Программирование в системе Scilab

Что такое Scilab

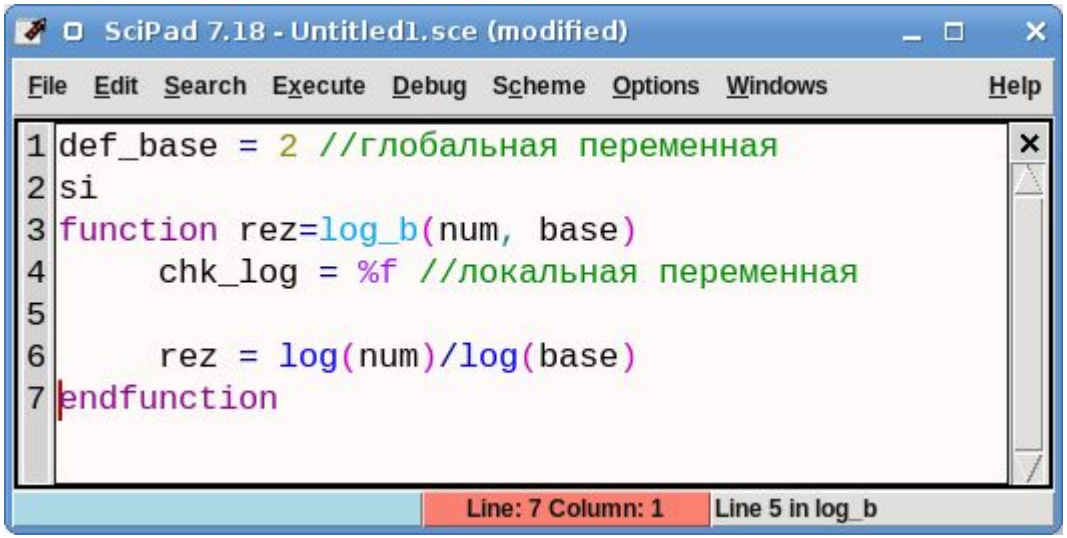
Scilab – это кроссплатформенная система компьютерной алгебры.

Изначально это был коммерческий проект под названием **Blaise**, а затем **Basile**.

С 2003 года продукт получил новое имя Scilab и стал бесплатным. В настоящее время он распространяется по свободной лицензии **CeCILL**.

Редактор SciPad

Для удобства написания скриптов (функций) в Scilab имеется встроенный редактор – Scipad.



Стандартные конструкции встроенного языка

Встроенный язык **Scilab** – это язык структурного программирования не имеющий, в отличие от **Matlab**, средств для работы с объектами.

- Переменные не описываются, а создаются путем присвоения им начального значения, например так:

```
a = 1
b='Hello'
c= %t
```

- Вследствие Unix-корней системы, важен регистр букв в имени переменных, например:

```
-->d=3;D='три';
-->d*3
ans =
   9.0
-->D*3
!--error 144
Операция для заданных операндов не
   определена.
-->D+' – это текст'
ans =
   три – это текст
```

Глобальные и локальные переменные

```
def_base=2 //глобальная
  переменная

function rez=log_b(num, base)
   chk_log=%f //локальная
  переменная

   rez=log(num)/log(base)
endfunction
```

Описание функции

function [выходные параметры]=имя_функции
(входные параметры)

…

тело функции

…

[выходные параметры]=…

endfunction

Линейный процесс вычислений

```
function [outS]=Hello1(Name)
   outS='Привет, '+Name+'!'
endfunction
```

- Вот пример выполнения этой функции:

```
-->Hello1('незнакомец')
ans =
Привет, Незнакомец!
```

- **Укажем массив значений:**

```
Hello1(['Незнакомец';'Инкогнито'])
ans =
!Привет, Незнакомец!  !
!Привет, Инкогнито!   !
```

Эти операции служат для выполнения матричных действий по правилам матричной алгебры. Например:

```
-->a=[1 2 3],b=[3 2 1]
 a =
   1.  2.  3.
 b =
   3.  2.  1.
-->a*b
   !--error 10
Некорректное умножение.
```

Согласно правилам матричной алгебры, важен порядок множителей:

```
-->a*b'
ans =
   10.
-->b' * a
ans =
   3.  6.  9.
   2.  4.  6.
   1.  2.  3.
```

Для выполнения поэлементного умножения двух массивов необходимо поставить перед знаком действия точку:

```
-->a .* b
ans =
   3.  4.  3.
```

Операторы ветвления

Функция для расчета логарифма числа по произвольному основанию:

```
function rez=logB(num,base)
   rez = log(num)/log(base)
endfunction
```

if <Условие> **then** <Выражения>
elseif <условие2> **then** <Выражения2>
...
elseif <условиеN> **then** <ВыражениеN>
else <Выражения>**end**

Новый вид функции с проверкой входных данных на корректность:

```
function [rez]=logB(num, base)
  //Проверка размера массивов
  if or([length(num)>1, length(base)>1]) then
    error('Ошибка: массив не может быть входным
  параметром');
  else
    if and([num>0, base>0, base<>1]) then
       rez = log(num)/log(base)
    else
      error('Ошибка: неверные входные данные');
    end
  end
endfunction
```

Способ использования прост:

warning('on') //включение режима вывода предупреждений

warning('сообщение') //вывод сообщения

warning('off') //выключение режима вывода предупреждений

Общий вид оператора множественного выбора

```
select <переменная>
  case <значение 1> then <действия 1>
  case <значение 2> then <действия 2>
  case <значение 3> then <действия 3>
…
  case <значение N> then <действия N>
  else
     <действия по умолчанию>
end
```

Рассмотрим функцию, получающую количество информации в байтах и выдающее название наибольшей единицы измерения.

```
function rez=edIzm(N)
sN = string(N)
select length(sN)
case 1 then rez='Байт'
case 2 then rez='Байт'
case 3 then rez='Байт'
case 4 then rez='Килобайт'
case 5 then rez='Килобайт'
case 6 then rez='Килобайт'
case 7 then rez='Мегабайт'
case 8 then rez='Мегабайт'
case 9 then rez='Мегабайт'
else
warning('on')
warning('Введенное больше чем 999 Мегабайт')
warning('off')
rez='Много'
end //select
endfunction
```

Циклы

Счетный (for) и условный (while).

Общий вид оператора счетного цикла следующий:

```
for <счетчик>=<Выражение>
   <тело цикла>
End
```

Рассмотрим описанную нами функцию edIzm.

```
-->edIzm([1,2,4])
ans =
  Байт
-->edIzm([1,23,4])
WARNING: Введенное больше чем 999
  Мегабайт
ans =
  Много
```

Счетный оператор цикла:

```
function [rez]=edIzm(N)
i=0
for iN=N
i=i+1
sN=string(iN)
select length(sN)
case 1 then rez(i)='Байт'
case 2 then rez(i)='Байт'
case 3 then rez(i)='Байт'
case 4 then rez(i)='Килобайт'
case 5 then rez(i)='Килобайт'
case 6 then rez(i)='Килобайт'
case 7 then rez(i)='Мегабайт'
case 8 then rez(i)='Мегабайт'
case 9 then rez(i)='Мегабайт'
else
warning('on')
warning(sN+' больше чем 999 Мегабайт')
warning('off')
rez(i) ='Много'
end //select
end //for
endfunction
```

Возможен и такой вариант начала функции:

```
function [rez]=edIzm(N)
   NSize=length(N)
   for i=1:NSize
     sN=string(N(i))
     select length(sN)
...
```

Цикл while.

Общий вид этого оператора:

while <выражение>
<тело цикла>
end

- **Вместо строк**

```
sN=string(N(i))
select length(sN)
```

- **можно вставить следующее:**

```
iN=0
NTemp=N(i)
while NTemp>0 do
   iN=iN+1
   NTemp=int(NTemp/10)
end
select iN
```

Спасибо за внимание