DHA Suffa University
CS 206 – Operating Systems – Lab
Fall 2017

Lab 04 – Loops and Conditional execution in Shell Scripting

Twitter: @oslabatdsu

Objective(s):

- Writing Loops in Shell Script
- For Loop
- While Loop
- Case Statements
- Conditional Execution
- Patterns using Nested For-Loop

Loops in Shell Scripts

Loop defined as:
"*Computer can repeat particular instruction again and again, until particular condition satisfies. A group of instruction that is executed repeatedly is called a loop.*"

Bash supports:

- for loop
- while loop

Note that in each and every loop,

- First, the variable used in loop condition must be initialized, then execution of the loop begins.

- A test (condition) is made at the beginning of each iteration.

- The body of loop ends with a statement that modifies the value of the test (condition) variable.

for Loop

Syntax:

```
for { variable name } in { list }
 do
      execute one for each item in the list until the list is
      not finished (And repeat all statement between do and done)
 done
```

Even you can use following syntax:

Syntax:

```
for (( expr1; expr2; expr3 ))
   do
         .....
              ...
         repeat all statements between do and
         done until expr2 is TRUE
   Done
```

- In above syntax BEFORE the first iteration, ***expr1*** is evaluated. This is usually used to initialize variables for the loop.

- All the statements between do and done is executed repeatedly UNTIL the value of ***expr2*** is TRUE.

- AFTER each iteration of the loop, ***expr3*** is evaluated. This is usually use to increment a loop counter.

```
for ((  i = 0 ;  i <= 5;  i++  ))
do
  echo "Welcome $i times"
done
```

Nesting of for Loop

As you see the if statement can nested, similarly loop statement can be nested. You can nest the for loop. To understand the nesting of for loop see the following shell script.

```
for (( i = 1; i <= 5; i++ ))      ### Outer for loop ###
do

    for (( j = 1 ; j <= 5; j++ )) ### Inner for loop ###
    do
          echo -n "$i "
    done

  echo "" #### print the new line ###

done
```

Following script is quite intresting, it prints the chess board on screen.

```
for (( i = 1; i <= 9; i++ )) ### Outer for loop ###
do
   for (( j = 1 ; j <= 9; j++ )) ### Inner for loop ###
   do
        tot=`expr $i + $j`
        tmp=`expr $tot % 2`
        if [ $tmp -eq 0 ]; then
            echo -e -n "\033[47m "
        else
            echo -e -n "\033[40m "
        fi
  done
 echo -e -n "\033[40m" #### set back background colour to
black
 echo "" #### print the new line ###
done
```

Above shell script cab be explained as follows:

Command(s)/Statements	Explanation
for ((i = 1; i <= 9; i++)) do	Begin the outer loop which runs 9 times., and the outer loop terminets when the value of **i** exceeds 9
for ((j = 1 ; j <= 9; j++)) do	Begins the inner loop, for each value of **i** the inner loop is cycled through 9 times, with the varible **j** taking values from 1 to 9. The inner for loop terminates when the value of **j** exceeds 9.
tot=`expr $i + $j` tmp=`expr $tot % 2`	See for even and odd number positions using these statements.

if [$tmp -eq 0]; then
echo -e -n "\033[47m "
else
echo -e -n "\033[40m "
fi
	If even number posiotion print the white colour block (using **echo -e -n "\033[47m "** statement); otherwise for odd postion print the black colour box (using **echo -e -n "\033[40m "** statement). This statements are responsible to print entier chess board on screen with alternet colours.

done	End of inner loop
echo -e -n "\033[40m"	Make sure its black background as we always have on our terminals.
echo ""	Print the blank line
done	End of outer loop and shell scripts get terminted by printing the chess board.

While loop

Syntax:

```
while [ condition ]
       do
             command1
             command2
             command3
             ..
             ....
        done
```

Lab Tasks:

1. **Write a Shell Script using vi to check whether a number (input by user) is prime or not.**
2. **Write a Shell Script using vi to print the sum of the digits of a given number. (number can be of any number of digits)**

Case Statements

The case statement is good alternative to Multilevel if-then-else-fi statement. It enable you to match several values against one variable. Its easier to read and write.

Syntax:

```
case  $variable-name  in
        pattern1)   command
                        ...
                        command;;
        pattern2)   command
                        ...
                        command;;
        patternN)   command
                        ...
                        command;;
        *)             command
                        ...
                        command;;
esac
```

The *$variable-name* is compared against the patterns until a match is found. The shell then executes all the statements up to the two semicolons that are next to each other. The default is *) and its executed if no match is found.

Conditional execution i.e. && and ||

The control operators are && (read as AND) and || (read as OR). The syntax for AND list is as follows

Syntax:
command1 && command2
command2 is executed if, and only if, command1 returns an exit status of zero.

The syntax for OR list as follows
Syntax:
command1 || command2
command2 is executed if and only if command1 returns a non-zero exit status.

You can use both as follows
Syntax:
command1 && comamnd2 if exist status is zero || command3 if exit status is non-zero
if command1 is executed successfully then shell will run command2 and if command1 is not successful then command3 is executed.

Example:
$ rm myf && echo "File is removed successfully" || echo "File is not removed"

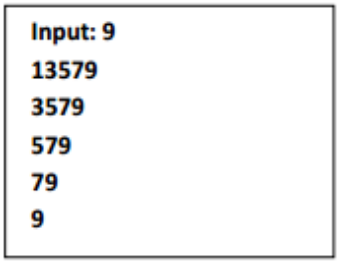
If file (myf) is removed successful (exist status is zero) then "*echo File is removed successfully*" statement is executed, otherwise "*echo File is not removed*" statement is executed (since exist status is non-zero)

Lab Assignment 04

1. **Write a shell script to calculate the logarithm of a given number(input by user) to a given base(input by user).**

2. **Write a shell script using function(s) to calculate Factorial of a given number.**

3. **Write a shell script to print the following pattern against a given number.**



Submission Instructions:

1. **Number your .sh files as question number e.g. Q1.sh, Q2.sh, etc. (Q is in upper case)**
2. **For every .sh file, there should be a word file containing at least ten snapshots on different stages of editing the code in vi.**
3. **Create a new folder named cs152abc where abc is your 3 digit roll #. e.g. cs152111.**
4. **Copy all the .sh files and word files into this folder.**
5. **Right-click on the folder you created and create a zip file by selecting the option**
   - **"Send to" and selecting "Compressed (zipped) folder" [for windows].**
   - **"Create Archive" and change option to ".zip" instead of ".tar.gz" and click on "Create". [for linux]**

   Now make sure a zip file named cs152abc.zip is created e.g. *cs152111.zip*.
6. **(A) Compose a new email, attach this zip file and send email to *oslabatdsu@gmail.com*.**
   - **The subject of the email must be:**

     4A-Lab03-cs152abc **where 4A is the section you're enrolled in and abc is your 3 digit unique roll#.**

     4A-Lab03-cs152001

   (B) The body of the email must contain:

   Full Name
   DSU Roll #
   Section
   Lab Number #
7. **To double check whether your assignment was submitted or not make sure the Sent Items folder of your email account contains the email you just sent.**
8. **Due Date for assignment submission is Sept 24, 2017.**