To be completed by student – PLEASE PRINT CLEARLY

Name: **MUHAMMAD IBADRURRAHMAN BIN MOHAMAD HAFIRIZUL**		
ID Number: **AM2207011624**		
Lecturer **NURSHAFINAS BINTI ROSLAN**		Lab group / Tutorial group / Tutor (if applicable)
Course and Course Code **ARC3043**		Submission Date: 9/2/2024
Assignment No. / Title **ASSIGNMENT WRITE BASH SCRIPTING**		Extension & Late submission: Allowed / Disallowed
Assignment type: **INDIVIDUAL**	% of Assignment Mark	Returning Date:

Penalties:

1. 10% of the original mark will be deducted for every one-week period after the submission date
2. No work will be accepted after two weeks of the deadline
3. If you were unable to submit the coursework on time due to extenuating circumstances you may be eligible for an extension
4. Extension will not exceed one week

Declaration: I/we the undersigned confirm that I/we have read and agree to abide by these regulations on plagiarism and cheating. I/we confirm that this piece of work is my/our own. I/we consent to appropriate storage of our work for checking to ensure that there is no plagiarism/ academic cheating.

Signature: *aiman*

Full Name: **MUHAMMAD IBADRURRAHMAN BIN MOHAMAD HAFIRIZUL**

Question 1

a. Write a bash script that prints tomorrow's date. (4 Marks)

```
aiman04@LAPTOP-KHOK30OL:~$ nano tomorrow_date.sh
aiman04@LAPTOP-KHOK30OL:~$ chmod +x  tomorrow_date.sh
aiman04@LAPTOP-KHOK30OL:~$ ./tomorrow_date.sh
Today's date is 2024-02-09
Tomorrow's date will be 2024-02-10
aiman04@LAPTOP-KHOK30OL:~$
```

Bash script file named tomorrow_date.sh

```
#!/bin/bash

TODAY=$(date +%Y-%m-%d)

TOMORROW=$(date  -d "$TODAY +1 day" +%Y-%m-%d)

echo "Today's date is $TODAY"

echo "Tomorrow's date will be $TOMORROW"
```

b. Create a Bash Script that print out message based upon which day of the week it is

in Spanish. (4 Marks)

```
aiman04@LAPTOP-KHOK30OL:~$ nano hari_dalam_minggu.sh
aiman04@LAPTOP-KHOK30OL:~$ ./hari_dalam_minggu.sh
-bash: ./hari_dalam_minggu.sh: Permission denied
aiman04@LAPTOP-KHOK30OL:~$ chmod +x hari_dalam_minggu.sh
aiman04@LAPTOP-KHOK30OL:~$ ./hari_dalam_minggu.sh
Hoy es lunes. ¡Buena semana!
aiman04@LAPTOP-KHOK30OL:~$
```

```
#!/bin/bash

day=$(date +%u)

if [ "$day" -eq 1 ]; then
        echo "Hoy es lunes. ¡Buena semana!"  # Today is Monday. Have a good week!

elif [ "$day" -eq 2 ]; then
        echo "Hoy es martes. ¡Sigue adelante!"  # Today is Tuesday. Keep going!

elif [ "$day" -eq 3 ]; then
        echo "Hoy es miércoles. ¡Mitad de semana!"  # Today is Wednesday. Halfway through the week!

elif [ "$day" -eq 4 ]; then
        echo "Hoy es jueves. ¡Casi al final!"  # Today is Thursday. Almost there!

elif [ "$day" -eq 5 ]; then
        echo "Hoy es viernes. ¡Finalmente viernes!"  # Today is Friday. Finally Friday!

elif [ "$day" -eq 6 ]; then
        echo "Hoy es sábado. ¡Disfruta el fin de semana!"  # Today is Saturday. Enjoy the weekend!

elif [ "$day" -eq 7 ]; then
        echo "Hoy es domingo. ¡Relájate y descansa!"  # Today is Sunday. Relax and rest!

else
        echo "Error: Something went wrong."  # Error message if the day is not recognized
fi
```

c. Write a bash script that tells the user if the user entered a value greater than 100, the system will echo “That is a very large number” and if the number is even, then it will echo “This is also even number”. (If Statement)

```
aiman04@LAPTOP-KHOK30OL:~$ nano identify_category_numbers.sh
aiman04@LAPTOP-KHOK30OL:~$ chmod +x identify_category_numbers.sh
aiman04@LAPTOP-KHOK30OL:~$ ./identify_category_numbers.sh
Enter a number: 2
This is also an even number.
aiman04@LAPTOP-KHOK30OL:~$ ./identify_category_numbers.sh
Enter a number: 5
aiman04@LAPTOP-KHOK30OL:~$ ./identify_category_numbers.sh
Enter a number: 101
That is a very large number.
aiman04@LAPTOP-KHOK30OL:~$
```

```bash
#!/bin/bash

read -p "Enter a number: " num

# Check if the entered number is greater than 100
if [ "$num" -gt 100 ]; then
    echo "That is a very large number."

elif [ $((num % 2)) -eq 0 ]; then

# Check if the entered number is even
    echo "This is also an even number."
fi
```

Question 2

a. Using sort as a filter, rewrite the following sequence of commands: (4 Marks)

$ sort list > temp

$ lpr temp

$ rm temp

b. Assume that the following files are in the working directory:

$ ls

intro notesb ref2 section1 section3

section4b notesa ref1 ref3 section2

section4a sentrev

Give commands for each of the following, using wildcards to express filenames with

as few characters as possible.

i. List all files that begin with section.

```
aiman04@LAPTOP-KHOK30OL:~$ ls section*
section1  section2  section3  section4a  section4b
aiman04@LAPTOP-KHOK30OL:~$
```

ii. List the section1, section2, and section3 files only.

```
aiman04@LAPTOP-KHOK30OL:~$ ls section[1-3]
section1  section2  section3
aiman04@LAPTOP-KHOK30OL:~$
```

iii. List the intro file only.

```
aiman04@LAPTOP-KHOK30OL:~$ ls intro
intro
aiman04@LAPTOP-KHOK30OL:~$
```

iv. List the section1, section3, ref1, and ref3 files. (8 Marks)

```
aiman04@LAPTOP-KHOK30OL:~$ ls section[13] ref[13]
ref1  ref3  section1  section3
aiman04@LAPTOP-KHOK30OL:~$
```

c. Redirect standard output from a sort command to a file named phone_list. Assume the input file is named numbers.

```
aiman04@LAPTOP-KHOK30OL:~$ touch phone_list numbers
aiman04@LAPTOP-KHOK30OL:~$ nano numbers
aiman04@LAPTOP-KHOK30OL:~$ sort numbers > phone_list
aiman04@LAPTOP-KHOK30OL:~$ cat phone_list
1
2
3
4
5
6
aiman04@LAPTOP-KHOK30OL:~$
```

Question 3

a. Create a Bash Script that check whether a directory named Bash Scripting existed or not. If exists, it will echo “Directory Exists”, if not, it will echo “Directory is not existed”. Show the appropriate permission set to run the file (10 marks)

```
aiman04@LAPTOP-KHOK30OL:~$ mkdir "Bash Scripting"
aiman04@LAPTOP-KHOK30OL:~$
```

```
aiman04@LAPTOP-KHOK30OL:~$ nano check_whether_a_directory_Bash_Scripting_existed.sh
aiman04@LAPTOP-KHOK30OL:~$ chmod +x  check_whether_a_directory_Bash_Scripting_existed.sh
aiman04@LAPTOP-KHOK30OL:~$ ./check_whether_a_directory_Bash_Scripting_existed.sh
Directory Exists
aiman04@LAPTOP-KHOK30OL:~$
```

```
#!/bin/bash

directory="Bash Scripting"

if [ -d "$directory" ]; then
     echo "Directory Exists"
else
     echo "Directory is not existed"
fi
```

```
aiman04@LAPTOP-KHOK30OL:~$ nano check_whether_a_directory_Bash_Scripting_existed.sh
aiman04@LAPTOP-KHOK30OL:~$ chmod +x  check_whether_a_directory_Bash_Scripting_existed.sh
aiman04@LAPTOP-KHOK30OL:~$ ./check_whether_a_directory_Bash_Scripting_existed.sh
Directory is not existed
aiman04@LAPTOP-KHOK30OL:~$
```

```
#!/bin/bash

directory="Bash Scriptin"

if [ -d "$directory" ]; then
     echo "Directory Exists"
else
     echo "Directory is not existed"
fi
```

b. Create a Bash Script to Check How Long the High CPU Consumption Processes

Runs on Linux. Show the appropriate permission set to run the file. (10 Marks)

```
aiman04@LAPTOP-KHOK30OL:~$ nano Check_How_Long_the_High_CPU_Consumption.sh
aiman04@LAPTOP-KHOK30OL:~$ ./Check_How_Long_the_High_CPU_Consumption.sh
High CPU Processes:
User     PID  Running Time
----     ---  --------------
root      1     06:13:36
root      13    06:13:33
root      14    06:13:33
aiman04   15    06:13:33
aiman04   1675  00:00:00
```

```
#!/bin/bash

# Set CPU usage threshold
threshold=50

# Get top 5 CPU process IDs
procs=$(ps aux --sort=%cpu | awk 'NR<=6{print $2}' | tail -n +2)

# Check running time
echo "High CPU Processes:"

# Print header
echo "User     PID  Running Time"
echo "----     ---  --------------"

# Loop through processes and print table
for pid in $procs; do
   etime=$(ps -p $pid -o etimes=)
   user=$(ps -p $pid -o user=)

   # Convert elapsed time to HH:MM:SS format
   hours=$(( etime / 3600 ))
   minutes=$(( (etime % 3600) / 60 ))
   seconds=$(( etime % 60 ))

   # Print table row
   printf "%-4s %-5d %02d:%02d:%02d\n" "$user" "$pid" "$hours" "$minutes" "$seconds"
done | column -t
```