B. Sc./HD CIT

ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)
THE ORGANIZATION OF THE ISLAMIC CONFERENCE (OIC)
Department of Computer Science and Information Technology (CIT)

MID SEMESTER EXAMINATION **SUMMER SEMESTER, 2011**
SOLUTION

CIT 4613: Unix Programming

---

1. a) *Name five features of Unix systems that made it popular. Briefly describe two of them in respect of your own experience.* 5+4

You are asked to write the **name** of the features that made Unix "popular". These include the fact that Unix is free, open source, customizable, scalable, portable, secured etc.

In the second part, the important point is you are asked to describe the features in light of your **own experience**. So, giving bookish descriptions or quoting texts from the slides would only be marked partially. On the other hand, if you even write one or two sentences describing your hands-on experience with Solaris or Ubuntu, you are given full marks. Unfortunately, only a few of you answered it correctly.

b) *Define kernel and shell. Describe how the works of Richard Stallman and Linus Torvalds complement each other in building a complete operating system.* 3+5

The standard definitions of kernel and shell will do. Almost all of you got it right.

In the second part, the focus of the question was not only on the works of these two gurus but **how** their works help build Linux. Stallman build the utilities but was lacking a good kernel. Then Torvalds came up with a nicely written kernel which was free. These utilities and kernel together builds up the whole operating systems. Hence, the complement each other.

Moreover, in the first part of the question you were asked about kernel and shell. The second part is only a logical extension of the first part to validate your understanding. You can draw the hint of the answer to the second part from the flow of the entire question.

A few of you gets it right. Others are only given partial mark.

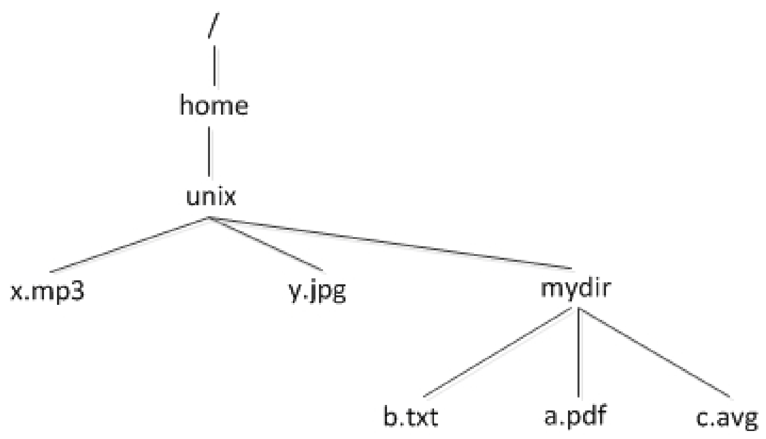
c) *What are CLI and GUI? Give an example task (e.g. copying files, opening programs) each for these two systems that demonstrates their strength.* 2+6

CLI = Command Line Interface/Interpreter.
GUI = Graphical User Interface

The second part of the questions asks you to describe a task which can be efficiently done in CLI and another that can be efficiently done in GUI. The examples are given to you as hints and you cannot just describe them as answers! You could cite new examples or elaborate case studies which clearly shows the point. For example, CLI would be unparallel in copying files with a particular string in their name. CLI would be also efficeint in viewing the processes that are running in the system and changing foreground processes to background. On the other hand, GUI has its own strengths. You can cite examples of drawing software (Photoshop, Gimp) that needs extensive use of mouse. GUI is also useful for novice users and users with disability.

This question carried higher marks to give you a chance to be creatively elaborate. However, only a few could avail of the opportunity. Many of you have described how one can copy files in Unix by writing the **cp** command and in Windows by left click-right click combination. These answers are given partial marks, but ideally they are just wrong answers!

2. a) *The directory structure of a Unix system is shown below. Suppose the present working directory is* ***/home/unix*** 3



Write the output of each of the following commands:

i. ls x*
   Ans: x.mp3
ii. ls 'x*'
   Ans: ls: cannot access x*: No such file or directory
As the input to ls is inside single quotes, * is not expanded and ls tries to list files with the name "x*" but as there is no such files, it displays the error message.
iii. ls */*
   Ans: mydir/a.pdf mydir/b.txt mydir/c.avg
To get full marks, one should notice that ls lists the files alphabetically.

b) *What are absolute and relative paths? If, for the directory structure shown in the figure above, your present working directory is /home/unix, answer the following questions:* 2+2

i. What is the relative path of the home directory? Ans: ../

ii. What is the absolute path of the file a.pdf? Ans: /home/unix/mydir/a.pdf

c) **date** is a Unix command that prints out the current date of the system, as shown below 8

```
$ date
Sun Jul 10 10:24:49 2011
```

Write the output of each of the following commands:

```
  i.   `date`                  : bash: Sun: command not found
 ii.   echo `date`             :  Sun Jul 10 10:24:49 2011
iii.   echo `echo date`        : date
 iv.   echo `echo \`date\``    :  Sun Jul 10 10:24:49 2011
```

d) *For the directory **mydir** shown in the figure above, write whether the following statements are **True** or **False**. Moreover, state the reasoning behind your answer in one sentence.* 10

i. If the permission for mydir is set to r-- , the command **cd mydir** executes successfully.
False. cd needs x permission.

ii. If the permission for mydir is set to -w-, the command **ls mydir** does not execute successfully.
True. ls needs r permission.

iii. If the permission for mydir is set to r-x, the command **cp x.mp3 mydir** executes successfully but **cd mydir** fails.
False. cd does not fail as it has x permission.

iv. If the permission for mydir is set to rw-, the command **ls mydir** executes successfully but **cd mydir** fails.
True. ls needs r, which is given but cd needs x, which is not given.

v. If the permission for mydir is set to -wx, the commands **cp mydir/a.pdf .** and **cp mydir/* .** both execute successfully.
False. The first cp works but the second one fails as there is no r permission. No one gets this one totally right.

No mark is given for only writing True/False. Partial mark is given in most cases, based on the display of your understanding.

3. a) *Which one is more powerful between **sed** and **awk**? Why?* 5

As the lecture slides said, sed is more powerful. The reasons include speed, functionality, library functions, ability to do field specific operations etc.
Most of you got it right.
However, sed may excel in some especial cases as substitution is one of its strongholds. If you could reason enough to show sed is more powerful, you would be given full marks.

b) *Write short note on each of the following topics with respect to Regular Expressions. Provide appropriate examples.* 3x3

i. Anchors
ii. Backreference
iii. Greedy and lazy repetition

Typical definitions from the text books should suffice. Examples carry 1 mark in each part.

c) *Write an awk file that takes a text file as argument and prints each of the words in the text file along with the number of times they occur in that file. A sample run of the file is shown below:* 6

```
$ cat test.txt
A quick brown fox jumps over and over
$ awk -f wordfreq.awk test.txt
fox 1
A 1
quick 1
over 2
brown 1
jumps 1
and 1
```

The script uses associative arrays that we have discussed in the lectures. Many of you got it right. Some of you had problems with printing the results, some of you failed to iterate over the array. Generous partial marks are given.

A probable solution:

```
{ for (i = 1; i <= NF; i++)
        num[$i]++
}
END{
         for (word in num)
                   print word, num[word]
}
```

d) *Using the* ***wordfreq.awk*** *file and Unix filters in a pipeline, write a shell command that prints the 10 most frequent words of a text file in decreasing order.* 5

You have to sort the output of wordfreq with respect to the second column, numerically and in decreasing order. Then you have to take the top 10 values.

```
$  awk -f wordfreq.awk test.txt | sort -nrk2 | head -10
```

```
$  awk -f wordfreq.awk test.txt | sort +1 -nr | head -10
```

If you look at the question carefully, you were asked to print only “the words” but not the occurrences. Thus, the above mentioned solutions are not fully correct as they print both the word and the corresponding frequency. Though you are given full marks for this type of answers, the correct command would be -

$ **awk -f wordfreq.awk test.txt | sort -nrk2 | head -10 | cut -d” “ -f1**

4. a) *Demonstrate how one can redirect both error and standard output to a single file.* 5

An example showing any one of the two methods gives you full marks. Writing both of the methods does not earn any extra marks though! ☺

b) *What are the meanings of **$$, $?** and **$@** in a shell script? Briefly describe the three ways one can use to return a value from a function in shell script.* 3+3

$$ is id of the Shell, $? Is the exit value of the last command, $@ is the array holding all the command line arguments.

The three ways are - echo, global variables and return.

c) *Write a shell script that reads a number from the user and prints out the factorial of that number.* 7

A probable solution:

```
#!/bin/bash
read n
fact=1
if [ n -eq 0]
        fact=1
else
   while [ $n -ge 1 ]
   do
        fact=`expr $fact \* $n`
        n=`expr $n - 1`
   done
fi
echo $fact
```

d) *Write a shell script that takes a directory path from the user and list all the files in that directory to which you have read, write and execute permissions.* 7

The important point is you are asked to list only the “files”, not the directories. This can be done in many ways - using grep’s filtering ability or using the find command with -perm option, or using the Unix file permission tests.

A probable solution:

```
#!/bin/bash
prsnt="$PWD"
cd "$1"
for file in *
do
if [ ! -d $file ]
then
        if [ -r $file ]
        then
                if [ -w $file ]
                then
                        if [ -x $file ]
                        then
                                ls -l $file
                        fi
                fi
        fi

fi
done
cd "$prsnt"
```

Sketches of some other solutions:

```
$find "$1"-type d -perm 777 -exec ls -l '{}'
$ ls –l | grep ‘^-rwx(.)*’
```