Лекция 2

ПРОГРАММИРОВАНИЕ в SciLab.

Создание программы

1. **Открыть текстовый редактор SciNotes.**

2. Набрать текст программы.

3. **Сохранить текст программы с помощью команды File – Save в виде файла с расширением *sci* , или, например, file.sce.**

4. Вызвать программу, набрав в командной строке ***exec***, например, ***exec("file.sci").***

5. Или воспользоваться командой меню ***File – Exec***. . . или, находясь в окне **SciNotes**, выполнить команду ***Execute – Load into Scilab (F5).***

Создание программы.
1 способ

Например, требуется найти значения функции:

$$f(x)=\begin{cases}\log_3(x-2) & \text{если } x>3\\ \dfrac{e^x-3}{2^x} & \text{если } x\leqslant 3\end{cases}$$

в точках 3.2, -2.7, 4.6

Создание программы. 1 способ

Создаем файл-сценарий *f.sci* в редакторе *SciNotes* и сохраняем его в текущем каталоге:

```
function y=f(x)
if (x > 3) then
  y=log(x-2)/log(3);
else
  y=(exp(x)-3)/2^x;
end
endfunction
```

Создание программы. 1 способ

После этого, функция может быть вызвана в командной строке SciLab с помощью команды:

-->exec('Z:\gr3094.1\~\f.sci')

```
-->function y=f(x)
-->    if x>3 then
-->        y=log(x-2)/log(3.)
-->    else
-->        y=(exp(x)-3)/2^x
-->    end
-->endfunction
```

```
-->[f(3.2) f(-2.7) f(4.6)]
 ans  =

    0.1659562  - 19.057355    0.869744
```

Создание программы. 2 способ

Использовать графический интерфейс SciNotes.

1. Нажать кнопку меню Edit(Редактор).

2. Затем в меню SciNotes нажать мышью кнопку Execute (**Выполнить**) и в выпадающем меню выбрать пункт Save and execute into Scilab (**Сохранить и выполнить в Scilab**).

В результате программа будет выполнена в основном окне редактора SciLab

Создание программы

При использовании функции (файла программы) вычисления внутри нее производятся не интерактивно.

Это означает, что нет способа видеть значение каждого выражения и на этой основе динамически формировать процесс вычислений.

Поэтому нужно предусмотреть команды ввода/вывода

Основные операторы sci-языка

Оператор ввода значений:

x=input(’текст’);

Пример. В программе:

a=input(’a=’);

В командной строке:

a=-->-6

Основные операторы sci-языка

Для вывода в текстовом режиме используют функцию disp:

disp(b)

b - имя переменной или заключенный в кавычки текст.

Пример. В программе:

disp('Вывод y='); disp(y);

В командной строке:

Вывод y=

0.3476307

Оператор вывода **printf** на экран

Функция **printf** служит для вывода на экран значений переменных и текстовых комментариев по формату.

Обращение к **printf** выглядит следующим образом:

printf("текст для вывода %символ1 %символ2 ...%символ*n",переменная1,переменная2,..., переменнаяn)*

В результате обращения к функции **printf** на экран выводится текст, указанный в кавычках, и значения переменных **(переменная1,переменная2,..., переменная*n)***

Символы **(%символ1 %символ2 … %символ*n)*** определяют формат вывода.

Символ всегда ставится после знака **%:**

s – для символов (текст), **f** – для чисел

Запись ***%.nf*** означает, что число должно быть выведено с ***n*** *знаками после запятой.*

Примеры вывода **printf**

Пример 1. Зададим переменную ***h***, в которой будут храниться символы variable и выведем значение переменной ***h*** на экран:

Листинг команд в основном окне:

```
-->h='variable'
h =
variable
-->printf("text: %s",h)
text: variable
```

Примеры вывода **printf**

Пример 2. Выведем на экран значения числовых переменных в сочетании с текстом:

Листинг команд в основном окне:

```
-->d=1.15*%pi
d =
3.6128316
-->printf("значение d = %.5f \n e=%.7f " ,d,%e)
значение d = 3.61283
e=2.7182818
```

```
-->printf("%.5f",1/3)
0.33333
```

Оператор присваивания

Оператор присваивания имеет следующую структуру

a=b

здесь **a** - имя переменной или элемента массива, **b** - значение или выражение.

В результате выполнения оператора присваивания переменной **a** присваивается значение выражения **b**.

Пример.

```
X=5;
X=X+1;
disp(“ Вывод X= ”); disp(X)
```

Условный оператор

Условные операторы могут быть в двух формах: **общая форма** (две ветви) и **расширенная форма** (три и более ветвей).

Логические выражения:

&, and (логическое и),
|, or (логическое или),
~, not (логическое отрицание)

операторы отношения:
< (меньше), **>** (больше),
== (равно), **~=,** (не равно),
<= (меньше или равно),
>= (больше или равно).

Логические операторы

- Обычная форма:

```
if условие
операторы1
else
операторы2
end
```

- Операторы для задания условий: & (and), | (or), ~ (not),~=(<>),==,>,<, >=, <=

- Расширенная форма

```
if условие1
операторы1
elseif условие2
операторы2
...
else
операторы
end
```

Пример программы

Решение квадратного уравнения $ax^2 + bx + c = 0$.

Входными данными этой задачи являются коэффициенты квадратного уравнения a, b, c.

Алгоритм решения:

1. **Ввод** коэффициентов **a, b, c.**
2. Вычисление дискриминанта уравнения **D.**
3. Если **d < 0,** выводится сообщение **“Корней нет”.**
4. Если **d ≥ 0,** то вычисляются 2 корня уравнения
5. **Вывод** значений корней уравнения **x1, x2.**

Листинг программы

```
// Ввод значений коэффициентов уравнения
a=input(’a=’);
b=input(’b=’);
c=input(’c=’);
// Вычисляем дискриминант
d=b*b-4*a*c;
        if d<0 then
        disp(’Корней нет’);
        else
        x1=(-b+sqrt(d))/(2*a);
        x2=(-b-sqrt(d))/(2*a);
        printf(“Корни уравнения: \n %.5f  \n %.5f”,x1,x2);
        end
```

Результат работы программы

```
-->exec("G:/Lecture Scilab EG/2/l1.sci");
-->a=3
-->b=8
-->c=-1
Корни уравнения
-0.34588
0.34588
```

Оператор альтернативного выбора

```
select параметр
case значение1 then операторы1
case значение2 then операторы2
...
else операторы
end
```

Пример оператора выбора

Имеется пронумерованный список деталей: 1) шуруп, 2) гайка, 3) винт, 4) гвоздь, 5) болт. По введенному номеру детали выводить на экран ее название.

```
D=input('Введите число от 1 до 5');
select D
case 1 then disp('Шуруп');
case 2 then disp('Гайка');
case 3 then disp('Винт');
case 4 then disp('Гвоздь');
case 5 then disp('Болт');
else
disp('Такого номера нет');
end
```

```
-->exec('G:\Lecture Scilab EG\2\l2.sci');
        Введите число от 1 до 5 - -> 4
        Гвоздь
```

Операторы цикла в Scilab

Оператор цикла **while** имеет вид:

```
while условие
   операторы тела цикла
end
```

Оператор цикла **for** имеет вид:

```
for x = xn : hx : xk
   операторы тела цикла
end
```