Scilab

Programming
&
Functions

By M. D. Bhopatkar
Bhaskaracharya Pratishthan

Scilab Workshop at Bhaskaracharya Pratishthan
6th July, 2009

Programming

- Interpreter with it's own syntax
  - Execution of commands
    - Line by line
    - By block
- Set of programming tools
  - Loops
  - Conditionals

for loop

```
for variable = expression
    .
    .
    .
end
```

Examples

```
→ x=1; for k=1:4, x=x*k, end
ans  x=24
→ x=1; for k=[-1 3 0], x=x+k, end
ans  x=3
→ l=list(1, [1,2;3,4], 'str')
→ for k=l, disp(k), end
ans  1
        ! 1.  2. !
        ! 3.  4. !
        str
```

while loop

while expr

.

.

.

end

example

```
→ i=1; x=0
→ while   i<=4
→      x=x+i;
→ end
→ disp (x)
```

ans x=10

Loop breaks

- Loop can be ended by command break
- In nested loops, break exits from innermost loop

Conditionals

- if then else

  → x=1;

  → if x>0 then, y= -x, else, y=x, end

  ans y= -1

- select case

  → x= -1;

  → select x, case 1, y=x+5, case -1, y=sqrt(x), end

  ans y=1

Functions

- function [y1, ..., yn]=f00(x1, ..., xm)

  .

  .

  .

  endfunction

  <u>Note</u>: function has local environment that communicates with the outside thru input and output arguments

Features of Functions

- Functions can be defined online or offline
- Arguments can be any Scilab objects
- More than one output arguments
- Input or output arguments can be functions
- Functions can be nested

Online definition : general format

```
-->function [y]=f00(x)
→     y=x*abs(x)/(1+x^2)
-->endfunction

-->x=[1.  2.  3.];
-->f00(x)
 !--error 10
Inconsistent multiplication.
 -->f00(.5)
 ans = 0.2
```

Online definition : general format

```
-->function [y]=f01(x)
→       y=x.*abs(x)./(1+x.^2)
-->endfunction


-->f01(x)
 ans =
    0.5    0.8    0.9
 -->[x ; f01(x)]
 ans =
     1.     2.     3.
    0.5    0.8    0.9
```

Online definition : simple format

```
-->deff('y=f01(x)','y=x^3-2*x-5')
-->deff('y=f02(x)','y=3*x^2-2')
-->deff('y=f03(x)','y=x-(f01(x)/f02(x))')

-->f03(2)
 ans = 2.1

 -->f03(ans)
 ans = 2.0945681
```

Function written in file

- Create functions in any editor like Scipad
- Such function should be loaded in the Scilab environment
- Commands are getf('filename') or exec('filename', -1)
- A file may contain several functions

Example : Vector argument

```
function y=fv1(x)
  t1=x^2+2
  t2=2*x.*sin(x)
  y=t1./t2
endfunction
```

Example : Recursive function

```
function [y]=fact(x)
    if x==1 then  y=1
    else
        y=x*fact(x-1)
    end
endfunction
```

Example : Recursive function

```
function y=g(n,m)
    if m==0 then y=n
    else
    r=modulo(n,m)
    y=g(m,r)
    end
endfunction
```

Example : Multiple defined function

```
function [y]= mdf1(x)
   if x>=1 then
    y=x^2
  elseif x>=-1&x<1 then
    y=sin(2*x)
  else
    y=x/(x^3+2)
end
endfunction
```

Execution

```
-->getf('d:\mdb\myscilab\mdef1.sci')
 -->for j=1:3, y(j)=mdf1(x(j));end
 -->y
 y  =
    4.
   0.0406504
   0.1986693
  -->[ x ;y' ]
 ans  =
    2.        - 5.                0.1
    4.          0.0406504    0.1986693
```

Example : Multiple-defined function (Matrix argument)

```
function [y]= mdf2(x)
  [m,n]=size(x)
  y=zeros(m,n)
  for i=1:m
    for j=1:n
      if x(i,j)>=1 then
        y(i,j)=x(i,j)^2
```

continued...

```
    elseif x(i,j)>=-1&x(i,j)<1 then
       y(i,j)=sin(2*x(i,j))
    else
      y(i,j)=x(i,j)/(x(i,j)^3+2)
    end
  end
 end
endfunction
```

Execution

```
-->getf('d:\mdb\myscilab\mdef2.sci')
-->x=[2 -5 0.1];
-->mdf2(x)

 ans  =

    4.    0.0406504    0.1986693
```

Example : Multiple-defined function (vector argument)

```
function [y]= mdf3(x)
  n=size(x,2), y=zeros(1,n)
     for j=1:n
       if x(j)>=1 then y(j)=x(j)^2,
      elseif x(j)>=-1&x(j)<1 then
         y(j)=sin(2*x(j))
      else y(j)=x(j)/(x(j)^3+2),
      end
  end
endfunction
```

Execution

```
-->getf('d:\mdb\myscilab\mdef3.sci')
-->mdf3(x)
 ans  =
    4.    0.0406504    0.1986693
 -->[x ;y']
 ans  =
    2.       - 5.              0.1
    4.         0.0406504    0.1986693
 -->[x;mdf3(x)]
 ans  =
    2.       - 5.              0.1
    4.         0.0406504    0.1986693
```

More on functions

- If last argument of a function definition is named varargin, then the function can be called with more than N arguments.
- In a function input argument can be a function

  e. g. y=regfl(a, b, f, n)
- Similarly in a function output argument can be a function

More on functions

- Functions can have global variables
- Functions can be invoked with less input or output parameters
- Introducing a pause command permits debugging of Scilab function
- Execution of function is resumed by 'return' or 'resume' command

Thank you