Shell Programming

CS342,Tutorial 2

Date : 19th Jan, 2021

OUTLINE

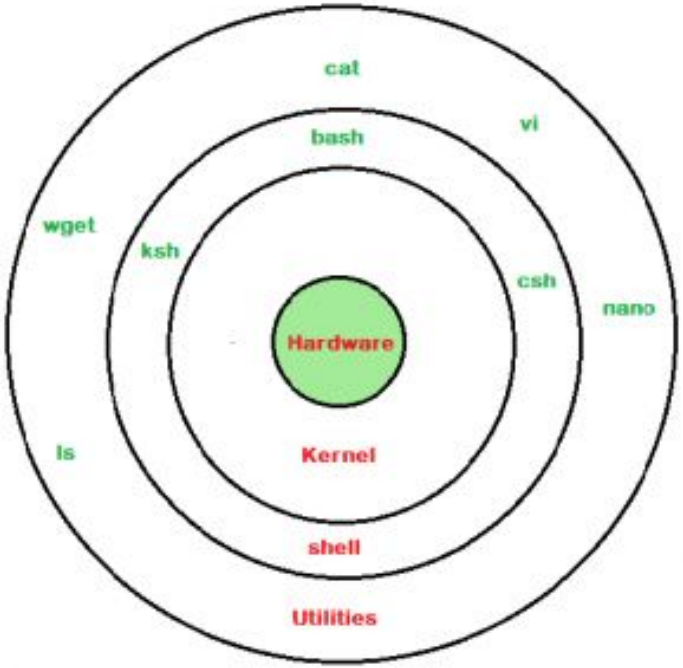
- Shell Script
- Data Structure in Shell Script
- Conditional Statement and Loops
- Functions
- Today’s Assignment
- References

No programming language is perfect. There is not even a single best language; there are only languages well suited or perhaps poorly suited for particular purposes.

— *Herbert Mayer*

How it works ?

- An interface to user to use operating system services
- Shell accept human readable commands from user and convert them into something which kernel can understand
- Bash(Bourne-Again SHell) is the shell, or command language interpreter, for the Linux operating system.



How it is different from other Languages ?

- It has a more **permissive** programming style.
- SSs are best for **automating** things that you would normally do manually - moving files around, searching in program or filtering log files for things that happened in a given time range.
- Shells have specialized features for **working with files** and getting data from one program into another

Array in Shell Script

In Shell script Array is a variable which contains multiple values may be of same type or different type.

```
declare -a A
A=([0]=10 [1]=20 [2] = 30 [3] = CS342)
echo ${A[*]}
echo ${A[1]}
echo ${A[@]:1:4}
echo ${#A[*]}
echo ${#A[3]}
unset A[1]
```

1. Array Declaration
2. Initialization
3. Content
4. Array Element Access : Second Array Element
5. Elements in range
6. Length of Array
7. Length of Particular Array Element
8. Deleting a particular Array Element

IF-THEN-ELSE

```
if command
then
        command
         ……
         Command
else
         Command
fi
```

```
if [ "$NAME"="CS342" ]
then
        echo  “ Welcome to " $NAME Lab “
else
        echo “ ! 404 : Not Found !  “
fi
```

Loops

For Loop

```
for i in 1 2 3 4 5
do
  echo "Looping ... number $i"
done
```

While Loop

```
INPUT_STRING=start

while [ "$INPUT_STRING" != "bye" ]
do
  echo "Please type something in
(bye to quit)"

  read INPUT STRING
  echo "You typed: $INPUT_STRING"
done
```

Function

```
#!/bin/sh

# A simple script with a function...

Database()

{

  USER=$1

  PASSWORD=$2

  shift; shift;

  ROLE=$@

  echo "Adding user $USER ..."

  echo useradd -c "$COMMENTS" $USER

  echo passwd $USER $PASSWORD

  echo "Added user $USER with role ($ROLE) having
password $PASSWORD"

}

# Main body of script starts here

echo "Registration details  :  "

Database bob rsfkass 1997 M Presenter

Database alice Tttfg 2001 M Listener

echo "Task Completed "
```

Some useful Commands

- chmod +x : filename.sh : To make filename.sh executable
- $0 : The filename of the current script
- $# : The number of arguments supplied to a script.
- $$: The process number of the current shell.
- -eq : Checks if the value of two operands are equal or not
- -ne : Checks if the value of two operands are equal or not
- -gt : Checks if the value of left operand is greater than the value of right operand;
- -lt : Checks if the value of left operand is less than the value of right operand
- -ge : Checks if the value of left operand is greater than or equal to the value of right operand;
- -le : Checks if the value of left operand is less than or equal to the value of right operand
- ! : logical negation
- -o : logical OR
- -a : logical AND

Assignment 2

1. Write a shell program that will create an array of size N having values n_1, n_2,n_3……. n_N.Print a message “Search found along with its index of searched item S”. All the values(N, n_i, S) should be taken from the Command Line Argument(CLA). Note: If the searched item does not contain in the array then output an error message.

2. Write a recursive shell program that should output the product of factorial of a number with the sum of all the prime no. less than equal to that number. Take N from CLA.
   E.g : N = n , Output = fact(n) * PrimeNoLessThan(n)
   = n * fact(n-1) * (if(n∈ Prime number) + PrimeNoLessThan(n-1))

3. Implement an XOR function using a shell script that will take two numbers as CLA and output its XOR. Note: You need to convert the inputs in binary form. Also, implement the same using python/C and observe time taken by both the programs for the same input(> 100).

Continued. . .

4. Write a shell script to validate password strength. Here are a few assumptions for the password string.

- Length – a minimum of 7 characters.
- Contain both alphabet and number.
- Contain one special char (/, (,),<, >, ?)

If the password doesn’t comply with any of the above conditions, then the program should report it as a <Invalid Password>.

References

- The Shell and Shell Scripting : https://www.cs.cmu.edu/~guna/15-123S11/Lectures/Lecture22.pdf
- Shell Programming : http://www.docs.is.ed.ac.uk/skills/documents/2630/2630.pdf

Thank you