CS108 - Software Systems Lab

Lab 7 - Shell Scripting

Student: Aditya Sanapala, 23b0912@iitb.ac.in
Lecturer: Kameswari Chebrolu, chebrolu@cse.iitb.ac.in

Problem 1: Sorting Mail IDs

In this activity write a bash script with the name `getEmails.sh`, which satisfies the following conditions:

1. It should accept a single argument representing a `.txt` file. Otherwise it should show `Usage: ./getEmails.sh <file>`. Meaning if `./getEmails.sh <arg1> <arg2>` is used then it should show the error message.
2. Check if the argument file exists or not. If not, show the message `Input File doesn't exist`.
3. When executed, It should create 3 files - `emails.txt`, `sortedEmails.txt`, `cseEmails.txt`.
4. The bash script should extract all the emails of format `<alpha_num>@<alpha>.iitb.ac.in` and store them in `emails.txt`. Here `<alpha_num>` consists of alphabets and digits and `<alpha>` means only alphabets. Assume the `input.txt` file has one email address per line, and only email addresses will be present in those lines.
5. Next it should sort all the emails based on descending order with case-insensitivity and store the results in `sortedEmails.txt`. The sorting should be based on lexicographical order.
6. Next it should extract all the CSE department emails from sorted emails and store the results in `cseEmails.txt`. The CSE department emails are of format `<alpha_num>@cse.iitb.ac.in`.

Problem 2: Appendages

Write a bash program to append a string (given as a command line argument) to all files in the same directory as the bash script having the extension `.out`.

Usage: `bash submission.sh <string>`

Command: `bash submission.sh Hello World 123 @`

On executing this command, the string `Hello World 123 @` will be appended to every file in the directory having the extension `.out`. All other files should remain untouched.

Example: If there is a file in the directory `a.out` containing data `Hello CS108 Students!`

then after executing the script the data contained in `a.out` would become `Hello CS108 Students!Hello World 123 @`

Though only one file (`a.out`) has been supplied to the GUI, feel free to create more files for testing using the `touch` command from the terminal and populate them with data by piping the output of the `echo` command. The contents can be viewed using the `cat` command.

Problem 3: Let's rename some files!

Write a bash program to rename all `.pdf` files in the current directory in the form of `rollno.pdf` to `rollno_name.pdf`. The mapping between roll numbers and names is provided in a file, the address of which is provided as a command line argument. Assume all names are a single word.

Mapping File Format: `<rollno> <name>`

Example:
```
200050049 Harsh
200050100 Parth
200050129 Sarthak
... (continued)
```

Usage: `bash submission.sh <path_to_mapping_file>`
Example: If we use the above mapping file, the file `200050100.pdf` should be renamed to `200050100_Parth.pdf`. If for some file, the mapping does not exist, that file should remain untouched.

The directory has an editable mapping file, along with the `submission.sh` file. Feel free to create more files for testing using `touch`, and see the change in their names on running the script by using the `ls` command.

For reading from a file, you can use the following piece of code:
```
while read -r line
do
arr=($line)
done < a.txt
```

This piece of code reads from the file `a.txt` line by line. The line is broken up into tokens using spaces as a separator and stored as an array in `arr`. Now, the words can be indexed as in a normal array.

Problem 4: Data Segregator and Grader

You have been given a `.csv` file called `grades.csv` which consists of comma separated columns `rollno,quiz1,quiz2,midsem,endsem,total-marks`. The column names should be self-explanatory for this `.csv` file.

The first line of the `.csv` file contains header in the sequence described above, and from line 2 onwards the `.csv` file contains actual data within their respective column.

The `.csv` file contains all the students from B.Tech. batch of 2023 (roll numbers starting from 23B001 to 23B999) and 2024 (roll numbers starting from 24B001 to 24B999) in an unordered manner.

Your task is to create a bash script in `submission.sh` file where below conditions are fulfilled:

1. Segregate the data based on if a row contains student of batch 2023 or 2024. All the data of batch 2023 should be written in a separate file called `ug23.csv`, and the same way all the data of batch 2024 should be written in a separate file called `ug24.csv`.
2. You also have to add a new `grades` column in these data with following matrix:

   ```
   | marks | grades |
   | >85 | AA |
   | >65 and <=85 | AB |
   | >45 and <= 65 | BB |
   | >35 and <= 45 | CC |
   | <=35 | F |
   ```

3. The final `.csv` file must be sorted by grades column (from AA to F) and if 2 students have same grades than then are sorted in non-descending order of the roll number.

The filename must be taken via command-line argument. Ideally the execution of bash script must be done with the command `bash submission.sh grades.csv`.

Note: You can assume that the total-marks add up using `quiz1, quiz2, midsem,` and `endsem` marks. You can also assume that total-marks in range [0, 100].

Extra: You are also given you some extra testcases in `/home/labDirectory/testcases` folder for you to refer.

Example:
`grades.csv`:

```
rollno,quiz1,quiz2,midsem,endsem,total-marks
23B001,10,8,25,45,88
23B010,9,8,25,40,82
23B009,5,0,25,45,75
23B002,10,5,25,5,45
24B001,10,8,25,45,88
24B010,3,2,10,5,20
24B009,5,10,25,45,85
24B002,10,5,25,25,65
```

`ug23.csv`:

```
rollno,name,quiz1,quiz2,midsem,endsem,total-marks,grades
23B001,10,8,25,45,88,AA
23B009,5,0,25,45,75,AB
23B010,9,8,25,40,82,AB
23B002,10,5,25,5,45,CC

ug24.csv:
rollno,quiz1,quiz2,midsem,endsem,total-marks,grades
24B001,10,8,25,45,88,AA
24B009,5,10,25,45,85,AB
24B002,10,5,25,25,65,BB
24B010,3,2,10,5,20,F
```