E. ✔ Time complexity for function `mystery` is $O(nm)$

# System Commands

**Number of Questions :** 20

**Section Marks :** 50

**Question Number : 191 Question Type : MCQ**

**Correct Marks : 0**

Question Label : Multiple Choice Question

**THIS IS QUESTION PAPER FOR THE SUBJECT "DIPLOMA LEVEL: SYSTEM COMMANDS"**

**ARE YOU SURE YOU HAVE TO WRITE EXAM FOR THIS SUBJECT?**
**CROSS CHECK YOUR HALL TICKET TO CONFIRM THE SUBJECTS TO BE WRITTEN.**

**(IF IT IS NOT THE CORRECT SUBJECT, PLS CHECK THE SECTION AT THE TOP FOR THE SUBJECTS REGISTERED BY YOU)**

**Options :**

A. ✔ YES

B. ✖ NO

**Question Number : 192 Question Type : MCQ**

**Correct Marks : 2**

Question Label : Multiple Choice Question

Which of the following shell commands will concatenate the two files `f1.txt` and `f2.txt` into a new file named `f3.txt`.

**Options :**

A. ✖ `mv f1.txt f2.txt f3.txt`

B. ✔ `cat f1.txt f2.txt > f3.txt`

C. ✖ `cp f1.txt f2.txt f3.txt`

D. ✖ `mv f1.txt f2.txt | f3.txt`

**Question Number : 193 Question Type : MCQ**

**Correct Marks : 2**

Question Label : Multiple Choice Question

What is the output of the following bash script?

Note: Operators `||` and `&&` have same precedence. And the operations will be executed from left to right.

```
#!/bin/bash
x=1
y=10
z=2
if [ $x -eq 1 ] && [ $y -gt 5 ] || [$z -eq 10 ]
then
 echo $y/$z
else
 echo $z/$x
fi
```

**Options :**

A. ✖ 5

B. ✖ 2

C. ✔ 10/2

D. ✖ 2/1

**Question Number : 194 Question Type : MCQ**

**Correct Marks : 2**

Question Label : Multiple Choice Question

Choose the output of the following bash script if the input to stdin is `123321`.

```
#!/bin/bash

read line
rline=`echo $line | rev` # rev command will reverses the input string
if [[ $line == $rline ]]; then
  exec bash -c "echo It is a palindrome"
fi
echo It is not a palindrome
```

**Options :**

A. ✔
```
It is a palindrome
```

B. ✖
```
It is not a palindrome
```

C. ✖
```
It is a palindrome
It is not a palindrome
```

D. ✖ No output will be shown

**Question Number : 195 Question Type : MCQ**

**Correct Marks : 2**

Question Label : Multiple Choice Question

Consider a file named `fnames` that contains the first name of a student in each line, located in the current working directory. What will be the output of the below command?

```
awk '++seen[$0] == 2 {print $0}' fnames
```

**Options :**

A. ✖ Names that are present only one time.

B. ✔ Names that are repeated more than one time.

C. ✖ Distinct names with the number of occurrences.

D. ✖ Prints only the distinct names present in the file.

**Question Number : 196 Question Type : MCQ**

**Correct Marks : 2**

Question Label : Multiple Choice Question

Consider a file `fnames` that contains the first name of a student in each line, located in the current working directory. What will be the output of running the below awk script on the input as file `fnames`?

```
#!/usr/bin/awk -f
{
  fname[$0]++
}
END {
  for ( i in fname ) {
    print i, fname[i]
  }
}
```

**Options :**

A. ✖ Names that are present only one time.

B. ✖ Names that are repeated more than one time.

C. ✔ Distinct names with the number of occurrences.

D. ✖ Prints only the distinct names present in the file.

**Question Number : 197 Question Type : MCQ**

**Correct Marks : 2**

Question Label : Multiple Choice Question

What will be the output of the below command?

```
history | awk '{
  arr[$2]++
}
END {
  for (i in arr) {
    print arr[i],i
  }
}' | sort -rn | head -n 10
```

Note: `-n` and `-r` options in sort command are used for numerical sorting and reverse sorting respectively, and check the below sample history command output for the output format of history command. The below options refer `vim`, `cat`, `egrep` etc as commands, not the complete command syntax executed on the shell.

```
$ history
 1029  vim input
 1030  cat input
 1031  cat input | egrep "abd"
 1032  cat input | egrep "def"
 1033  cat input
 1034  vim input
```

**Options :**

A. ✖ First ten commands after the system booted.

B. ✖ The last ten commands executed.

C. ✖ Least frequently used ten commands.

D. ✔ Most frequently used ten commands.

**Question Number : 198 Question Type : MCQ**

**Correct Marks : 3**

Question Label : Multiple Choice Question

The file `file1.txt` contains the following text

```
abc,def
ghij,klm
n,o
```

What is the output of the below command?

```
sed 's/\(.*\)abc,/\1DEF/g' file1.txt | awk 'BEGIN{FS=",";}{print$2","$1}'
```

**Options :**

A. ✖

```
DEF,def
ghij,klm
n,o
```

B. ✖

```
def,abc
klm,ghij
o,n
```

C. ✔

```
,DEFdef
klm,ghij
o,n
```

D. ✖

```
DEF,def
klm,ghij
o,n
```

**Question Number : 199 Question Type : MCQ**

**Correct Marks : 3**

Question Label : Multiple Choice Question

Given below are the top five lines of access log file of a server. Each line corresponds to one

request received on the server. In each line

- The first field(before space), containing four numbers separated by a dot('.'), is the ip address of the client that made this request.
- The second field(inside square brackets), contains date, time and timezone offset. Frist 11 characters represents the date, followed by a colon, followed by time represented by next 8 characters, followed by a space and finally next 5 characters representing the timezoneoffset.

Note: In the below sample the lines are very big and hence wrapped. So there are only 5 lines, where each line starts with an ip address.

```
103.47.219.249 - - [27/Jan/2022:00:01:11 +0530] "GET / HTTP/1.1" 301 429 "-"
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/601.7.7 (KHTML, like
Gecko) Version/9.1.2 Safari/601.7.7"
54.209.123.136 - - [27/Jan/2022:00:01:18 +0530] "GET /AlloyOnto/AlloyOnto.owl HTTP/1.1"
301 494 "-" "Python-urllib/3.6"
54.209.123.136 - - [27/Jan/2022:00:01:18 +0530] "GET /AlloyOnto/AlloyOnto.owl HTTP/1.1"
301 494 "-" "Python-urllib/3.6"
54.209.123.136 - - [27/Jan/2022:00:01:19 +0530] "GET /AlloyOnto/AlloyOnto.owl HTTP/1.1"
200 1410215 "-" "Python-urllib/3.6"
54.209.123.136 - - [27/Jan/2022:00:01:19 +0530] "GET /AlloyOnto/AlloyOnto.owl HTTP/1.1"
200 1410215 "-" "Python-urllib/3.6"
```

What is the expected output of the below awk script, if executed on the complete access log file?

```
#!/usr/bin/awk -f

{
  datetime = $4":"$5
  time=substr(datetime, 14, 8)

  if (time < "06:00:00" ) {
    if ( $1 in ip ) { ip[$1]++ }
    else { ip[$1]=1 }
  }
}

END {
  mi=0
  for (i in ip) {
    if (ip[i] > mi) {
      mi = ip[i]
      mip = i
    }
  }
  print mip
}
```

**Options :**

A. ✖ The ip address of the client that made the least number of requests from midnight to 6 am.

B. ✖ The ip address of the client that made the least number of requests from 6 am to midnight.

C. ✔ The ip address of the client that made the most number of requests from midnight to 6 am.

D. ✖ The ip address of the client that made the most number of requests from 6 am to midnight.

**Question Number : 200 Question Type : MCQ**

**Correct Marks : 3**

Question Label : Multiple Choice Question

Consider the below state of the current working directory,

```
$ ls -l
total 78864
-rw-rw-rw- 1 user user     6274 Feb 14 15:25 access50.log
-rw-rw-r-- 1 user user 80386681 Feb 14 15:21 access-full.log
-rwxrw-r-- 1 user user      157 Feb  5 10:50 email.awk
-rw-rw-r-- 1 user user      180 Feb  4 11:27 employee_details.txt
-r-xrwxr-x 1 user user    16456 Feb  5 02:07 fact
-rwxrw-r-- 1 user user       62 Feb  5 02:06 f.c
--wx-wx-wx 1 user user        0 Feb 15 16:16 frep
-rw-rw-r-- 1 user user       81 Feb  5 11:56 groceries.csv
-rwxrwxrwx 1 user user        0 Feb 15 16:07 gui
-rw-rw-r-- 1 user user       46 Feb  4 21:19 sc
-rwxrw-rwx 2 user user      213 Feb  5 02:10 script.sh
-rwxrw-r-- 2 user user      397 Feb 14 17:42 temp.sh
-r--rw-rw- 1 user user        8 Feb  4 17:24 v
```

What will be the output of the below command , when executed in the current working directory?

```
ls -l | egrep "[-]([-rwx]{3})\1\1" | awk '{print $NF}'
```

**Options :**

A. ✖ Files/directories with the same set of permissions for user and group.

B. ✖ Files/directories with the same set of permissions for group and others.

C. ✖ Files/directories with the same set of permissions for user and others.

D. ✔ Files/directories with the same set of permissions for user, group and others.

**Question Number : 201 Question Type : MCQ**

**Correct Marks : 3**

Question Label : Multiple Choice Question

Consider a fruit shop selling only countable fruits like mango, apple, banana etc. The shopkeeper creates a file named `list_X` for each order, where `X` is the order number. Each order has a unique order number. Each file contains the information of fruits and quantity sold. Each line in the file contains two fields separated by a space, the first field is the fruit name and the second is the quantity of that fruit. A sample file for order number `13` is given below.

```
$ cat list_13
Mango 3
Pomegranate 5
Orange 3
```

The above file `file_13` tells that, for order number 13, 3 mangoes, 5 Pomegranates and 3 Oranges were sold.

Consider that all the files are located in the current working directory. The shopkeeper wants to calculate in total how many fruits were sold out. Choose the awk script which will print the total number of fruits sold by this shopkeeper till now. The below commands are executed in current working directory.

**Options :**

A. ✖ `awk '{ fruits[$1]++ } END { for (i in fruits) print i, fruits[i] }' list_*`

B. ✔ `awk '{ fruits[$1] += $2 } END { for (i in fruits) print i, fruits[i] }' list_*`

C. ✖ `awk '{ fruits[$2] += $1 } END { for (i in fruits) print i, fruits[i] }' list_*`

D. ✖ `awk '{ fruits[$1] += $2 } END { for (i in fruits) print i, fruits[i] }' list_`

**Question Number : 202 Question Type : MCQ**

**Correct Marks : 3**

Question Label : Multiple Choice Question

Consider the below information,

```
$ whatis last
last (1)             - show a listing of last logged in users

$ last | head -n 20
user pts/4        tmux(8855).%4    Wed Feb 16 16:09 - 16:20  (00:11)
user pts/4        tmux(8855).%3    Wed Feb 16 13:19 - 15:56  (02:37)
user pts/3        tmux(8855).%2    Wed Feb 16 10:40   still logged in
user pts/2        tmux(8855).%1    Wed Feb 16 10:20   still logged in
user pts/0        tmux(8855).%0    Wed Feb 16 10:19   still logged in
user :0           :0               Wed Feb 16 09:21   still logged in
reboot   system boot  5.13.0-28-generi Wed Feb 16 09:21   still running
user pts/6        tmux(2923).%7    Mon Feb 14 15:08 - 09:20 (1+18:12)
user pts/6        tmux(2923).%6    Mon Feb 14 15:07 - 15:07  (00:00)
user pts/5        tmux(2923).%5    Mon Feb 14 15:01 - 09:20 (1+18:19)
user pts/4        tmux(2923).%4    Mon Feb 14 15:01 - 09:20 (1+18:19)
user pts/3        tmux(2923).%3    Fri Feb 11 10:56 - 09:21 (4+22:24)
user pts/3        tmux(2923).%2    Fri Feb 11 10:17 - 10:56  (00:38)
user pts/2        tmux(2923).%1    Fri Feb 11 10:05 - 09:21 (4+23:15)
user pts/1        tmux(2923).%0    Fri Feb 11 09:21 - 09:21 (4+23:59)
user :0           :0               Fri Feb 11 09:20 - down  (5+00:00)
reboot   system boot  5.13.0-28-generi Fri Feb 11 09:20 - 09:21 (5+00:00)
user pts/4        tmux(34909).%3   Thu Feb 10 17:53 - 21:20  (03:27)
user pts/3        tmux(34909).%2   Thu Feb 10 17:34 - 21:20  (03:46)
user pts/2        tmux(34909).%1   Thu Feb 10 17:25 - 21:21  (03:55)
```

What will be the output of the below bash command?

```
last | grep . | grep -v reboot | awk '{ print $1 }' | sort --unique
```

Note: `sort --unique` command will sort the input and remove the duplicate lines.

**Options :**

A. ✖ List of users who have logged into the system since the last reboot.

B. ✖ List of users who have logged into the system before the last reboot.

C. ✔ List of users who have logged into the system previously.

D. ✖ List of users who failed to login to the system.

**Question Number : 203 Question Type : MCQ**

**Correct Marks : 3**

Question Label : Multiple Choice Question

Match the command used in vi editor with their use

| Commands | Function |
|---|---|
| 1. a | a. Move to start of the current line. |
| 2. 0(zero) | b. Enter into insert mode and write after the cursor. |
| 3. dd | c. Deletes a character at the position of the cursor. |
| 4. i | d. Goes into insert mode at the position of the cursor. |
| 5. x | e. Delete line. |

OR

Match the command used in emacs editor with their use. Note that Ctrl and Meta refers to Control and Alt key respectively.

| Commands | Function |
|---|---|
| 1. Ctrl+*n* | a. Move to start of the current line. |
| 2. Ctrl+*a* | b. Move to the next line. |
| *3. Ctrl+e* | c. Move forward a word. |
| 4. Ctrl+*b* | d. Move one character to the left. |
| 5. Meta+*f* | e. Move to the end of the current line. |

**Options :**

A. ✔ 1—>b; 2—>a; 3—>e; 4—>d; 5—>c;

B. ✖ 1—>a; 2—>b; 3—>c; 4—>d; 5—>e;

C. ✖ 1—>b; 2—>a; 3—>c; 4—>d; 5—>e;

D. ✖ 1—>a; 2—>b; 3—>e; 4—>d; 5—>c;

**Question Number : 204 Question Type : MSQ**

**Correct Marks : 2**

Question Label : Multiple Select Question

Which of the following commands can be used to match a valid phone number i.e. exactly in the format `twoDigit-tenDigit`, where `twoDigit` is a two digit number and `tenDigit` is a ten digit number and these two numbers are separated by a hyphen( - ), for example "91-9876543210" is a valid phone number. Every character as given in the pattern is required for a match including the hyphen. The command should print only the lines that contain a valid phone number anywhere in the file `file1.txt`.

**Options :**

A. ✔ `grep "[0-9]\{2\}[-][[:digit:]][[:digit:]]\{9\}" file1.txt`

B. ✔ `egrep "[[:digit:]]{2}[-][[:digit:]]{10}" file1.txt`

C. ✖ `egrep "[[:digit:]]{2}[-]?[[:digit:]]{10}" file1.txt`

D. ✖ `grep "[0-9][0-9][[:punct:]][[:digit:]].*" file1.txt`

**Question Number : 205 Question Type : MSQ**

**Correct Marks : 2**

Question Label : Multiple Select Question

In Linux any file/directory whose name starts with a `.` is considered a hidden file/directory, for e.g. `.file_hid` is a name of a hidden file. Command `la` is used to display all the files including the hidden files, it is similar to `ls -a`. Select the commands that show only the hidden files and directories in the home directory of the current user.

**Options :**

A. ✖ `la ~ | grep "^."`

B. ✔ `la ~ | grep "^\."`

C. ✖ `la ~ | egrep "^."`

D. ✔ `la ~ | grep -v "^[^.]"`

E. ✖ `la ~ | grep -v "^[^.]*"`

**Question Number : 206 Question Type : SA**

**Correct Marks : 2**

Question Label : Short Answer Question

How many lines will be printed by the given bash script?

```
#!/bin/bash

flag=0
for i in {1..20}; do
  if [[ $((i%7)) == 0 ]]; then
    continue
    flag=1
  fi
  if [[ $((i%2)) && flag == 1 ]]; then
    break
  fi
  echo $i
done
```

**NOTE:** Enter your answer to the nearest integer.

**Response Type :** Numeric

**Evaluation Required For SA :** Yes

**Show Word Count :** Yes

**Answers Type :** Equal

**Text Areas :** PlainText

**Possible Answers :**

18

**Question Number : 207 Question Type : SA**

**Correct Marks : 2**

Question Label : Short Answer Question

Consider the below information,

```
$ wc -l file
6 file
```

How many lines of text will be printed to stdout by the below command?

```
sed -e "1~2p" file
```

**NOTE:** Enter your answer to the nearest integer.

**Response Type :** Numeric

**Evaluation Required For SA :** Yes

**Show Word Count :** Yes

**Answers Type :** Equal

**Text Areas :** PlainText

**Possible Answers :**

9

**Question Number : 208 Question Type : MSQ**

**Correct Marks : 3**

Question Label : Multiple Select Question

Choose the commands that will show the csv file `file.csv` without first and last column.

**Options :**

A. ✖ `sed -e 's/[(^.*,)|(,.*$)]//g' file.csv`

B. ✖ `sed -e 's/^.*,//g' -e 's/,.*$//g' file.csv`

C. ✔ `awk 'BEGIN {FS=","} { for (i=2; i<NF; i++) printf $i",";printf "\n" }' file.csv`

D. ✔ `sed -e 's/^[^,.]*,//g' -e 's/,[^,.]*$//g' file.csv`

**Question Number : 209 Question Type : MSQ**

**Correct Marks : 3**

Question Label : Multiple Select Question

Choose the correct statement(s) based on the below bash script that is stored in a file named `script.sh`.

```
#!/bin/bash
opt="^-"
files="^-x"
prefix="^-p"
curropt=
for arg in "${@}"; do
  if [[ $arg =~ $opt ]]; then
    curropt=$arg
    continue
  fi
  if [[ $curropt =~ $files ]]; then
    arr+=($arg)
  elif [[ $curropt =~ $prefix ]]; then
    prefixtext=$arg
  fi
done
for i in ${arr[@]}; do mv "$i" "$prefixtext$i"; done
```

**Options :**

A. ✔ Commands `./script.sh -x 1 2 3 4 -p file` and `./script.sh -p file -x 1 2 3 4` will give the same output.

B. ✖ The arguments after `-x` option are limited to 4.

C. ✔ The value of `$1` after the completion of the outer while-loop will be the first command line argument given while running the script.

D. ✖ If the files/directories with absolute paths are given as arguments to `-x` option, then the files/directories will be renamed but will not be moved to other directories irrespective of the absolute paths given.

**Question Type : COMPREHENSION**

**Question Numbers : (210 to 211)**

Question Label : Comprehension

Consider the below command outputs,

```
$ ls
08ad5739dcd918615bba8dc86aa645f3
0fbe20919af19912c5c3a9dd8d911ce1
26c54d48bf60e8c63260ec5f24a2278e
4affcecd92c100f6caf3b07a7242a892
5069f6b45f6a5bc73c3acc21aba2c762
8755d0e42097448f27a584e0fc8c1037
87784fa86ab94449917ac98aa5e22d92
af2f43fefe6d0c519889947fffb011d7
e90ba71c91ecf45a53a25452d7ddabb2
eed46795a227c013280e9b0371f734fe

$ cat eed46795a227c013280e9b0371f734fe
[DETAILS]
2021R0001
Himesh Pratap Borase
himesh@studentmail.com
EDUCATIONAL_BACKGROUND:NON-TECHNICAL
[COURSES COMPLETED]
BSCMA1001
BSCMA1003
BSCCS1001
BSCCS1002

BSCHS1001
BSCHS1002
BSCMA1002
BSCMA1004
BSCSE2001
BSCCS2001
[COURSES ENROLLED]
BSCCS2002
BSCCS2003
```

Each file in the current working directory contains details of a single student. All the files contains information in the same format, as the sample given above. Each section in these files are named within square brackets

Based on the above data, answer the given subquestions.

**Sub questions**

**Question Number : 210 Question Type : MSQ**

**Correct Marks : 3**

Question Label : Multiple Select Question

Consider every command/script below is run in the same directory. Choose the bash script(s) from the options that will give the same output as the given grep command.

```
grep -r -l "^BSCCS30.*\b" | wc -l
```

Note: The usage of options,

- `-r` will recursively go through the files in the current directory and its sub-directories.
- `-l` will print only names of files with a match.

**Options :**

A. ✔

```
count=0
for file in *; do
  while read line; do
    pat="^BSCCS30.*"
    if [[ $line =~ $pat ]]; then
      ((count++))
      break
    fi
  done < $file
done
echo $count
```

B. ✖

```
count=0
for file in *; do
  while read line; do
    pat="^BSCCS30.*"
    if [[ $line =~ $pat ]]; then
      ((count++))
    fi
  done < $file
done
echo $count
```

C. ✖

```
count=0
for file in *; do
  grep "^BSCCS30.*" $file > /dev/null
  if [[ $? == 0 ]]; then
    ((count++))
    break
  fi
done
echo $count
```

D. ✔

```
count=0
for file in *; do
  grep "^BSCCS30.*" $file > /dev/null
  if [[ $? == 0 ]]; then
    ((count++))
  fi
done
echo $count
```

**Question Number : 211 Question Type : MSQ**

**Correct Marks : 3**

Question Label : Multiple Select Question

Consider an associative array `level` (*available as a shell variable*) that contains the current level information of the student in the programme. A student can be currently in any one of the three levels representated by the strings `FOUNDATION`, `DIPLOMA` or `DEGREE`. The key in this array is the student roll number and the corresponding value gives the level the student is in.

We want to add a new line to each student file at the end of `[DETAILS]` section i.e. just before the line containing the text `[COURSES COMPLETED]`. This newly added line should contain the text `LEVEL:<level>` where `<level>` is the current level the student is in.

Which of the following scripts will do the required changes to all the student files.

**Options :**

A. ✔

```
pat="COMPLETED"
for file in *; do
  rollno=`head $file_tmp -n 2 | tail -n 1`
  file_tmp=${file}_tmp

  mv "$file" "$file_tmp"
  while read line; do
    if [[ $line =~ $pat ]]; then
      echo "LEVEL:${level[$rollno]}"
    fi
    echo $line
  done < "$file_tmp" > "$file"
done
```

B. ✖

```
pat="COMPLETED"
for file in *; do
  rollno=`head $file -n 2 | tail -n 1`
  while read line; do
    if [[ $line =~ $pat ]]; then
      echo "LEVEL:${level[$rollno]}"
    fi
    echo $line
  done > "$file"
done
```

C. ✔

```
for file in *; do
  file_tmp=${file}_tmp
  mv "$file" "$file_tmp"

  rollno=`head $file_tmp -n 2 | tail -n 1`
  completed=`grep -n "COMPLETED" $file_tmp | cut -d ":" -f 1`
  total=`wc -l $file_tmp | cut -d " " -f 1`

  head -n $((completed-1)) $file_tmp > $file
  echo "LEVEL:${level[$rollno]}" >> $file
  tail -n $((total-completed+1)) $file_tmp >> $file
done
```

D. ✖

```
for file in *; do
  file_tmp=${file}_tmp
  mv "$file" "$file_tmp"

  rollno=`head $file_tmp -n 2 | tail -n 1`
  completed=`grep -n "COMPLETED" $file_tmp | cut -d ":" -f 1`
  total=`wc -l $file_tmp | cut -d " " -f 1`

  head -n $((completed-1)) $file_tmp > $file
  echo "LEVEL:${level[$rollno]}" > $file
  tail -n $((total)) $file_tmp > $file
done
```

# Sem1 CT

**Number of Questions :** 11

**Section Marks :** 50