CS 108 Project - Bash Grader

Navya Garg

April 26, 2024

Contents

0.1 Objective

The bash script file `submission.sh` implements a csv file manager and interpreter for analysing various examination data for a group of students in different exams. It also implements a version of git (Version Control System). Few other files add some customisations to it.

0.2 Task of the Bash Grader

There are two parts of the project:

- Suppose we are provided with a large number of such files. Each such file has data for a student for a particular exam. This grader combines all those files at one place and finds the total for each student.
- It implements some basic git commands such as commit and checkout, to maintain various versions of the csv files at different instances.

0.3 Working of the `combine` command

The most important part of compiling the files together is done by the combine function.
By iterating line by line over all the csv files in the working directory, it stores the roll numbers and names uniquely in a dictionary `main_array`. Then again, while iterating over all the files, it creates an `exam_field` array, which it attaches to the roll number and name fields using the Unix's `paste` command.[5]

0.4 Working of the code

The bash script calls different functions (or files in some cases) upon invocation of different commands. There is a bash file `submission.sh`, some awk files and some python files, which together carry out the tasks in a time-efficient manner. Apart from combining the files, various statistics are provided using `numpy` and `matplotlib` python libraries. There is a GUI interface included for major commands.

0.5 Usage

The usage of the grader is simple. Copy the Grader files in the working directory where the csv files for all the exams are present. Now, run on the terminal:
`bash submission.sh <command> <arguments(if needed)>`

`combine`: This command calls the **combine** function of the submission.sh file, which compiles all the csv files present in the working directory into one file `main.csv`.

`upload <path/to/the/file>`: This command upoads an external csv file to the working directory from the path provided as CLI argument.

`total`: This command adds a column for total marks of each student in `main.csv`. It calls the **total** function, which in turn runs the **total.awk** file and appends the total at the end of each record in `main.csv`.

`git_init <path/to/the/remote/repository>`: This command creates a folder at the given path, which acts as the remote repository for the working directory.

`git_commit -m <commit_message>`: This copies the current version of all the files in the current directory to the remote repository and prints all the files that have been modified at this commit. Each commit generates a unique 16 digit hash value[1] used to identify it.

`git_checkout`: This lets us checkout to some older commit we made.There are two ways of doing this:

- Using Hash value
- Using commit message

`update`: This lets the user update records for a particular student in the database.

0.6 Customisations

There are a number of customisations included:

- Two more git commands have been added, namely:

  `git_log`: It maintains and displays the commit history of the directory.

  `git_main`: The checkout function stores the latest work in the working directory, which has not been committed yet while you checkout to a previous commit. Running this command restores the directory back to the latest modifications.

- **Extended `update` command functionality:** If the roll number entered doesn't exist already in the database, it will allow you to add a new record with that roll number.
- The `update` also does a lot of case handling, in case the user is not acquainted with the database.
- Graphs and various measures of central tendency have been added to help analyze the data better, which includes scatter plots of marks of students for different exams, student performance summary and student comparison with the highest marks in a particular exam. This is governed by `stats.awk, plot.py, Get_record.py` files and `stats` function of submission.sh, which are invoked mainly by these commands:

  `grading_stats`: This command displays the Mean, Median and Standard Deviation of marks for a particular exam and also plots a scatter plot for the marks of all students in that exam.

  `overall_stats`: This provides the statistics for total marks of all students in all the exams combined.

- **Extract records for a particular student:** Using the `get_record` command, the performance of a particular student can be summarised. Also his/her comparison is provided with the highest marks in every exam and a graph is plotted.
- The Roll Numbers in the main.csv file are sorted by implementing a **merge-sort** algorithm.
- The terminal commands are made less specific by incorporating case insensitivity while taking inputs in various functions.[2]

0.6.1 GUI interface

A GUI interface has been implemented using the `tkinter` python module.[3][4] It can be initialized through the command:
`bash submission.sh start_gui`
There are buttons for most of the commands, which on the back-end, call corresponding functions in the `GUI.py` file, which in turn invoke the bash or the python script to carry out the task. This was achieved by using the `subprocess` python module.

It lets us view grading statistics for each exam and overall grading statistics for the Course. The main window displays graph for the marks scored by all students in total.

Separate buttons for different exams pop up new windows which display the respective Mean, Median and Standard Deviation along with providing graphical analysis.

On clicking the **Git Version Control** button in the main window, a new window pops up where you can use the git functionality through a GUI.

An option for **Student Performance** in the main window pops up to ask for the roll number, and then displays the performance summary of that student in different exams.

0.7 Precautions

The python libraries `numpy, tkinter, subprocess, re, os, matplotlib, pandas, sys` must be installed. Certain utilities may not work properly without the proper compilation of the main.csv using `combine` command.

Bibliography

[1] https://ioflood.com/blog/bash-random-number/.

[2] https://earthly.dev/blog/bash-string/.

[3] https://www.javatpoint.com/python-tkinter-button.

[4] https://www.tutorialsteacher.com/python/create-gui-using-tkinter-python.

[5] Kameshwari Chebrolu. *CS 108 Slides.* 2024.