LAB 2 : SHELL PROGRAMMING

Nirajan Bekoju
PUL076BCT039
076bct039.nirajan@pcampus.edu.np

Nishant Luitel
PUL076BCT041
076bct041.nishant@pcampus.edu.np

Nabin Da Shrestha
PUL076BCT037
076bct037.nabin@pcampus.edu.np

Prakash Chaulagain
PUL076BCT045
076bct045.prakash@pcampus.edu.np

I. Introduction

SHELL programming is the process of writing computer programs that are executed in a shell environment, typically a Unix or Linux shell. A shell is a command-line interface that allows users to interact with the operating system by executing commands and scripts.

Shell programming can be used to automate repetitive tasks, perform system administration tasks, and create complex scripts that can be used to manipulate data and perform various tasks. Shell scripts are often used to automate tasks such as backups, file and directory management, system monitoring, and software installation.

The most commonly used shells for programming are the Bash shell (which is the default shell on most Linux distributions) and the Zsh shell. Shell programming requires knowledge of the shell's syntax and built-in commands, as well as familiarity with other programming concepts such as variables, control structures, and functions.

In addition to the built-in commands provided by the shell, shell programming also supports the use of external commands and utilities that can be executed from within the shell script. Shell scripts can also accept command-line arguments and input from users, making them highly customizable and flexible.

Overall, shell programming is a powerful tool for automating tasks and managing systems in a Unix or Linux environment, and is an essential skill for system administrators, developers, and anyone who works with command-line interfaces.

II. Concatenation of two strings

Aim : To write a shell program to concatenate two strings.

Algorithm :

Step1: Enter into the vi editor and go to the insert mode for entering the code
Step2: Read the first string.
Step3: Read the second string
Step4: Concatenate the two strings
Step5: Enter into the escape mode for the execution of the result and verify the output

Program :

```
#!/bin/sh

echo "enter the first string"
read str1

echo "enter the second string"
read str2

echo "Concatenated string : $str1 $str2"
```

Output :

```
enter the first string
luffy
enter the second string
zoro
Concatenated string : luffy zoro
```

III. Comparision of two strings

Aim : To write a shell program to compare the two strings.

Algorithm :

Step1: Enter into the vi editor and go to the insert mode for entering the code
Step2: Read the first string.
Step3: Read the second string
Step4: Compare the two strings using the if loop
Step5: If the condition satisfies then print that two strings are equal else print two strings are not equal.
Step6: Enter into the escape mode for the execution of the result and verify the output

Program :

```
#!/bin/sh
echo "enter first string"
read str1
echo "enter second string"
read str2

if [ $str1 = $str2 ]
then
  echo "strings are equal."
else
  echo "strings are not equal."
fi
```

Output:

```
enter first string
lion
enter second string
lion
strings are equal.
```

IV. Maximum of three numbers

Aim : To write a shell program to find the greatest of three numbers.

Algorithm :

Step1: Declare the three variables.
Step2: Check if A is greater than B and C.
Step3: If so print A is greater.
Step4: Else check if B is greater than C.
Step5: If so print B is greater.
Step6: Else print C is greater.

Program :

```
#!/bin/sh
echo "Enter first number"
read a
echo "Enter second number"
read b
echo "Enter third number"
read c

if [ $a -gt $b -a $a -gt $c ]
then
  echo "$a is greater than $b and $c"
elif [ $b -gt $a -a $b -gt $c ]
then
  echo "$b is greater than $a and $c"
elif [ $c -gt $a -a $c -gt $b ]
then
```

```
    echo "$c is greater than $a and $b"
  else
    echo "All are equal"
  fi
```

Ouput :

```
  Enter first number
  15
  Enter second number
  39
  Enter third number
  7
  39 is greater than 15 and 7
```

V. Fibonacci Series

Aim : To write a shell program to generate fibonacci series.

Algorithm :

Step 1 : Initialise a to 0 and b to 1.
Step 2 : Print the values of ’a’ and ’b’.
Step 3 : Add the values of ’a’ and ’b’. Store the added value in variable ’c’.
Step 4 : Print the value of ’c’.
Step 5 : Initialise ’a’ to ’b’ and ’b’ to ’c’.
Step 6 : Repeat the steps 3,4,5 till the value of ’a’ is less than 10.

Program :

```
  #!/bin/sh
  a=0
  b=1
  echo "Enter n for series : "
  read n

  count=0

  echo "Printing the fibonacci series"

  while [ $count -lt $n ]
  do
    echo -n "$a "
    count=`expr $count + 1`
    c=`expr $a + $b`
    a=$b
    b=$c
  done
  echo " "
```

Output :

```
  Enter n for series :
  14
  Printing the fibonacci series
  0 1 1 2 3 5 8 13 21 34 55 89 144 233
```

VI. Arithmetic Operations using Case

Aim : To write a shell program to perform the arithmetic operations using case.

Algorithm :

Step 1 : Read the input variables and assign the value
Step 2 : Print the various arithmetic operations which we are going to perform
Step 3 : Using the case operator assign the various functions for the arithmetic operators.

Step 4 : Check the values for all the corresponding operations.
Step 5 : Print the result and stop the execution.

Program :

```
#!/bin/sh
echo "Enter first number"
read a
echo "Enter second number"
read b

echo "Operation type"
read o

case "$o" in
  "+") echo "Addition"
  c=`expr $a + $b`
  echo "$a + $b = $c"
  ;;

  "-") echo "Subtraction"
  c=`expr $a - $b`
  echo "$a - $b = $c"
  ;;

  "/") echo "Divide"
  c=`expr $a / $b`
  echo "$a / $b = $c"
  ;;

  "*") echo "Multiply"
  c=`expr $a \* $b`
  echo "$a * $b = $c"
  ;;

  "%") echo "Modulo division"
  c=`expr $a % $b`
  echo "$a % $b = $c"
  ;;
esac
```

Ouput :

```
Enter first number
15
Enter second number
39
Operation type
-
Subtraction
15 - 39 = -24
```

VII. Conclusion

In this lab report, various programs were implemented using shell programming to perform string concatenation, string comparison, finding the maximum of three numbers, generating the Fibonacci series, and carrying out various arithmetic operations based on user input. Shell programming is a useful tool for automating tasks and carrying out system administration tasks efficiently. These programs demonstrate the power of shell programming and its ability to handle complex tasks with ease. By gaining knowledge of shell programming, individuals can become more proficient in system administration and be better equipped to handle various tasks related to operating systems.