PSG COLLEGE OF TECHNOLOGY

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCES

UNIX SHELL AND SYSTEM PROGRAMMING LAB

PROBLEM SHEET -3

1. grep

a) Write a grep command that selects the lines from the file1 that have exactly three characters

```
$ grep  ^...$  filename
```

b) Write a grep command that selects the lines from the file1 that have at least three characters.

```
$ grep  ...  filename
```

c) Write a grep command that selects the lines from the file1 that have three or fewer characters

```
$ grep  -v  ....  filename
```

d) Write a grep command that count the number blank lines in the file1

```
$ grep  -c  ^$  filename
```

e) Write a grep command that count the number nonblank lines in the file1

```
$ grep  -cv  ^$  filename
```

f) Write a grep command that selects the lines from the file1 that have the string UNIX.

```
$ grep  "UNIX"  filename
```

g) Write a grep command that selects the lines from the file1 that have only the string UNIX.

```
$ grep  ^UNIX$  filename
```

h) Write a grep command that copy the file to the monitor, but delete the blank lines.

```
$ grep  -v  ^$  filename
```

i) Write a grep command that selects the lines from the file1 that have at least two digits without any other characters in between

```
$ grep  [0-9][0-9]  filename
```

j) Write a grep command that selects the lines from the file1 that do not start with A to G

```
$ grep  -v  ^[A-G]  filename
```

2. sed

a) Write a sed command that print numbers of lines beginning with "O"

```
$ sed  -ne  '/^O/ ='  filename
```

b) Write a sed command that delete digits in the given input file.

```
$ sed  's/[0-9]*//g'  filename
```

c) Write a sed command that delete lines that contain both BEGIN and END

```
$ sed  '/begin.*end/d '  filename
```

d) Write a sed command that delete lines that contain BEGIN but not END

```
$ sed  -ne  '/begin/p'  filename | sed  '/end/!d'
```

e) Write a sed command that deletes the first character in each line in a file

```
$ sed  's/^.//g '  filename
```

f) Write a sed command that deletes the last character in each line in a file

```
$ sed  's/.$//g '  filename
```

g) Write a sed command to delete character before last character in each line in a file

```
$ sed  's,\(.*\)\(.\)\(.\),\1\3,g '  filename
```

h) Write a sed command that swaps the first and second character in each line in the file

```
$ sed  's,\(.\)\(.\)\(.*\),\2\1\3,g '  filename
```

i) Write a sed command that swaps the first and second word in each line in the file

```
$ sed  's,\([^ ]*\) *\([^ ]*\),\2 \1 ,g '  filename
```

3. sed

a) Create a text file as an input as shown below

```
>cat file.txt
unix is great os. unix is opensource. unix is free os.
learn operating system.unixlinux which one you choose.
```

b) Sed command to replaces the first occurrence of the word "unix" with "linux" in the file.

```
sed 's/unix/linux/' file.txt
```

c) Sed command to replaces the second occurrence of the word "unix" with "linux" in a line.

```
sed 's/unix/linux/2' file.txt
```

d) Sed command to replace all the occurrences of the string “unix” with “linux”

```
sed 's/unix/linux/g' file.txt
```

e) Sed command to replaces the third, fourth, fifth... "unix" word with "linux" word in a line.

```
sed 's/unix/linux/3g' file.txt
```

f) Sed command search for the pattern “unix” and replace that pattern by “{unix}” (Hint: use &)

```
sed 's/unix/{&}/' file.txt
```

g) Replace the word "unix" in a line with twice as the word like "unixunix" using the sed command

```
sed 's/\(unix\)/\1\1/' file.txt
```

h) Replace the word “unix” with “linux” and

a. Print the replaced line twice on the terminal. If a line does not have the search pattern and is not replaced, then the prints that line only once.

```
sed 's/unix/linux/p' file.txt
```

b. Print only the replaced lines

```
sed -n 's/unix/linux/p' file.txt
```

i) Sed command to replace the string “unix” with “linux” on a specific line number 3.

```
sed '3 s/unix/linux/' file.txt
```

j) Sed command to look for the lines which has the pattern "linux" and then replaces the word "unix" in that line with "centos".

```
sed '/linux/ s/unix/centos/' file.txt
```

k) Sed command to delete the second line from the file

```
sed '2 d' file.txt
```

l) Sed command to print each line of a file two times.

```
sed 'p' file.txt
```

m) Sed command looks for the pattern "unix" in each line of a file and prints those lines that has the pattern.

```
sed -n '/unix/ p' file.txt
```

n) Sed command looks for the pattern "unix" in each line of a file and prints those lines that does not have the pattern.

```
sed -n '/unix/ !p' file.txt
```

o) Sed command to add a new line “new line” after a pattern “unix” is found

```
sed '/unix/ a "new line"' file.txt
```

p) Sed command to add a new line “new line” before a pattern “unix” is found in the file

```
sed '/unix/ i "Add a new line"' file.txt
```

q) Sed command to replace an entire line with pattern “unix” with a new line “Change line” in the file

```
sed '/unix/ c "Change line"' file.txt
```

r) Sed command to convert the lower case letters “u” and “l” to upper case letters “U” and “L” in the file

sed 'y/ul/UL/' file.txt

2. awk

Some string functions in awk

index(string,search)
length(string)
split(string,array,separator)
substr(string,position)
substr(string,position,max)
tolower(string)
toupper(string)

1. Write an awk command to print the lines and line number in the given input file .

```
$ cat > cmds.awk
{print NR, $0 }
$ awk -f cmds.awk fileinput
```

2. Write an awk command to print first field and second field only if third field value is >=50 in the given input file. (input field separator is “:” and output field separator is “,”)

input data file

```
$cat > file1
sachin:10:100
rahul:11:35
rohit:12:89
```

```
$awk -F':' '$3>=50 {print $1”,”$2}' file1
```

output:

```
sachin,10
rohit,12
```

3 . Consider the marks.txt is a file that contains one record per line(comma separate fields) of the student data in the form of studentid, student name, Telugu marks, English marks, Maths Marks, Science marks, Social Marks. Write an awk script to generate result for every students in the form of studentid, studentname, Total Marks and result. Result is PASS if marks is >=30 in TELUGU and English, and if marks>=40 in other subjects. Result is fail otherwise.

Input file

```
$cat > marks.txt
  1001,name1,99,69,85,56,75
  1002,name2,89,69,65,56,55
  1003,name3,50,50,50,55,55
  1004,name4,69,29,85,56,75
  1005,name5,99,69,85,56,11
```

create marks.awk script file

```
 $cat > marks.awk

   {

    total=$3+$4+$5+$6+$7

    if($3>=30 &&  $4>=30 && $5>=40 && $6>=40 && $7>=40)

        print $1,$2,total, "Pass";

    else

        print $1,$2,total, "fail";

   }
```

```
$awk -F “,” -f marks.awk marks.txt
```

---

4 Write an awk program to print the fields 1 and 4 of a file that is passed as command line argument. The file contains lines of information that is separated by “,” as delimeter. The awk program must print at the end the average of all 4th field data.

Input file

```
$cat > data

 12,13,14,15,one

 22,23,24,25,two
```

34,23,45,23,three

44,55,66,77,four

```
$awk -F',' '{print $1,$2,$3,$4,($1+$2+$3+$4)/4}'  data
```

5.Write an awk program to demonstrate user defined functions and system command.

```
$cat > data

12,13,14,15,one

22,23,24,25,two

34,23,45,23,three

44,55,66,77,four


$cat >user.awk
  {
   if($3>0)
       display($3)
  }
  function display(name)
  {
    print name
  }

$awk -F',' -f user.awk data
```

6. Write an awk script to count the number of lines in a file that do not contain vowels.

Input file

$cat > input

this is one

213

BCDEFG

This is last line

```
$cat vowels.awk
BEGIN{count=0}
!/[aeiou]/ {count++;print}
END{print "Number of lines="count}

 $awk -f vowels.awk input
```

7. Write an awk script to find the number of characters, words and lines in a file.

   Input file:

```
$cat > file7
        This is a file
         YEs NO
        1234
 $cat > lines.awk
 BEGIN{words=0;characters=0}
 {
 character+=length($0);
words+=NF;
 }
  END{print "lines=",NR," words=",words," Characters=",character}
$awk -f lines.awk file7
```

8. Awk command to read a list of integers from the file and print the total

```
awk '{ sum += $1 }; END { print sum }' file
```

9. Display the number of columns in each row.

```
awk '{print NF}' input_file
```

10. Display the line numbers from 1.

```
awk '{print NR}' input_file
```

11. Display the total number of lines in the filE

```
awk 'END {print NR}' input_file
```

12. awk script to insert a new line ‘9Z’ after every 2 line in file.txt

The input "file.txt" contains the below data:

```
1 A
2 B
3 C
4 D
5 E
6 F
```

The required output data after inserting a new line looks as

```
1 A
2 B
9 Z
3 C
4 D
9 Z
5 E
6 F
9 Z
```

```
awk '{
 if(NR%2 == 0)
 {
          print $0"\n9 Z";
 }
 else
 {
          print $0
 }
 }' file.txt
```

13. The input file contains the data.

AAA 1
BBB 2
CCC 3
AAA 4
AAA 5
BBB 6
CCC 7
AAA 8
BBB 9
AAA 0

Replace the fourth occurrence of the first field "AAA" with "ZZZ" in the file.

The required output is:

AAA 1
BBB 2
CCC 3
AAA 4
AAA 5
BBB 6
CCC 7
ZZZ 8
BBB 9
AAA 0

The awk command for getting this output is

```
awk 'BEGIN {count=0}
 {
  if($1 == "AAA")
  {
   count++
  }
  if(count == 4)
  {
   sub("AAA","ZZZ",$1)
  }
 }
 {
  print $0
}' file.txt
```

15. The input file data:

A 10
B 39
C 22

D 44
E 75
F 89
G 67

Write an awk script to get the second field and then find the sum the even and odd lines.
The required output is

174, 172

The awk command for producing this output is

```
awk '{
 if(NR%2 == 1)
 {
  sum_e = sum_e + $2
 }
 else
 {
  sum_o = sum_o + $2
 }
 }
      END { print sum_e,sum_o }' file.txt
```

16. Fibonacci series using awk command

```
wk ' BEGIN{

 for(i=0;i<=10;i++)
 {
  if (i <=1 )
  {
   x=0;
   y=1;
   print i;
  }
  else
  {
   z=x+y;
   print z;
   x=y;
   y=z;
  }
 }
    }'
```

17. Remove leading zeros from a file using the awk command. The input file contains the below data.

0012345
05678
01010
00001

After removing the leading zeros, the output should contain the below data.

12345
5678
1010
1

The awk command for this is.

```
awk '{print $1 + 0}' file.txt
awk '{printf "%d\n",$0}' file.txt
```

18. The file employee.txt contains the following data

Tom	Manager	Sales	$5,000
Jason	Developer	Technology	$5,500
Sanjay	Sysadmin	Technology	$7,000
Nisha	Manager	Marketing	$9,500
Randy	DBA	Technology	$6,000

a) write an awk script to produce the following output

Name	Designation	Department	Salary
Tom	Manager	Sales	$5,000
Jason	Developer	Technology	$5,500
Sanjay	Sysadmin	Technology	$7,000
Nisha	Manager	Marketing	$9,500
Randy	DBA	Technology	$6,000

Report Generated

```
$ awk 'BEGIN {print "Name\tDesignation\tDepartment\tSalary";}
> {print $1,"\t",$2,"\t",$3,"\t",$NF;}
> END{print "Report Generated\n--------------";
> }' employee.txt
```

b) Print the list of employees in Technology department

```
$ awk '$4 ~/Technology/' employee.txt
200  Jason   Developer  Technology  $5,500
300  Sanjay  Sysadmin   Technology  $7,000
500  Randy   DBA        Technology  $6,000
```

c) Print number of employees in Technology department

```
$ awk 'BEGIN { count=0;}
$4 ~ /Technology/ { count++; }
END { print "Number of employees in Technology Dept =",count;}' employee.txt
```

19. To print non-empty line from a file.

```
$ awk 'NF > 0' sample.txt
```

20. To print the length of the longest input line.

```
$ awk '{ if (length($0) > max) max = length($0) } END { print max }'
sample.txt
```

21. To print seven random numbers from zero to 100, inclusive.

```
$  awk 'BEGIN { for (i = 1; i <= 7; i++) print int(101 * rand()) }'
```

22. To count the lines in a file

```
$ awk 'END { print NR }' sample.txt
```

23. Awk script that asks for a number and then squares it.

```
BEGIN {
   print "type a number";
}
{
   print "The square of ", $1, " is ", $1*$1;
   print "type another number";
}
END {
   print "Done"
}
```