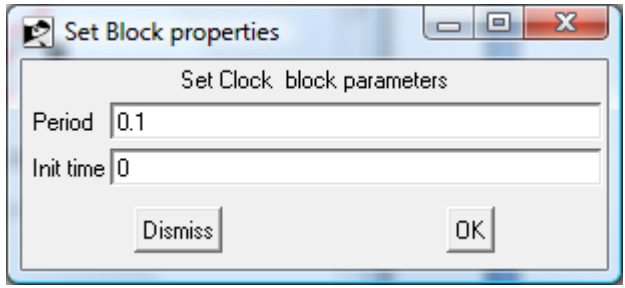
Двойным щелчком по блоку генератора синусоиды вызвать окно со свойствами блока. В его полях выбрать параметры. В данном случае установить амплитуду Magnitude и частоту Frequency (фазу Phase можно не менять).



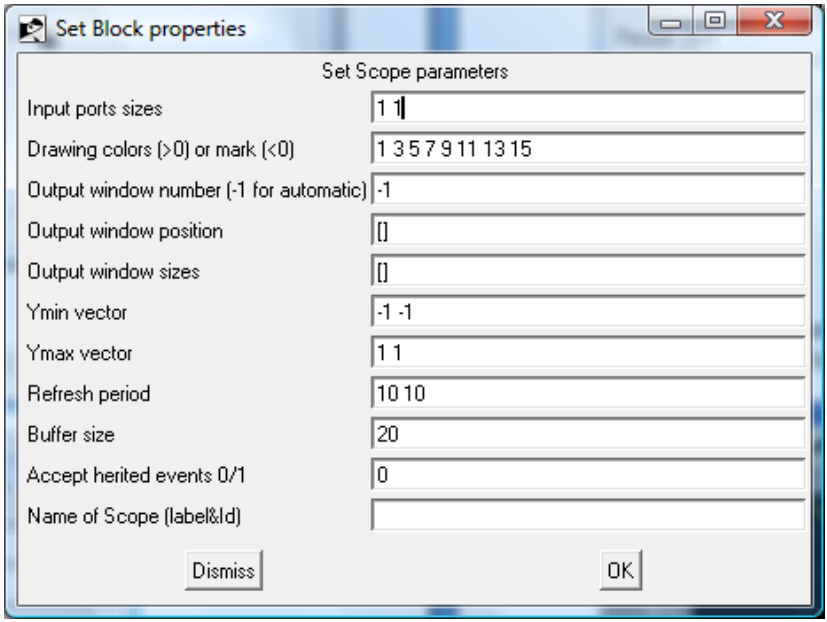
Двойным щелчком по блоку Saturation в модели вызвать окно со свойствами блока. В нем установить верхний и нижний пределы ограничения.



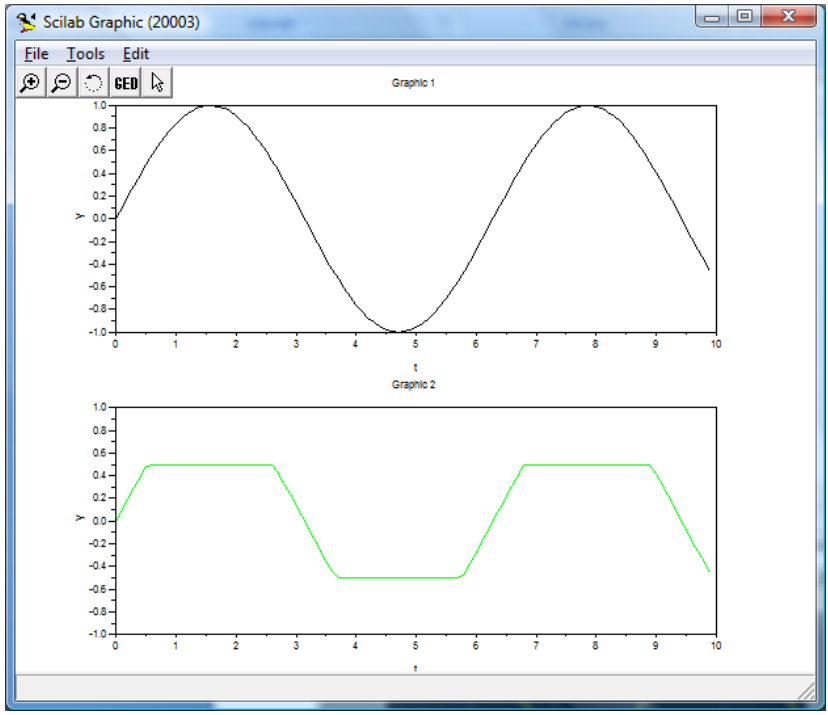
Двойным щелчком по блоку Clock в модели вызвать окно со свойствами блока. В нем установить период дискретизации Period и начальное время Init time.



Двойным щелчком по блоку Scope в модели вызвать окно со свойствами блока. В нем установить параметры. В поле Input ports sizes ввести вектор [1 1], задающий два поля в графическом окне блока. С остальными параметрами можно согласиться.



Включить симулирование (моделирование) командой Simulate => Run. В окне графики отображаются графики сигналов.



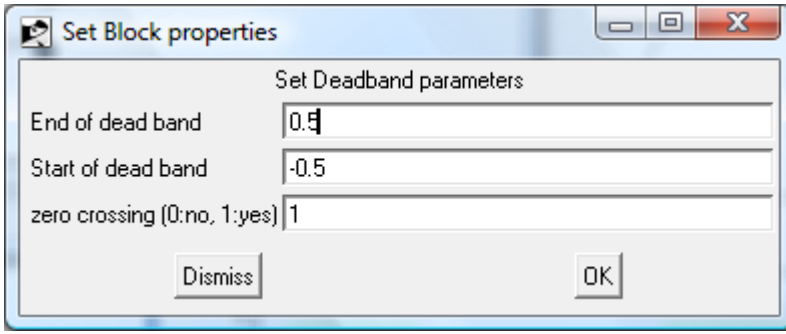
Если результат не совпадает с ожидаемым, то нужно изменить параметры модели.

Пример 2

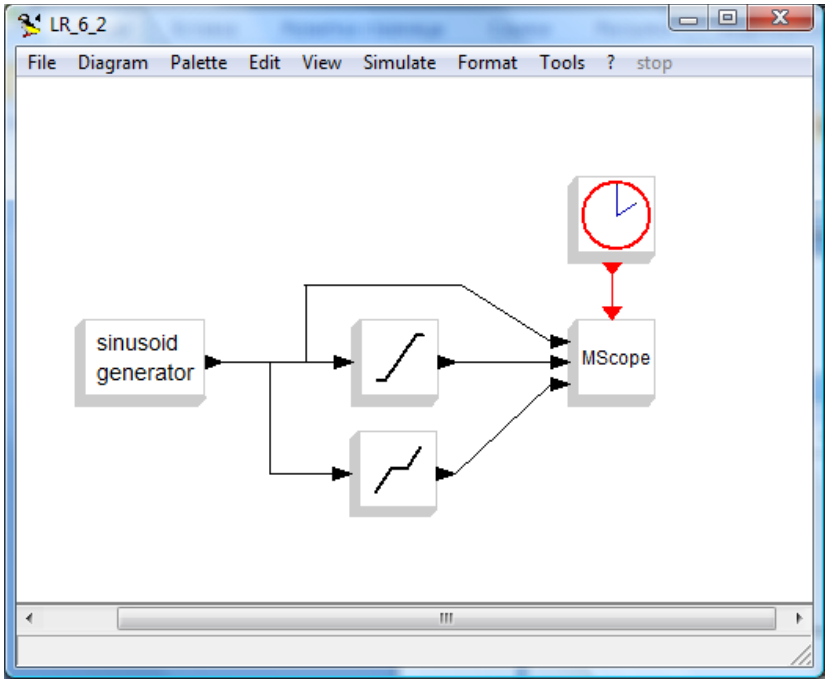
Задача 2. Двусторонний ограничитель синусоидального сигнала и блок мертвой зоны

- Создать модель.
- Провести ее моделирование.

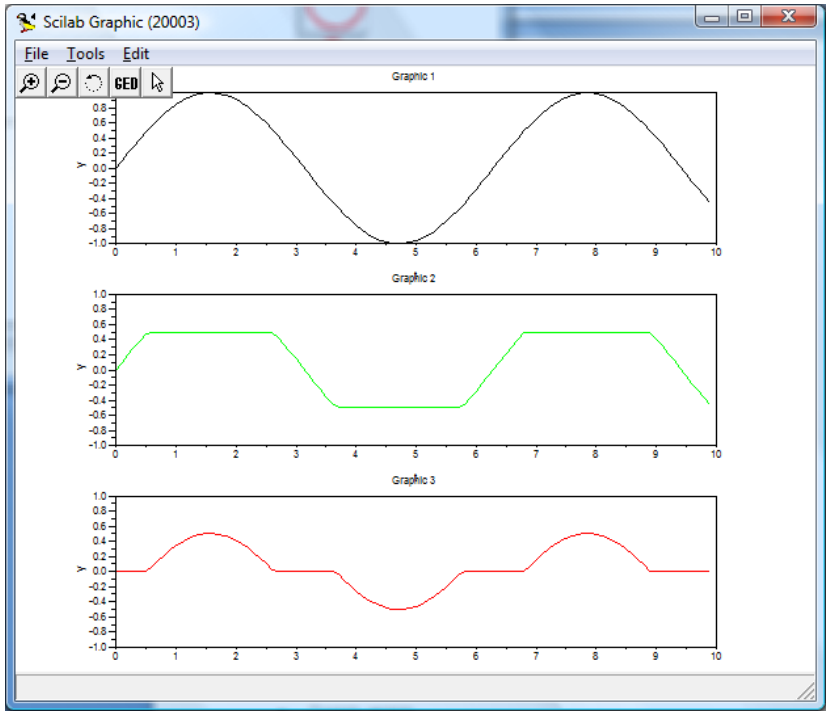
Добавим в модель задачи 1 в параллельную ветвь блок Deadband (Мертвая зона). Установим для него начало (Start of dead band) и конец (End of dead band) мертвой зоны



У регистратора MScope изменим число осей на 3.



Включить симулирование (моделирование) командой Simulate => Run. В окне графики отображаются графики сигналов.



Если результат не совпадает с ожидаемым, то нужно изменить параметры модели.

9. Моделирование логики с помощью Scicos

Предметная область

Пакет моделирования объединенных объектов Scicos (Scilab Connected Object Simulator) служит для имитационного моделирования систем, состоящих из блоков с заданными свойствами (параметрами). Это функциональный аналог пакета Simulink в СКМ MATLAB.

В блоках Scicos используются сигналы:

- Регулярные. Это сигналы данных.
- Активизации. Они управляют работой блоков.

Блоки Scicos могут работать в режимах непрерывной или дискретной активизации. Первые не имеют входов активизации, вторые имеют их.

При дискретной активизации регулярный сигнал обновляется в моменты активизации и сохраняет значение между моментами активизации. При непрерывной активизации регулярный сигнал обновляется в каждый текущий момент времени.

Контрольные вопросы

1. Назначение Scicos.
2. Правила построения моделей в Scicos.
3. Правила моделирования в Scicos.
4. Блок комбинационной логики.
5. Блок счетчика по модулю.

Задание к работе

Задача 1. Модель комбинационной логики.

Создать модель с блоком комбинационной логики с известной таблицей истинности. Блок использует два входных сигнала. Каждый сигнал формируется с использованием соединенных каскадно генератора прямоугольных исмпульсов, ограничителя и преобразователя формата чисел double в формат Integer.

Регистратор с тремя входами позволяет наблюдать сигналы на входах и выходе логического блока. Для выхода логического блока надо добавить преобразователь формата Integer в формат double. Провести моделирование.

Задача 2. Модель счетчика по модулю.

Создать модель со счетчиком. Счетчик получает импульсы от источника, в качестве которого используются часы. На каждый имульс содержимое счетчика увеличивается на 1, а при достижения модуля сбрасывается на 0. Регистратор

с одним входом позволяет наблюдать сигнал на входе и выходе счетчика. Провести ее моделирование.

Варианты заданий

№	Тип логики	Модуль счета
1.	Sinusoid Generator Синус	8
2.	Square Wave Generator Прямоугольные импульсы	10
3.	Sawtooth Generator Пила	12
4.	RMP Линейно нарастающий	16
5.	Sinusoid Generator Синус	10
6.	Square Wave Generator Прямоугольные импульсы	13
7.	Sawtooth Generator Пила	9
8.	RMP Линейно нарастающий	7

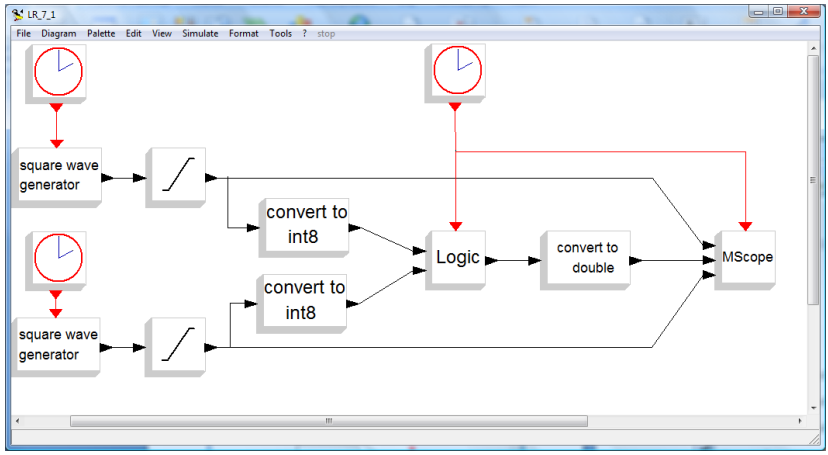
В таблице названия функциональных блоков даны на русском и английском языках (так, как они названы в браузере библиотеки блоков Scicos).

Пример 1

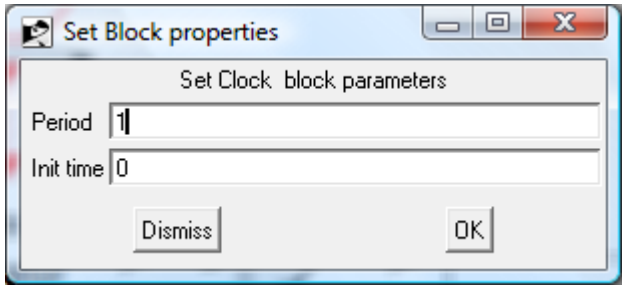
Задача 1. Логическое устройство.

- Создать модель.
- Провести ее моделирование.

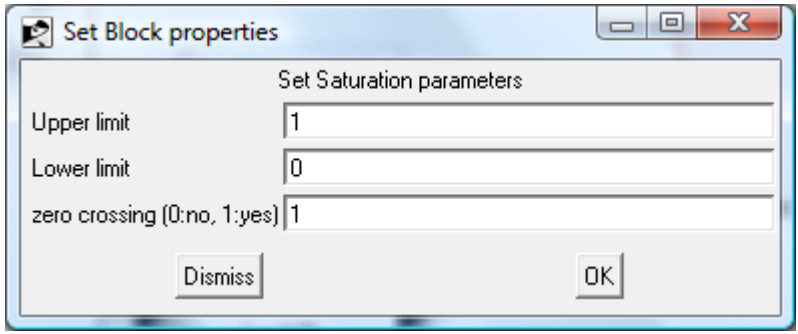
Создать на экране дисплея пустое окно модели и вызвать браузер библиотеки блоков. Из нее перенести в модель нужные блоки. Затем коннекторами соединяем блоки.



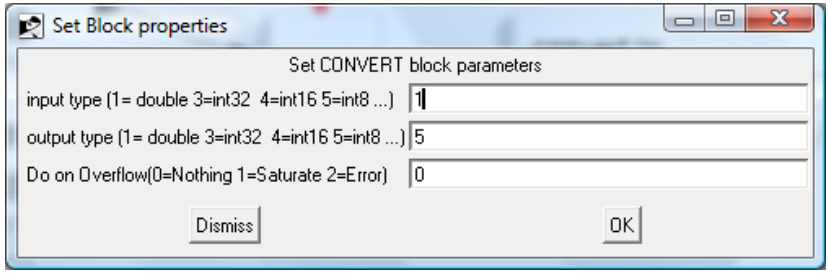
Двойным щелчком по блоку часов, соединенным с генератором прямоугольника вызвать окно со свойствами блока. В его полях выбрать параметры. В данном случае установить период Period (начальное время Init time можно не менять). Для одного генератора период равен 1, для второго 2.



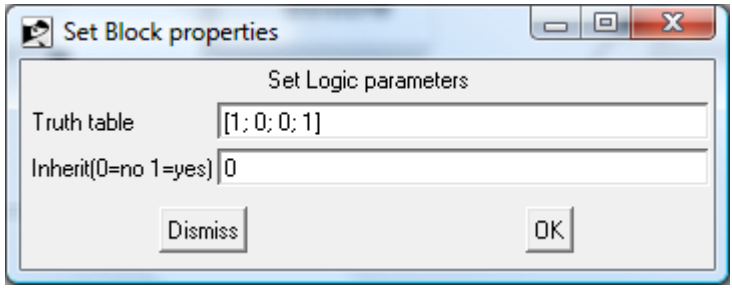
Двойным щелчком по блоку Saturation в модели вызвать окно со свойствами блока. В нем установить верхний и нижний пределы ограничения. Для двух блоков параметры одинаковы.



Двойным щелчком по блоку Convert в модели вызвать окно со свойствами блока. В нем установить типы данных на входе и выходе. На входе тип doble (номер 1), на выходе Int8 (номер 5). Для двух блоков параметры одинаковы.

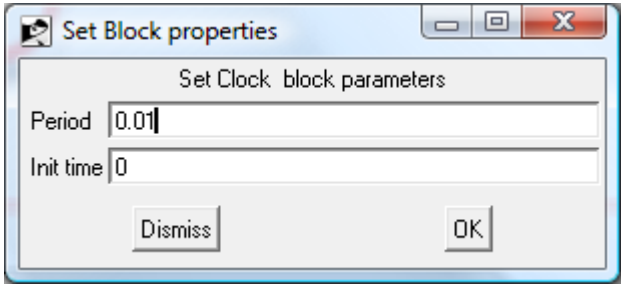


Двойным щелчком по блоку Logic в модели вызвать окно со свойствами блока. В нем установить данные из таблицы истинности. Это матрица строк, в которой задаются выходы блока для возможных входных наборов. В примере логика – совпадение (На выходе 1, если на входах 00 или 11).

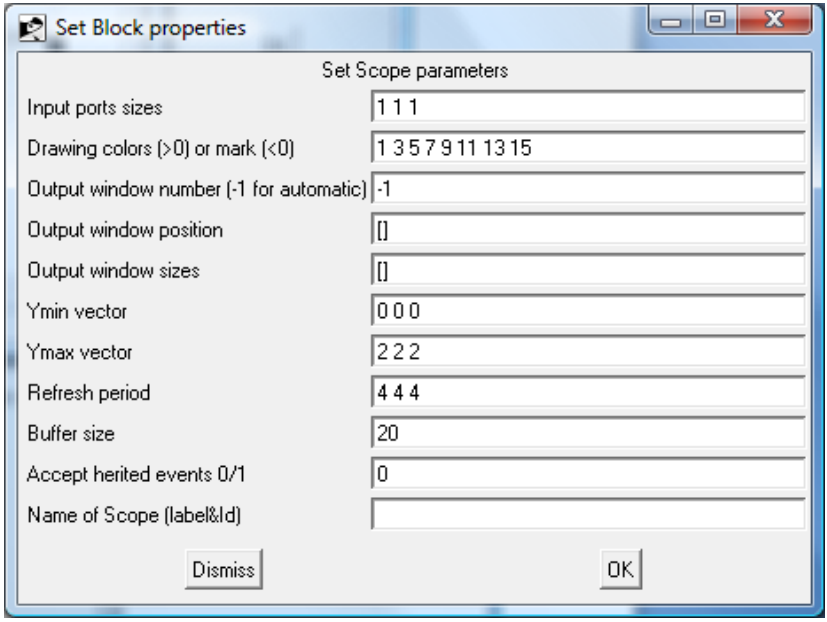


Двойным щелчком по блоку Clock в модели вызвать окно со свойствами блока. В нем установить период дискретизации Period и начальное время Init time.

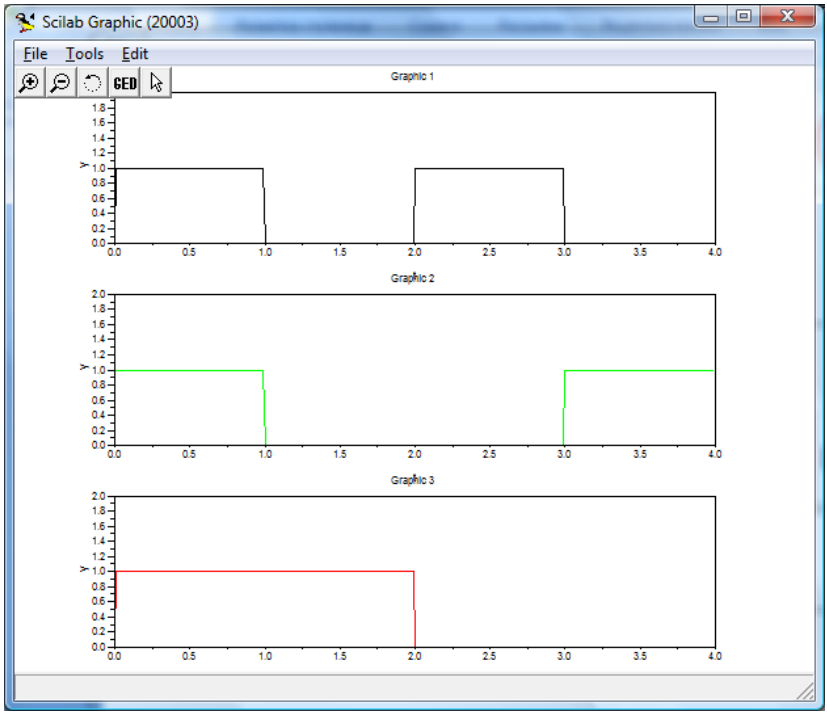
Период нужно взять значительно меньше, че периеоды генераторов импульсов, чтобы графики получались не изломанными



Двойным щелчком по блоку Scope в модели вызвать окно со свойствами блока. В нем установить параметры. В поле Input ports sizes ввести вектор [1 1 1], задающий три поля в графическом окне блока. С остальными параметрами можно согласиться.



Включить симулирование (моделирование) командой Simulate => Run. В окне графики отображаются графики сигналов.

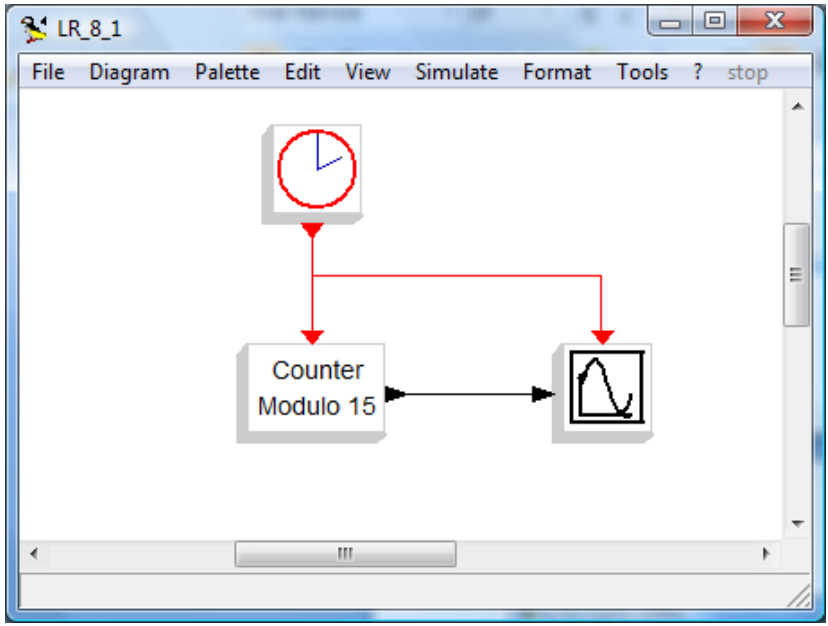


Если результат не совпадает с ожидаемым, то нужно изменить параметры модели.

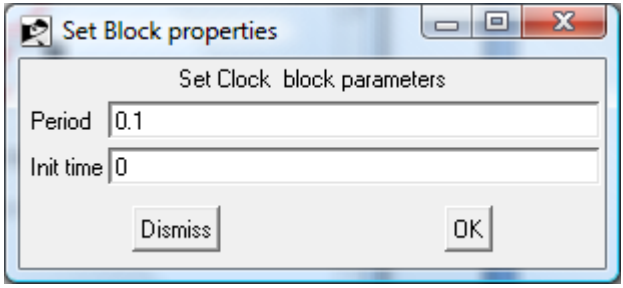
Задача 2. Счетчик по модулю 15.

- Создать модель.
- Провести ее моделирование.

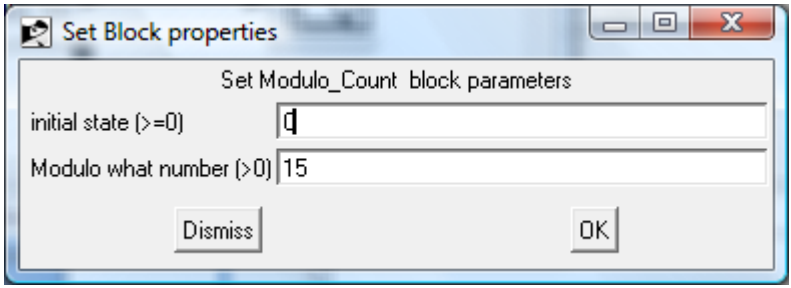
В модель включаем блоки часов, счетчик по модулю 15 и наблюдатель Scope.



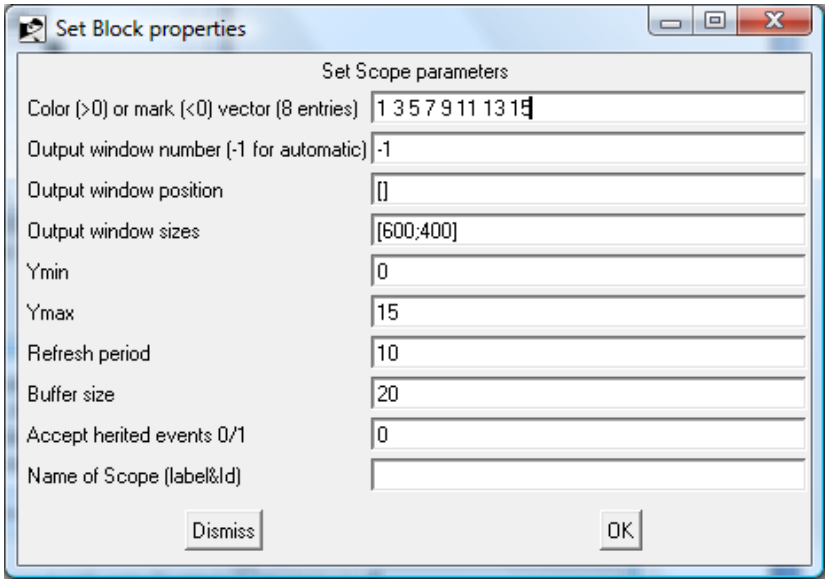
Двойным щелчком по блоку Clock в модели вызвать окно со свойствами блока. В нем установить период дискретизации Period и начальное время Init time.



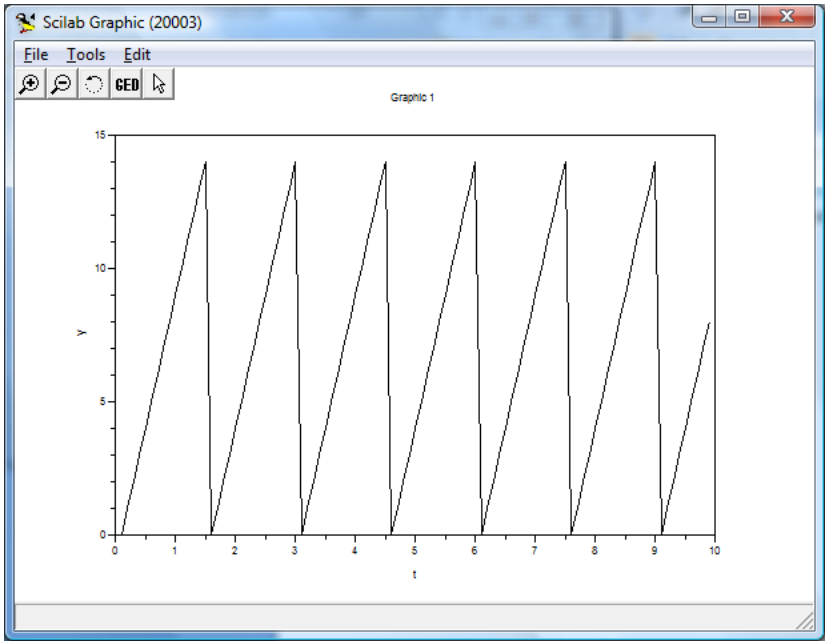
Двойным щелчком по блоку Modulo_Count в модели вызвать окно со свойствами блока. В нем установить модуль счета.



Двойным щелчком по блоку Scope в модели вызвать окно со свойствами блока. В нем установить параметры. Размеры графика определяются полями Ymin=0 и Ymax=15. В поле Refresh Period задаем размер графика по горизонтали (в примере 10). С остальными параметрами можно согласиться.



Включить симулирование (моделирование) командой Simulate => Run. В окне графики отображается график сигнала. Видно, что счетчик нарашивает содержимое от 0 до 15, после чего сбрасывается на 0.



Если результат не совпадает с ожидаемым, то нужно изменить параметры модели.

10. Графический интерфейс пользователя

Предметная область

Средства графического пользовательского интерфейса ГИП (GUI – Graphic User Interface) предназначены для создания в Scilab приложений с пользовательским интерфейсом. Определены две возможности:

- TCL/Tk interface. Интерфейс TCL/Tk. Эти средства предназначены для конструирования графического окна с использованием набора управляющих компонентов.
- GUI and Dialogs. ГИП и диалоги. Эти средства предназначены для работы с графическим редактором Scilab.

TCL (от англ. Tool Command Language — «Командный язык инструментов», читается «тикль») — скриптовый язык высокого уровня. TCL часто применяется совместно с графической библиотекой Tk (Tool Kit). Связку TCL/Tk по-русски иногда называют «Так-тикль».

Контрольные вопросы

1. Назначение ГИП.
2. PushButton - Кнопка
3. CheckBox
4. Radiobutton

Задание к работе

Задача 1. ГИП с независимыми переключателями

Создать ГИП с двумя независимыми переключателями, которым соответствуют обработчики, рисующие в графическом окне функции f1(x) и f2(x). Команда рисования – нажатие кнопки. В ГИП рисуются все выбранные графики.

Задача 2. ГИП с зависимыми переключателями (радиокнопками)

Создать ГИП с двумя радиокнопками, которым соответствуют обработчики, рисующие в графическом окне функции f1(x) и f2(x). Команда рисования – нажатие кнопки. В ГИП рисуется один выбранный график.

Варианты заданий

№	f1(x)	f2(x)
1.	Синус	Прямоугольные импульсы
2.	Прямоугольные импульсы	Пила
3.	Пила	Линейно нарастающий
4.	Линейно нарастающий	Косинус

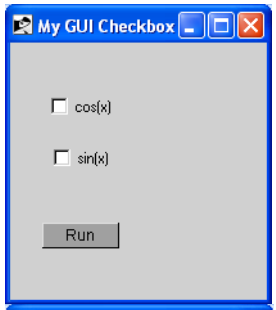
5.	Косинус	Прямоугольные импульсы
6.	Прямоугольные импульсы	Пила
7.	Пила	Линейно нарастающий
8.	Линейно нарастающий	Косинус

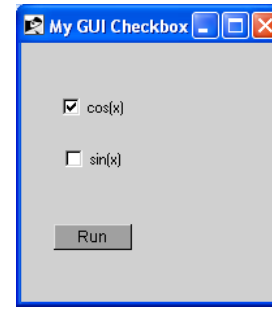
Пример 1

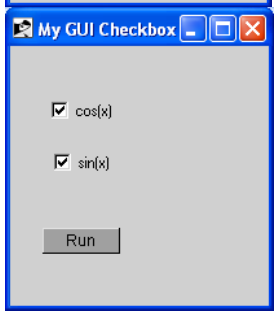
Листинг программы

```
// GUI с переключателями
f=figure('Position',[50 50 200 200]); // Создание графического окна
set(f,'figure_name','Ve GUI');
// Создание переключателей
hCBox1=uicontrol(…
      'Style','checkbox',…
      'String','sin(x)'…
      ,'value',0,…
      'Position',[25,100,60,20]);
hCBox2=uicontrol(…
      'Style','checkbox',…
      'String','cos(x)',…
      'value',0,...
      'Position',[25,140,60,20]);
hPButton=uicontrol(…
      'Style','pushbutton',…
      'Position',[25,80,60,20],...
      'String','Run',…
      'callback','Run');
// Функция Run, реакция на событие 'callback'
function Run()
   x=-2*%pi:0.1:2*%pi;
   y1=sin(x);
   y2=cos(x);
   pr=0;pr1=0; pr2=0; // Активных нет
   if get(hCBox1,'value')==1 then pr1=1, end; // Активен 1
   if get(hCBox2,'value')==1 then pr2=1, end; // Активен 2
   if pr1*pr2>0 then pr=3
      elseif pr1==1 pr=1
            elseif pr2==1 pr=2, end;
   if pr==1 then newaxes, plot(x,y1,'r'),xgrid(),end;          //Построение sin(x)
   if pr==2 then newaxes, plot(x,y2,'g'),xgrid(),end;          //Построение cos(x)
   if pr==3 then newaxes, plot(x,y1,x,y2),xgrid(),end;       //Построение sin(x), cos(x)
endfunction
```

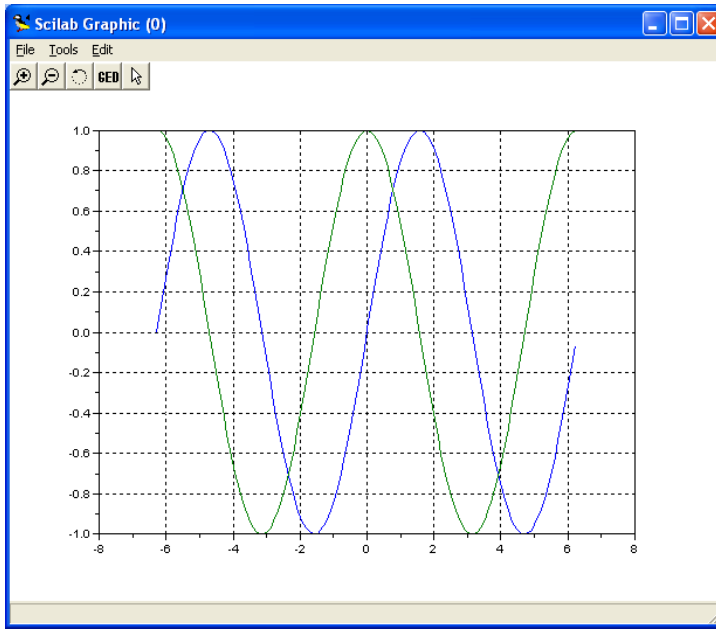
Ниже три снимка окна при разных состояниях объектов ‘Checkbox’.







Пример графика, рисуемого в третьем случае:



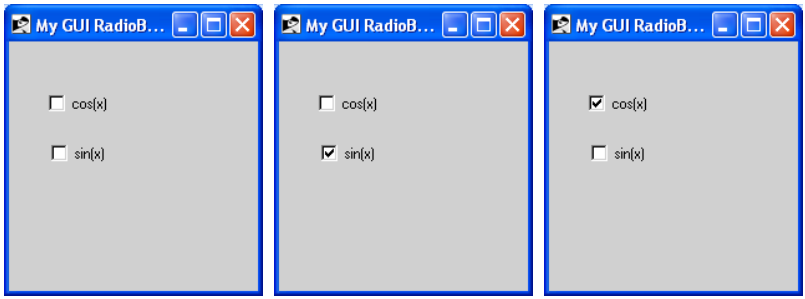
Пример 2

Листинг программы

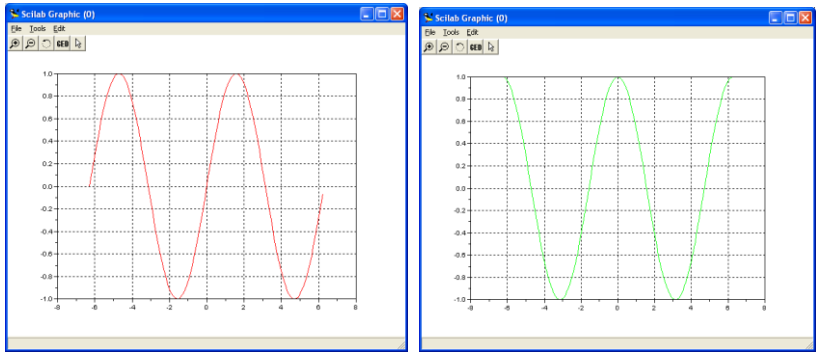
```
// GUI с радиокнопками
f=figure('Position',[50 50 200 200]); // Создание графического окна
set(f,'figure_name','My GUI Radiobuttons');
// Создание радиокнопок
hRb1=uicontrol(…
      'Style','radiobutton',…
      'String','sin(x)','value',0,...
      'Position',[25,100,60,20],…
      'callback','Radio1');
hRb2=uicontrol(…
      'Style','radiobutton',…
      'String','cos(x)',…
      'value',0,...
      'Position',[25,140,60,20],…
      'callback','Radio2');
x=-2*%pi:0.1:2*%pi;
// Функции Radio, реакция на событие 'callback'
```

```
function Radio1()
    newaxes;
    if get(hRb1,'value')==1 then        // Активна 1
        set(hRb2,'value',0);            // Сброс альтернативной кнопки
        plot(x,sin(x),'r'),xgrid();     // Построение sin(x)
    end;
    plot(x,sin(x),'r'),xgrid();   // Построение sin(x)
endfunction
function Radio2()
    newaxes;
    if get(hRb2,'value')==1 then        // Активна 1
        set(hRb1,'value',0);            // Сброс альтернативной кнопки
    end;
    plot(x,cos(x),'g'),xgrid();         // Построение cos(x)
endfunction
```

Ниже три снимка окна при разных состояниях объектов ‘RadioButton’.



В состоянии 1 ничего не рисуется. Примеры графиков для состояний 2 и 3:



11. Диалоги

Предметная область

Средства графического пользовательского интерфейса ГИП (GUI – Graphic User Interface) предназначены для создания в Scilab приложений с пользовательским интерфейсом. Определены две возможности:

- TCL/Tk interface. Интерфейс TCL/Tk. Эти средства предназначены для конструирования графического окна с использованием набора управляющих компонентов.
- GUI and Dialogs. ГИП и диалоги. Эти средства предназначены для работы с графическим редактором Scilab.

TCL (от англ. Tool Command Language — «Командный язык инструментов», читается «тикль») — скриптовый язык высокого уровня. TCL часто применяется совместно с графической библиотекой Tk (Tool Kit). Связку TCL/Tk по-русски иногда называют «Так-тикль».

Контрольные вопросы

1. Назначение ГИП.
2. Функция getvalue, получить значения.
3. Функция x_message, показать значения.
4. Функция x_matrix, редактировать значения.

Задание к работе

Задача 1. ГИП с вводом, выводом и редактированием значений.

Создать программу, в которой формируются ГИП для ввода, вывода и редактирования значений. При вводе предусмотреть предложение значений по умолчанию (см. варианты заданий).

Задача 2. ГИП с редактированием значений матрицы.

Создать ГИП с редактированием значений матрицы заданного размера.

Варианты заданий

№	Частота	Амплитуда	Матрица
1.	10	5	5x3
2.	25	8	6x5
3.	35	2	4x5
4.	45	6	3x5
5.	89	4	4x3
6.	34	3	7x5

Пример 1

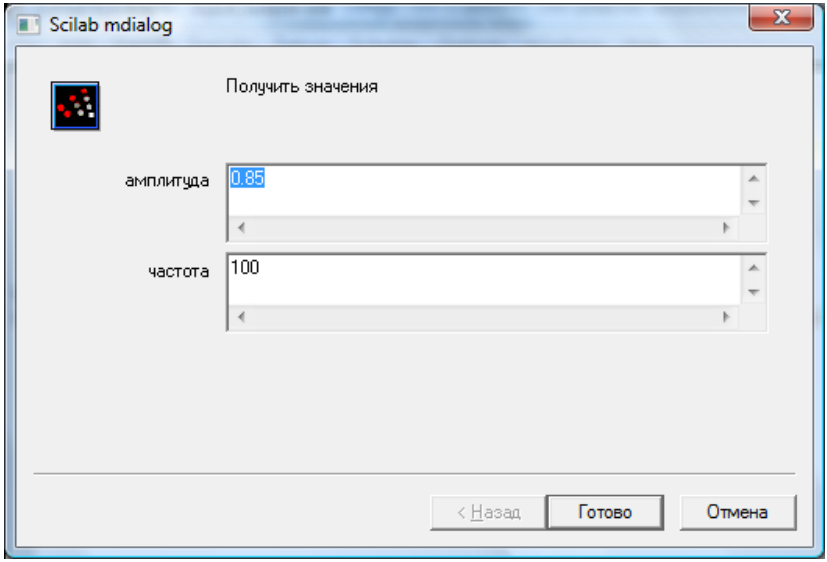
Создать программу, в которой формируются ГИП для ввода, вывода и редактирования значений амплитуды (amp) и частоты (frq) синусоиды. При вводе предусмотреть предложение значений по умолчанию: amp=0.85, frq=100..

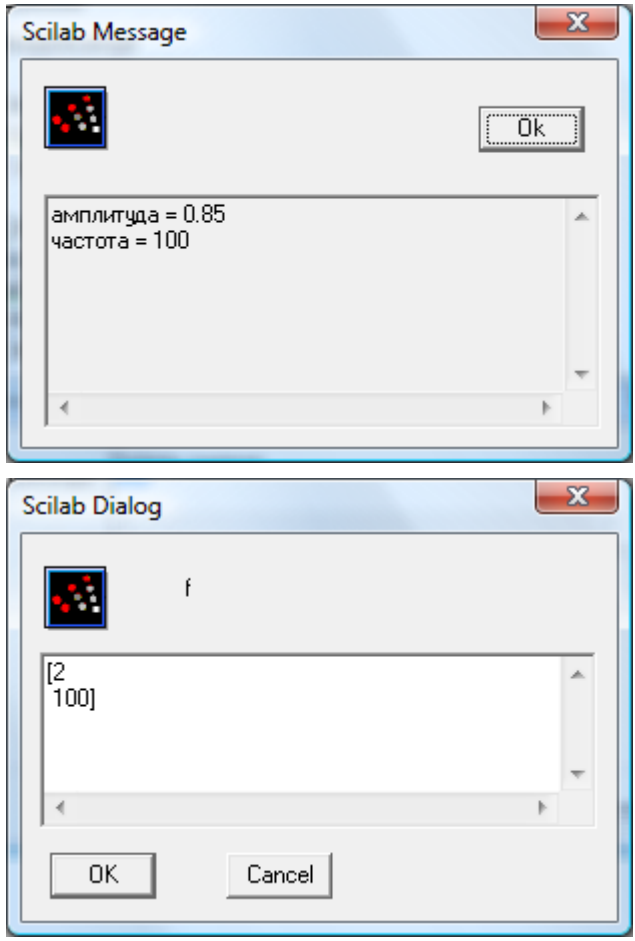
Листинг программы

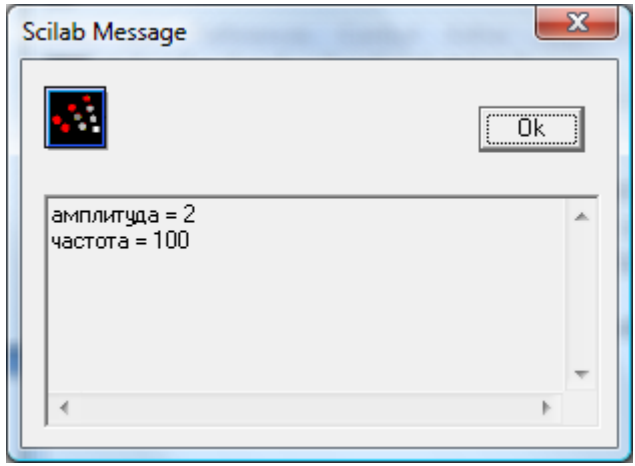
Листинг программы

```
// Диалоги ввода/вывода
// Ввод
labels=['амплитуда';'частота'];
[ok,mag,frq]=getvalue("Получить значения",labels,...
    list('vec',1,'vec',1),['0.85';'100']);
//Вывод
x_message(['амплитуда = '+string(mag),'частота = '+string(frq)]);
//Редактирование
[result]=x_matrix('f',[mag;frq]);
//Вывод после редактирования
x_message(['амплитуда = '+string(result(1)),'частота = '+string(result(2))]);
```

Ниже последовательно создаваемые окна диалога..




Scilab Message
Ok
амплитуда = 0.85
частота = 100
Scilab Dialog
f
[2
100]
OK
Cancel

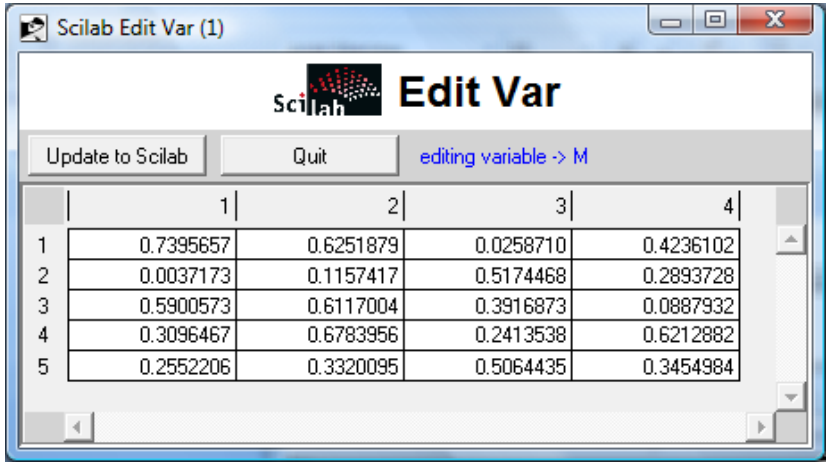


Пример 2

Создать ГИП с редактированием значений матрицы размера 5x4.

Листинг программы

```
// Редактирование матрицы
M=rand(5,4);   // Создать матрицу
//Редактирование
editvar M;
```



	1	2	3	4
1	0.7395657	0.6251879	0.0258710	0.4236102
2	0.0037173	0.1157417	0.5174468	0.2893728
3	0.5900573	0.6117004	0.3916873	0.0887932
4	0.3096467	0.6783956	0.2413538	0.6212882
5	0.2552206	0.3320095	0.5064435	0.3454984