SENECA COLLEGE OF APPLIED ARTS & TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATIONS TECHNOLOGY
DPS918 - UNIX Bash Shell Scripting
Winter 2017 Term Test V1

Name: Elvis Tran

Student Number: 065743114

Time Allowed: 90 minutes

Professor: Les Czegel

Student's Mark: 25 out of 60

Please Note: It is your responsibility to ensure that your test paper has all 9 pages. Check this now.

Please answer all questions on the test paper itself. If you need more room, use the back of the previous page.

Part 1 (10 marks) Each correct answer: 1 mark.

Circle the letter corresponding to the one correct answer to each of the following questions.

1. With a umask of 026, what will be the default permissions for newly-created ordinary files?

   a. r w x r - x - - x
   b. r w - r - - - - -
   c. - - - - w - r w -
   d. r w x r - - - - -

2. Which filename **will** be matched by the following extended glob:
   [!a-c]*(cat|dog)?ab

   a. qdogcataab
   b. ab
   c. ipdogaaab
   d. tcatdogab

3. The variable containing the exit status of the previous command is:

   a. $0
   b. $*
   c. $#
   d. $?

4. Failure of a command is usually indicated by:

   a. an exit status of 0
   b. a non-zero exit status
   c. an error message redirected to standard output
   d. getting results different than expected

5. If script "zoo" is called as follows: **zoo lion tiger bear**
   and "zoo" executes the following: **set 24 375 ; echo $1 $2 $3**
   what will be displayed?

   a. 24 375
   b. lion tiger bear
   c. 24 375 bear
   d. lion 24 375

6. One method to read "file1" one line at a time in a loop is:

   a. `read line < file1 | while`
   b. `for line in file1`
   c. `cat file1 | while read line`
   d. `while read line < file1`

Infinite loop

7. You need a file called "test2" which duplicates the contents of "test1", without deleting "test1". Which command **cannot** be used to meet this requirement?

   a. mv
   b. cat
   c. cp
   d. grep

8. What will this command do? **`cat < xfile | grep xfile`**

   a. pipes the standard input of cat to the standard output of grep
   b. pipes the standard input of cat to the standard input of grep
   c. pipes the standard output of cat to the standard input of grep
   d. pipes the standard output of cat to the standard output of grep

9. Which line will **not** be matched by the following extended regular expression: **`^[^a-c]*(cat|dog)?a+b$`**

   a. ab
   b. ipdogaaab
   c. qdogcataab
   d. tcatdogab

The number is take away from (take nothing from users, take away W from group, take away WX from other)

10. With a umask of 023, a directory called "testing" was created. Which command will ensure that **all** users can copy files to this directory?

   a. `chmod u+wx,g+x testing`
   b. `chmod o+w,go+x testing`
   c. `chmod go+w,o+x testing`
   d. `chmod g+wx,u+x testing`

Part 2 (10 marks) Each correct answer: 1 mark.

Match the ***Symbol/Command*** number beside the best ***Explanation***.

Matched Number	Explanation	Symbol / Command
(11)	**(EXAMPLE)** Input redirection.	1. $0
(17)	All script arguments as a single string.	2. 2>
(~~7~~ 15)	Command to assign files as file descriptors.	3. which
(20)	Current directory.	4. &&
(14)	Execute a command if the previous one failed.	5. find
(~~13~~ 3)	Command to display path of an executable.	6. <Ctrl>-d
(10)	Command to convert piped input to parameters.	7. >&2
(~~3~~ 1)	Name by which shell script was called.	8. $2
(~~19~~ 6)	End-of-file for STDIN.	9. ..
(~~6~~ 18)	Terminate a long-running command.	10. xargs
(~~2~~ 7)	Redirect command output to STDERR.	11. <
		12. file
		13. whereis
		14. ||
		15. exec
		16. $$
		17. $*
		18. <Ctrl>-c
		19. <Ctrl>-z
		20. .

4

Part 3 (10 marks) Each correct answer: 1 mark.

Write the shortest single command, including any required options and pathnames, to answer the following questions. Do not assume a specific current directory.

1. Write a command to delete a directory called "dir1" from the current directory, including any contents, adding any error messages to the end of file "messages" in your home directory:

rm -r dir1 2>> messages

rm -r dir1 2>> ~/messages

append

0.5

2. Write a command to list all filenames in the current directory that consist of an 'a' followed by one or more digits, assuming extended globbing is enabled:

ls -d a+([0-9])

0

3. Write a command to allow users in your group to add or delete files in the parent of your current directory, without changing any other permissions:

chmod g+wx ..

chmod g+wx ..

0

4. Write a command to place the PID (process ID number) of the current process into the shell variable "$1":

set $$

1

5. Write a command to subtract 2 from the first argument passed to the script, and place the result into a variable called "number":

((number=$1-2))

number = $(($1-2))

0.5

6. Write a command which will display **only** the 7th line in a file called "stock" in the current directory:

sed -n '7 p' stock

awk "NR==7" stock

1

7. Write a command to display **only** lines containing at least 1000 characters, in a file called "xfiles" in the current directory:

~~wc -c 1000 xfiles~~

egrep ".{1000}" xfiles

{1000} ← duplication

1 of any character

0

8. Write a command to display character positions 10 to 20 in every line of a file called "personnel" in the current directory:

~~sed~~ cut -c 10-20 personnel

0

9. Write a command to display "Enter your name" with no new-line on the terminal, regardless of any redirections specified when calling the containing script:

echo -n "Enter your name" > /dev/tty

0

10. Write a command to execute file "a.out" in the current directory, sending output to a file called "total.tax" in the "/tmp" directory:

a.out 1> /tmp/total.tax

a.out > /tmp/total.tax

./a.out ← For not in your Path

1

1.5

Part 4 (10 marks) Each correct answer: 2 marks.

Write the shortest single UNIX command line required to perform each of the following tasks, without using the semi-colon (;) command separator. Do not assume a specific current directory. Efficiency counts, the fewer and simpler the commands and arguments, the better:

1. Here is a sample line from the output of an "ls -al" command:

   -rw-r--r-- 1 jbloe users 1878 Mar 16 20:21 index.html

   Display this information for the largest file in your current directory.

   ls -al | sort -nk 5 | tail -1

   ls -al5 | head -1

2. If the first script argument is a single-digit number, display that field number within the tab-delimited file "file2". For example, if the first argument is "7", then the 7th field in each record of "file2" should be displayed. For full marks, do not use "grep" and use a named class.

   ~~sed -n '$1 p' file2~~ [[$1 = [[:digit:]]]] &&

   cut -f $1 file2

3. Display the contents of **all** files in the current directory that have names ending with ".txt", converting all alphabetic characters to uppercase.

   ~~ls -al *.txt~~ | tr '[a-z]' '[A-Z]'

   cat *.txt .*.txt | tr '[a-z]' '[A-Z]'

   ↖ will get hidden files

4. Display the permissions for an ordinary file called "file1" in the current directory, in the following (example) format:
   r w x r - x r - - (note the blanks between the characters).

   Same character ↓

   ls -l file1

   ls -l file1 | cut -c2 -10 | sed 's/./& /g'

5. Assign the number of fields in the file called "marks.dat" to the variable "fields". The number of fields is the same within each line of "marks.dat" and the fields are delimited by a space. The "marks.dat" file is contained in your current directory.

   fields=$(head -1 marks.dat | wc -w)

   ~~fields=$(awk 'NR==1~~

6

Part 5 (10 marks)

My attempt at writing a script is shown below. It doesn't seem to work. Place an "X" beside each incorrect or missing line, and show the corrections required. There are several errors, please identify and correct any 5 of them.

```
if [ $# = 1 ]

   then echo $1 > file1

        if cat file1 | egrep '^[0-9][0-9]*$' file1 > /dev/null

           then set $1 $(whoami)

           else set 25 $1

        fi
elif [ $# > 2 ]
   then "Usage: greatscript [ number ] [ userid ] " >&2
elif [ $# = 0 ]

   then set 25 sysadmin

else if echo $2 | egrep '^[0-9][0-9]*$' > /dev/null

        set $2 $1

   fi

echo "The answer is $2" | mail $1
```

or remove cat file1

remove file1

remove an extra ( )

(( )) = arithmetic
( ) = substitution

[$# -gt 2]

missing echo

$number

$user'id

exit 1 is another error

missing then

else if fi needs an if you keep it else if

for main if

if
elif
fi

if
else if
fi
fi

5.5

Part 6 (10 marks)

Write a script called "numdir" that accepts any number of arguments. Each argument will be a filename (with any required path), which may be a directory, another type of file, or a filename that doesn't exist.

Display an error message for each filename that doesn't exist. Display a message for each existing filename that is not a directory. Then display the number of directories.

Here is an example of an execution, assuming that the current directory contains "f1" and "f2" as ordinary files, "d1" is a directory, and "xxx" and "yyy" don't exist. Follow the output format shown:

```
$ ls -F
d1/   f1   f2
$ numdir f1 xxx d1 f2 yyy
Filename f1 is not a directory
Filename xxx does not exist
Filename f2 is not a directory
Filename yyy does not exist
Directories: 1
$
```

Efficiency counts, the simpler and shorter the script and each command in it, the better. The script will be marked for readability, including indentation and meaningful variable names. Comments are not necessary. You don't need to check for errors other than those specified. **Please read the question and the example output carefully.**

SOLUTION:

-le or [[$Count <= $#]]

```
directories=0
Count=1
while [$Count -le $#]
do
    if [ ls -l $(Count) | grep "^[d]" ]
    then
        ((directories=$directories+1))
    elif [ ls -l $(Count) | grep "^[-]" ]
    then
        echo "Filename $(Count) is not a directory"
    else
        echo "Filename $(Count) does not exist" >&2
    ((Count=$Count+1))
done
```

1 2 2 2 2 -1 -½ -½ -½ -½