Task 1. bash script named `script1` The name of your bash script must be **script1** and below is a description of what it should do when executed.

1. The script displays a two-line message. The first line is `Please enter three-letter code of the day of the week` and the second line is `examples: Mon, Tue, ..., Sun`
2. The script reads the user's response into a variable
3. Then it displays a message `++++++++++++++++`
4. Using command `date`, it obtains the current day of the week and stores it in a variable.
5. Then it compares the day entered by the user and the day obtained from `date`. If they are the same, it displays a two-line message: the first line is `Good answer` and the second line is `The day of the week is X` where `X` is the current day of the week, and terminates.
   If they differ, it displays a two-line message: the first line is `Bad answer` and the second line is `your answer: X, real day: Y` where `X` is the day entered by the user and `Y` the day obtained from `date`, and the script terminates.

A few useful hints:

- *What commands you might need:* `echo` , `date` *and* ***if statement****.*
- *If a variable* `xxx` *contains a string, then the content of the variable is accessed using* `$xxx` *. If we only want a certain part of the string -- this is called a* ***substring*** *-- we can access it using* `${xxx:P:L}` *where* `P` *is the starting position (a number from 0 to n-1 where n is the length of the string) and* `L` *is the length of the substring we want.*
  For instance, let `xxx="helloWorld"` *, then* `${xxx:0:3}` *is* hel,
  while `${xxx:2:5}` is `lloWo` .
- *The comparison of two variables containing strings* `xxx` *and* `yyy` *can be executed as*
- `if [ $xxx == $yyy ]`
- `then`
- `    ....`
- `else`
- `    ....`
  `fi`

A sample run:

```
Please enter three-letter code of the day of the week
examples: Mon, Tue, ..., Sun
Mon
++++++++++++++++
Bad answer
your answer: Mon, real day: Sun
```

A sample run:

```
Please enter three-letter code of the day of the week
examples: Mon, Tue, ..., Sun
Sun
++++++++++++++++
```

```
Good answer
The day of the week is Sun
```

Task 2. bash script named `script2` The name of your bash script must be **script2** and below is a description of what it should do when executed.

1. The script creates in the current directory a directory named `DIR1` and displays a message `DIR1 created`
2. Then the script creates in the directory `DIR1` a subdirectory named `DIR2` and displays a message `DIR1/DIR2 created`
3. In the directory `DIR2` , the scripts creates a file named `X` containing one line saying `I am file X`
4. Then it displays a message `contents of DIR2` and shows the contents of `DIR2`
5. Then it displays a message `contents of DIR1` and shows the contents of `DIR1`
6. Then it displays a message `contents of current directory` and displays the contents of the current directory
7. Then it moves the file `X` to current directory from `DIR`
8. Then it tries to remove `DIR1` using `rmdir` (it will not work)
9. Then it tries to remove `DIR1` using `rm` (it will not work)
10. Then it tries to remove `DIR1` using `rm -r` (and it should work)
11. Then it shows the contents of the current directory.
12. The current directory should contain a file named `X` containing one line `I am file X`

A few useful hints:

- *What commands and concepts you might need*: `cd mkdir rm echo ls mv cat`
- *Current directory is referred to as* `.` *, the parent directory as* `..`
  For instance, `ls .` *will show all files/subdirectories in the current directory, while* `ls ..` *will show all files/subdirectories in the parent directory.*
- *To terminate execution of a script, you can use the* `exit` *command.*

Sample run: executing `script2`

```
DIR1 created
DIR1/DIR2 created
contents of DIR2
X
contents of DIR1
DIR2
contents of current directory
DIR1  script1  script2  X
rmdir: DIR1: Directory not empty
rm: cannot remove `DIR1': Is a directory
script1  script2  X
```

And there is a file `X` in the current directory containing one line `I am file X`