System Programming lab

1. Write a Unix Shell that detects all lines containing a specific word in one or more file .

Syntax:

grep [options] pattern [filename]

The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern.(grep stands for globally search for regular expression and print out).

options:

-i → ignores uppercase & lowercase distinction

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# cat demo.txt
Hi, welcome to my file.
I'm a demo file.
Good Morning Everyone.

root@LAPTOP-DJ7TTK7Q:~/folder_1# grep -i file demo.txt
Hi, welcome to my file.
I'm a demo file.
```

All lines containing the word “file”, regardless of if it's lowercase, uppercase or a mix of both, are returned.

Options:

-n → outputs adds line numbers to the output

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# grep -n file demo.txt
1:Hi, welcome to my file.
2:I'm a demo file.
```

Options:

-c → Display how many lines contain the search pattern

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# grep -c file demo.txt
2
```

Options:

-o → Show only the part of line matching pattern

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# grep -o file demo.txt
file
file
```

2. Write a Unix Shell Script to find the given input file is a regular files , block file, character file , directory file or any other .

```
Bash File Testing

-b filename - Block special file
-c filename - Special character file
-d directoryname - Check for directory Existence
-e filename - Check for file existence, regardless of type (node, directory, socket, etc.)
-f filename - Check for regular file existence not a directory
-G filename - Check if file exists and is owned by effective group ID
-O filename - True if file exists and is owned by the effective user id
-r filename - Check if file is a readable
-w filename - Check if file is writable
-x filename - Check if file is executable
```

Syntax with example:

file_operations.sh: (create a shell script with ".sh" extension and execute it in terminal)

1. "-e" → check for file existence

```
echo -e "Enter the file name : \c"
read file_name

if [ -e $file_name ]
then
        echo "$file_name found"
else
        echo "$file_name not found!"
fi
```

Execute it by typing, "./file_operations.sh"

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# ls
demo.txt  dir  file_operations.sh
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./file_operations.sh
Enter the file name : demo.txt
demo.txt found
```

2. "-f" → checks for regular file existence

```
echo -e "Enter the file name : \c"
read file_name

if [ -f $file_name ]
then
        echo "$file_name is regular"
else
        echo "$file_name not regular!"
fi
```

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./file_operations.sh
Enter the file name : demo.txt
demo.txt is regular
```

3. “-b” → checks for block file existence

```
echo -e "Enter the file name : \c"
read file_name

if [ -b $file_name ]
then
        echo "$file_name is block file"
else
        echo "$file_name not a block file!"
fi
```

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./file_operations.sh
Enter the file name : demo.txt
demo.txt not a block file!
```

4. “-c” → checks for character file existence

```
echo -e "Enter the file name : \c"
read file_name

if [ -c $file_name ]
then
        echo "$file_name is character file"
else
        echo "$file_name not a character file!"
fi
```

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./file_operations.sh
Enter the file name : demo.txt
demo.txt not a character file!
```

5. “-c” → checks for directory existence

```
echo -e "Enter the file name : \c"
read file_name

if [ -d $file_name ]
then
        echo "$file_name exists"
else
        echo "$file_name doesn't exist!"
fi
```

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./file_operations.sh
Enter the file name : dir
dir exists
```

3. Design and develop a file in java program to simulate LS command .

The File API (java.io.File) is very flexible and portable, but in this example I'll just list files and directories of the current directory.

First, create a java file.

```
import java.io.File;

public class JavaLS{
        public static void main(String args[]){
                File dir = new File(System.getProperty("user.dir"));
                String childs[] = dir.list();
                for(String child: childs){
                        System.out.println(child);
                }
        }
}
```

Execute the java file.

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# javac JavaLS.java
root@LAPTOP-DJ7TTK7Q:~/folder_1# java JavaLS
JavaLS.class
JavaLS.java
demo.txt
dir
file_operations.sh
```

As you can see, it list the files and directories of the current directory.

4. Write a Unix shell script with option to extract substring from a string ,find the length of the given string.

To find length of the given string,

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# string=Linux
root@LAPTOP-DJ7TTK7Q:~/folder_1# echo ${string}
Linux
root@LAPTOP-DJ7TTK7Q:~/folder_1# echo ${#string}
5
```

If we put "#" before the variable name, which is variable *string* in our case, it displays length of the given string.

To extract substring from a string,

Sytntax:

string:P:L

P → is the number that indicates the starting index of the substring.

L → is the length of the substring.

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# echo ${string:0:3}
Lin
```

If we ommit the 'L' parameter, then the rest of string is returned, starting from position 'P'.

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# echo ${string:0}
Linux
root@LAPTOP-DJ7TTK7Q:~/folder_1# echo ${string:3}
ux
```

5. Write a shell program to perform the factorial of given number .

First, create a shell script with ".sh" extension, in our case, "factorial.sh"

```
#shell script for factorial of a number
#factorial using while loop

echo "Enter a number"
read num

fact=1

while [ $num -gt 1 ]
do
  fact=$((fact * num))  #fact = fact * num
  num=$((num - 1))      #num = num - 1
done

echo $fact
```

Execute it, "./factorial.sh"

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# nano factorial.sh
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./factorial.sh
Enter a number
5
120
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./factorial.sh
Enter a number
10
3628800
```

6. Write a Unix Shell Script to extract file pattern and display the no. of times pattern is accurred in the given file .

Syntax:

```
grep -wc $word $filename
```

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# cat demo.txt
Hi, welcome to my file.
I'm a demo file.
Good Morning Everyone.

root@LAPTOP-DJ7TTK7Q:~/folder_1# grep -wc file demo.txt
2
```

Options Description

-c : This prints only a count of the lines that match a pattern
-h : Display the matched lines, but do not display the filenames.
-i : Ignores, case for matching
-l : Displays list of a filenames only.
-n : Display the matched lines and their line numbers.
-v : This prints out all the lines that do not matches the pattern
-e exp : Specifies expression with this option. Can use multiple times.
-f file : Takes patterns from file, one per line.
-E : Treats pattern as an extended regular expression (ERE)
-w : Match whole word
-o : Print only the matched parts of a matching line,
with each such part on a separate output line.

7. Write a Unix Shell Script to print multiplication table for given number .

First, create a shell script with ".sh" extension, in our case, "multiplication_table.sh"

```
echo "Enter the Multilication number required :"
read number

for i in 1 2 3 4 5 6 7 8 9 10
do
echo "$number * $i = `expr $number \* $i`"
done
```

Execute it, "./multiplication_table.sh"

```
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./multiplication_table.sh
Enter the Multilication number required :
17
17 * 1 = 17
17 * 2 = 34
17 * 3 = 51
17 * 4 = 68
17 * 5 = 85
17 * 6 = 102
17 * 7 = 119
17 * 8 = 136
17 * 9 = 153
17 * 10 = 170
root@LAPTOP-DJ7TTK7Q:~/folder_1# ./multiplication_table.sh
Enter the Multilication number required :
18
18 * 1 = 18
18 * 2 = 36
18 * 3 = 54
18 * 4 = 72
18 * 5 = 90
18 * 6 = 108
18 * 7 = 126
18 * 8 = 144
18 * 9 = 162
18 * 10 = 180
```