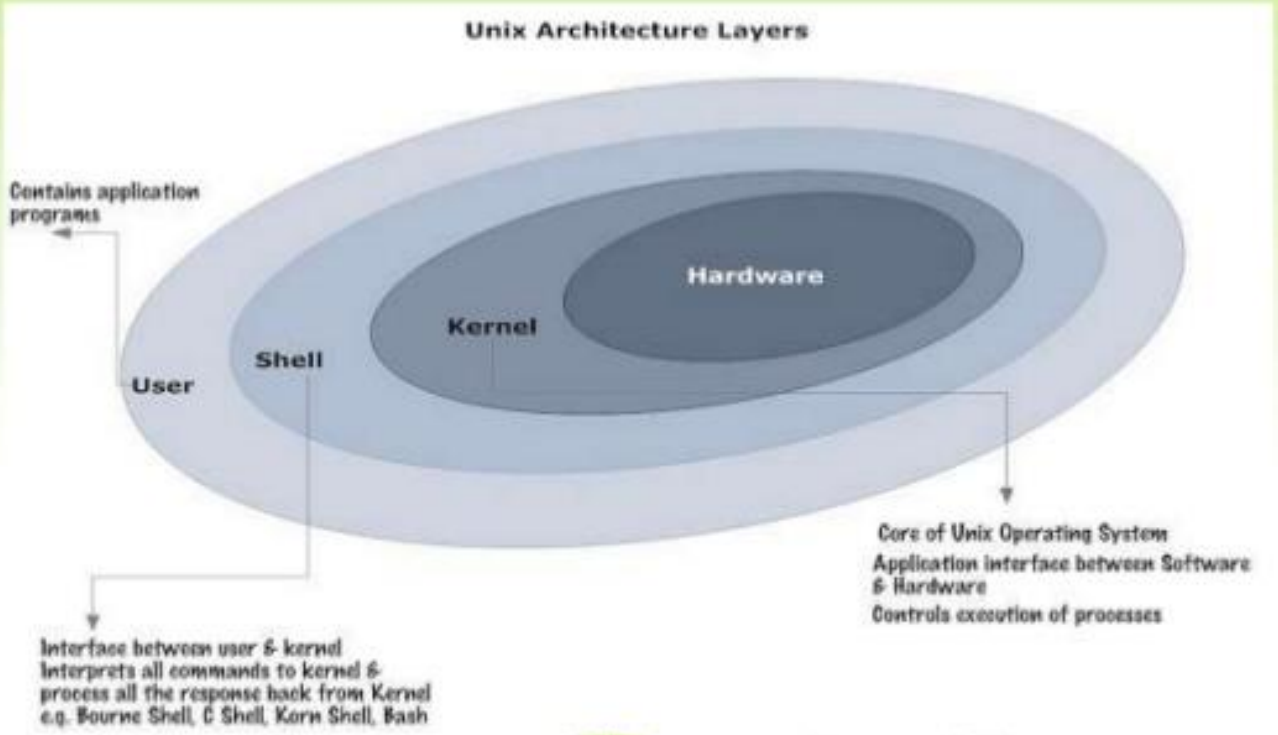
Shell Programming

Structure of Unix



What is Shell ?

- The Shell is an interface between User and Kernel.
- Shell accepts the commands from the user and converts them in language that Kernel understands
- Two major Rolls of Shell
  - Interpreter: Reads the Commands, works with the Kernel to execute them
  - Programming Capability: Shell Script is a file that contains shell commands to perform a specific task…Shell Program

What is a shell program?

- Simply put, a shell program (sometimes called a shell script) is a text file that contains standard UNIX and shell commands.
- Each line in a shell program contains a single UNIX command exactly as if you had typed them in yourself.
- Shell programs are interpreted and not compiled programs.

Types of Shell

Bourne shell (sh)

C shell (csh)

TC shell (tcsh)

Korn shell (ksh)

Bourne Again SHell (bash)

Bash shell programming

- Input
  - prompting user
  - command line arguments
- Decision:
  - if-then-else
  - case
- Repetition
  - do-while, repeat-until
  - for
  - select
- Functions

Simple Shell Programs

```
# This is a comment
echo "Hello World !!”
```

Save it as :

Hello.sh

Run on the prompt as :

sh Hello.sh

Taking input from User :

- shell allows to prompt for user input

Syntax:

```
read varname [more vars]
```

- words entered by user are assigned to **varname** and "**more vars**"
- last variable gets rest of input line

User input example

E.g. 1:

```
echo "First name:”
read  first
echo "Last name: ”
read last
echo “Your name is : $first  $last”   //Retrieving Value of
   a variable
```

E.g. 2:

```
echo “What is your name?”
read MY_NAME
echo "Hello $MY_NAME - hope you're well."
```

#print date - today.sh

```
echo "Today is:"
date
```

Save it as :

today.sh

Run:

sh today.sh

UNIX Commands can be run through shell files

E.g.1 :

```
echo "Hello $USER"
echo "This machine is `uname -n`"
echo "The calendar for this month is:"
cal
echo "You are running these processes:"
ps
```

Output

```
% chmod u+x hello
% ./hello
Hello ege!
This machine is turing
The calendar for this month is
    February 2008
 S  M Tu  W Th  F  S
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
You are running these processes:
   PID TTY      TIME CMD
 24861 pts/18   0:00 hello.csh
 24430 pts/18   0:00 csh
```

E.g. 2:

```
echo “Files listed by default command :”
echo
echo
ls
echo “Files listed in Seven Attribute Format :”
echo
Echo
ls -l
```

bash control structures :

- if-then-else
- case
- loops
  - for
  - while
  - until
  - select

if statement :

Syntax :

```
if command
then
  statements
fi
```

- Executes the statements only if **condition** is true

The if-then-else statement

```
if [ condition ];
then
    statements-1
else
    statements-2
fi
```

- executes statements-1 if condition is true
- executes statements-2 if condition is false

The if...else statement

```
if [ condition ]; then
    statements
elif [ condition ]; then
    statement
else
    statements
fi
```

- The word **`elif`** stands for “else if”
- It is part of the if statement and cannot be used by itself

1. if *command1*
then
command-list
fi

2. if *command1*
then
command-list1
elif command1
command-list2
fi

3. if *command1*
then
command-list1
else
command-list2
fi

#if stmt - if_ex.sh

```
echo "Enter the search string:"
read string
echo "Enter the file name:"
read filename
grep -n "$string" $filename
# $? gives exit status of last command
if [ $? -eq 0 ]
then
echo " Yes,the word present in the file"
else
echo "No,word not present"
fi
```

o/p :

[oss@pc021698 ~]$ sh if_ex.sh

Enter the search string:

date

Enter the file name:

today.sh

2:date

Yes,the word present in the file

o/p:

[*oss@pc021698 ~]$ sh if_ex.sh*

Enter the search string:

year

Enter the file name:

today.sh

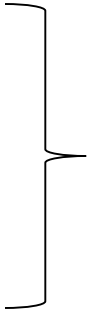
No,word not present

Relational Operators

Meaning	Numeric	String
Greater than	-gt	
Greater than or equal	-ge	
Less than	-lt	
Less than or equal	-le	
Equal	-eg	= or ==
Not equal	-ne	!=
str1 is less than str2		str1 < str2
str1 is greater str2		str1 > str2
String length is greater than zero		-n str
String length is zero		-z str

Compound logical expressions :

!	not
&&	and
||	or



and, or
must be enclosed within

[[ ]]

If...else

```
#Program to decide if a number is a two-digit number


n=68
if [[ ( $n -gt 9  &&  $n -lt 100 ) ]];
then
        echo "It is a two digit number”
else
        echo "It is not a two digit number”
fi
```

Example: Using the || Operator

```
#!/bin/bash

read -p "Enter calls handled:" CHandle
read -p "Enter calls closed: " CClose
if [[ "$CHandle" -gt 150 || "$CClose" -gt 50 ]]
   then
   echo "You are entitled to a bonus"
else
   echo "You get a bonus if the calls"
   echo "handled exceeds 150 or"
   echo "calls closed exceeds 50"
fi
```

Example: if..elif... Statement

```
#!/bin/bash

read -p "Enter Income Amount: " Income
read -p "Enter Expenses Amount: " Expense

let Net=$Income-$Expense

if [ "$Net" -eq "0" ]; then
   echo "Income and Expenses are equal - breakeven."
elif [ "$Net" -gt "0" ]; then
   echo "Profit of: " $Net
else
   echo "Loss of: " $Net
fi
```

While loop :

E.g. 1:

```
valid=true
count=1
while [ $valid ]
do
        echo $count
if [ $count -eq 5 ];
then
        break
Fi
((count++))
done
```

- **Output :**

0

1

2

3

4

5

while loop – syntax

```
while [ condition ]

do

    code block;

done
```

E.g. 2 :

```
#while_ex.sh

verify="n"

while [ "$verify" != y ]

do

    echo "Enter option: "

    read option

    echo "You entered $option.  Is this correct? (y/n)"

    read verify

done
```

E.g. 3 :

```
a=0
while [ "$a" -lt 10 ]  # this is loop1
do
   b="$a"
   while [ "$b" -ge 0 ]  # this is loop2
   do
          echo -n "$b "
          b= $b - 1
   done
   echo
    a= $a + 1
done
```

Output :

```
0
1 0
2 1 0
3 2 1 0
4 3 2 1 0
5 4 3 2 1 0
6 5 4 3 2 1 0
7 6 5 4 3 2 1 0
8 7 6 5 4 3 2 1 0
9 8 7 6 5 4 3 2 1 0
```

While Loop

E.g. 4:

```
INPUT_STRING=hello

while [ "$INPUT_STRING" != "bye" ]

                do
                echo "Please type something in (bye to quit)"
                read INPUT_STRING
                echo "You typed: $INPUT_STRING"
                done
```

For Loop :

```
for (( counter=10; counter>0; counter-- ))
do
        echo -n "$counter”
done
```

Arrays :

```
NUMS="1 2 3 4 5 6 7“

for NUM in $NUMS
do
    Q= $NUM % 2
    if [ $Q -eq 0 ] then
            echo “Even number!!"
            continue
    fi
    echo “Odd number"
done
```

Output :

```
Odd number
Even number!!
Odd number
Even number!!
Odd number
Even number!!
Odd number
```

```
#!/bin/sh

NUMS="1 2 3 4 5 6 7"

for NUM in $NUMS
do
  Q=`expr $NUM % 2`
  if [ $Q -eq 0 ]
  then
    echo "$NUM is an Even number!!"
    continue
  fi
  echo "$NUM is an Odd number"
done
```

```
1  is an Odd number
2  is an  Even number!!
3  is an  Odd number
4  is an  Even number!!
5  is an  Odd number
6  is an  Even number!!
7  is an  Odd number
```

Functions

- E.g. 1:

```
function disp
{
        echo "Hello World !!"
}
disp
```

O/p: Hello World!!

- **E.g. 2:**

```
function disp
{
        echo "Hello World !!"
}
i=1
while((i<=3))
do
   disp
   i=i+1
done
```

O/p: Hello World!!
Hello World!!
Hello World!!

Lab Assignments:

Write a shell programs to

1. Add 2 numbers
2. Check if a number entered is even and odd
3. Find sum of n numbers
4. Determine if a person is eligible to vote or not
5. Display all filenames beginning with character 'a' and displays its contents
6. Find Factorial of a number
7. Check validity of a username and password with a function defined in the code

1. Program to add 2 nos

```
echo enter a

read a

echo enter b

read b

s=$((a + b))

echo sum is $s
```

2. Program to check if a number is even and odd

```
echo enter a number
 read n

rem=$(( $n % 2 ))

if [ $rem -eq 0 ] then
        echo $n is even
 else
        echo $n is odd
fi
```

3. Program to find sum of n numbers

```
a=1
echo " enter n"
read n
s=0
while [ $a -le $n ]
do
        s=$((s + a))
        a=$((a + 1))
done
echo " SUM is " $s
```

4. Program to determine if a person is eligible to vote or not

```
read -p "Enter your age: " Years
if [ " $Years " -lt 20 ];
Then
        echo " You can not Vote now "
else
                echo " You can Vote now "
fi
```

5. displays all filenames beginning with a and displays its contents

```
for k in a*
do
echo "file name is $k"
cat $k
done
```

6 factorial of a number

```
echo " enter the number "
read n
f=1
for((i=1; i<=n; i++))
do
   f=$((f * i))
done
echo "Factorial of " $n " is " $f
```

7. Program to check validity of a user with a function defined in the code

```
function entry()
{
   echo " Enter Username"
    read username
   echo "Enter password“
    read password
}
entry
if [[ ( $username == "admin" && $password == "secret" ) ]];
then    echo "valid user"
else    echo "invalid user“
fi
```