VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CSE 2005: OPERATING SYSTEMS

LAB L41+L42

DIGITAL ASSIGNMENT 1

Submitted by: ANUSHKA DIXIT

[19BCE0577]

Part 1:

Basic Command Execution

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ history
    1  man<mkdir>
    2  man <mkdir>
    3  mkdir file1
    4  pwd
    5  cat
    6  cat Game1
    7  cat Game 1
    8  mkdir newdir
    9  cd newdir
   10  finger
   11  sudo apt install finger
   12  ls newdir
   13  ls os1
   14  mkdir /tmp/tutorial
   15  cd /tmp/tutorial
   16  mk dir1 dir2 dir3
   17  mkdir dir1 dir2 dir3
   18  ls
   19  ls > output.txt
   20  ls
   21  cat output.txt
   22  echo "Hello World, I call myself a programmer" > test.txt
   23  ls
   24  cat test.txt
   25  cat output.txt test.txt
   26  echo "This is a test" > test1.txt
   27  ls
   28  cat output.txt test1.txt test.txt
   29  cp test1.txt test_2.txt
   30  cat test_2.txt
   31  mv test_2.txt new.txt
   32  ls
   33  rm new.txt
   34  ls
   35  find output.txt
   36  grep -i test file1.txt
   37  grep -i test test.txt
   38  cat test.txt
   39  cat test1.txt
   40  wc -l test.txt
   41  wc test.txt
   42  who
   43  who -b
   44  quota -v
   45  sudo apt install quota
   46  quota -v
   47  df
   48  du
   49  vi test_1.txt
   50  history
   51  grep test.txt
```

Basic Bash Command Execution

1)Bash printing

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                          BasicEcho.sh
#!/bin/sh
#pwd
variable ="Hello";
echo $variable
```

Output:

```
anushka_os@DESKTOP-96L9A8G:~$ nano
anushka_os@DESKTOP-96L9A8G:~$ chmod u+x ./BasicEcho.sh
anushka_os@DESKTOP-96L9A8G:~$ ./BasicEcho.sh
Hello!
anushka_os@DESKTOP-96L9A8G:~$
```

2)String Concatenation

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                          read.sh
#!/bin/bash
echo "What is your name?"
read name
echo "How are you doing, $name?"
read remark
echo "I am $remark too!"
```

Output:

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ nano
anushka_os@DESKTOP-96L9A8G:~$ chmod u+x ./read.sh
anushka_os@DESKTOP-96L9A8G:~$ ./read.sh
What is your name?
Anushka
How are you doing, Anushka?
great
I am great too!
anushka_os@DESKTOP-96L9A8G:~$
```

Part 2:

Question 1

Write a shell script to swap two numbers without using third variable

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                    lab1.sh
#1/bin/bash
read var1
read var2
echo "Before Swapping"
echo "First number: $var1"
echo "Second number:$var2"

var1 =$((var1-var2))
var2=$((var1+var2))
var1=$((var2-var1))

echo "After Swapping"
echo "First number:$var1"
echo "Second number:$var2"

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line
```

Output:

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ ./lab11.sh
-bash: ./lab11.sh: Permission denied
anushka_os@DESKTOP-96L9A8G:~$ chmod u+x lab11.sh
anushka_os@DESKTOP-96L9A8G:~$ ./lab11.sh
45
65
Before Swapping
First number:45
Second number:65
After Swapping
First number:65
Second number:45
anushka_os@DESKTOP-96L9A8G:~$
```

Question 2

Write a shell script to write sum of first ‘N’ numbers in Fibonacci series

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                          lab46.sh
#!/bin/bash
echo "Enter the number of terms"
read N
a=0
b=1
sum=0
echo "Fibonacci series is:"
for((i=0; i<N; i++))
do
        echo "$a"
        s=$((a+b))
        a=$b
        b=$s
        sum=$((s-1))
done
echo "Sum of the terms is:"
echo "$sum";
```

Output:

```
anushka_os@DESKTOP-96L9A8G:~$ nano
anushka_os@DESKTOP-96L9A8G:~$ chmod u+x ./lab46.sh
anushka_os@DESKTOP-96L9A8G:~$ ./lab46.sh
Enter the number of terms
5
Fibonacci series is:
0
1
1
2
3
Sum of the terms is:
7
```

```
anushka_os@DESKTOP-96L9A8G:~$ ./lab46.sh
Enter the number of terms
8
Fibonacci series is:
0
1
1
2
3
5
8
13
Sum of the terms is:
33
```

Question 3

Write a shell script to print the sum of all digits on the given number

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                                    lab3.sh
#!/bin/bash
echo "Enter number:"
read n
reverse=0
sum=0
while((n>0))
do
        reverse=$((reverse*10+n%10))
        sum=$((sum+n%10))
        n=$((n/10))
done
echo $sum
```

Output:

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ chmod u+x ./lab3.sh
anushka_os@DESKTOP-96L9A8G:~$ ./lab3.sh
Enter number:
784
487
19
anushka_os@DESKTOP-96L9A8G:~$ ./lab3.sh
Enter number:
6943
22
anushka_os@DESKTOP-96L9A8G:~$ ./lab3.sh
Enter number:
2581
16
anushka_os@DESKTOP-96L9A8G:~$
```

Question 4

Write a shell script to read two strings and display whether it is equal, not equal, null strings or string with special characters.

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8
#!/bin/bash

read VAR1
read VAR2
## syntax 1 ##
if [[ "$VAR1" = "$VAR2" ]]; then
   echo "Strings are equal"
else
   echo "Strings are not equal"
fi
if [[ -z "$VAR1" ]]; then
   echo "Empty $VAR1"
else
   echo "$VAR1 is not empty"
fi

if [[ $VAR1||$VAR2 == *['!'@#\$%^\&*()_+]* ]];then
  echo "String has special chatacters"
else
  echo "Strings are not equal"
fi
```

Output:

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ chmod u+x ./lab44.sh
anushka_os@DESKTOP-96L9A8G:~$ ./lab44.sh
car@
car@
Strings are equal
car@ is not empty
String has special chatacters
anushka_os@DESKTOP-96L9A8G:~$ nano
anushka_os@DESKTOP-96L9A8G:~$ ./lab44.sh
war
yta
Strings are not equal
war is not empty
String has special chatacters
anushka_os@DESKTOP-96L9A8G:~$
```

Question 5

Write a shell script to accept one integer argument and print its multiplication table.

Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                                    lab4.sh
#!/bin/bash
echo "Enter a Number"
read n

echo "Enter Range"
read r
i=0
while [ $i -le $r ]
do
    echo " $n x $i = `expr $n \* $i`"
    i=`expr $i + 1`
done
```

Output:

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ ./lab4.sh
Enter a Number
2
Enter Range
5
 2 x 0 = 0
 2 x 1 = 2
 2 x 2 = 4
 2 x 3 = 6
 2 x 4 = 8
 2 x 5 = 10
anushka_os@DESKTOP-96L9A8G:~$ ./lab4.sh
Enter a Number
9
Enter Range
5
 9 x 0 = 0
 9 x 1 = 9
 9 x 2 = 18
 9 x 3 = 27
 9 x 4 = 36
 9 x 5 = 45
anushka_os@DESKTOP-96L9A8G:~$
```

Part 3:

Fork Commands

1) Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main()
{

    // make two process which run same
    // program after this instruction
    fork();

    printf("Hello world!\n");
    return 0;
};
```

Output:

```
anushka_os@DESKTOP-96L9A8G:~$ ./fork
Hello world!
Hello world!
anushka_os@DESKTOP-96L9A8G:~$
```

2) Code:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8
#include <stdio.h>
#include <sys/types.h>
int main()
{
    fork();
    fork();
    fork();
    printf("hello\n");
    return 0;
}
```

Output:

```
anushka_os@DESKTOP-96L9A8G:~$ nano
anushka_os@DESKTOP-96L9A8G:~$ gcc fork2.c -o fork2
fork2.c: In function ‘main’:
fork2.c:5:5: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
    5 |     fork();
      |     ^~~~
anushka_os@DESKTOP-96L9A8G:~$ ./fork2
hello
hello
hello
hello
hello
hello
hello
hello
anushka_os@DESKTOP-96L9A8G:~$
```

3) Code:

```
anushka_os@DESKTOP-96L9A8G: ~
GNU nano 4.8                                                    fork3.c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
void forkexample()
{
    // child process because return value zero
    if (fork() == 0)
        printf("Hello from Child!\n");

    // parent process because return value non-zero.
    else
        printf("Hello from Parent!\n");
}
int main()
{
    forkexample();
    return 0;
}
```

Output:

```
anushka_os@DESKTOP-96L9A8G:~$ na
anushka_os@DESKTOP-96L9A8G:~$ gc
anushka_os@DESKTOP-96L9A8G:~$ ./
Hello from parent
Hello from child
```

TASK 1: Execution of exec

Linux Exec System Call: The **exec() system call** is used to **execute** a file which is residing in an active process. When **exec** is called the previous executable file is replaced and new file is executed.

Process 1

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8                                                    EXEC.c
#include<stdio.h>
#include<unistd.h>

int main()
{
    int i;

    printf("Hello world! Execution of exec. Replaces the cureent runnig program with a new process! ");
    printf("\n");

    return 0;
}
```

Process 2

```
Select anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
int main()
{

        char *args[]={"./EXEC",NULL};
        execvp(args[0],args);
        //THE FOLLOWING CODE IS IGNORED AS A PROCESS HAD TAKEN OVER

        printf("Ending-----");

    return 0;
}
```

Output

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ gcc EXEC.c -o EXEC
anushka_os@DESKTOP-96L9A8G:~$ gcc EXECdemo.c -o EXECdemo
anushka_os@DESKTOP-96L9A8G:~$ ./EXECdemo
Hello world! Execution of exec. Replaces the cureent runnig program with a new process!
anushka_os@DESKTOP-96L9A8G:~$
```

TASK 2: Execution of wait

Linux Wait System Call: The **wait**() **system call** suspends execution of the current process until one of its children terminates or a signal is received

Program:

```
anushka_os@DESKTOP-96L9A8G: ~
  GNU nano 4.8
#include<stdio.h>
#include<sys/wait.h>
#include<unistd.h>

int main()
{
    if (fork()== 0)
        printf("Hello from child\n");
    else
    {
        printf("Hello from parent\n");
        wait(NULL);
        printf("Child has terminated\n");
    }

    printf("Process ended\n");
    return 0;
}
```

Output:

```
anushka_os@DESKTOP-96L9A8G:~$ nano
anushka_os@DESKTOP-96L9A8G:~$ gcc WAIT.c -o WAIT
anushka_os@DESKTOP-96L9A8G:~$ ./WAIT
Hello from parent
Hello from child
Process ended
Child has terminated
Process ended
anushka_os@DESKTOP-96L9A8G:~$
```

TASK 2: Execution of kill

Linux Wait System Call: Kill command Linux is normally used to kill a suspended or hanging process or process group.

Execution:

```
anushka_os@DESKTOP-96L9A8G: ~
anushka_os@DESKTOP-96L9A8G:~$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
anushka_os@DESKTOP-96L9A8G:~$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
anushka+   124   123  0 21:20 tty1     00:00:00 -bash
anushka+   137   124  0 21:20 tty1     00:00:00 ps -f
anushka_os@DESKTOP-96L9A8G:~$ kill -9 124
```

Result:

Since the kill command was used to terminate process with PID 124, which is the –bash process, the terminal closed.

Anushka Dixit [19BCE0577]