CS 108 Project - Bash Grader

Sabil Ahmad

April 27, 2024

Contents

1 Introduction

The "Bash Grader" project for CS 108 aims to develop a robust CSV file manager and interpreter using Bash scripting. This tool facilitates handling student exam data stored in CSV files, allowing for seamless merging, updating, and analysis of student performance across multiple exams. Additionally, it incorporates a simplified version of Git for version control and introduces customizable features for enhanced data visualization and management.

To view the script in action go to this video I recorded:-
`https://drive.google.com/drive/folders/1lRMjopEklaMljrv0PdsfLnWo1TsdEORT`

2 Basic File Handling Functionalities

2.1 Combine

```
bash submission.sh combine
```

On writing the above command the bash script iterates through all the *.csv* files present in the current directory aiming to create a main.csv file , firstly it stores all the .csv files in a an *exam_files* array .Now, from all the files in the array it goes and gets the union of all the students and adds it to the main.csv file using the *get_students* function which works by basically creating an associative array based on key's with roll number and values as name of the students.Next, it marks all the students absent that is 'a' and we begin filling of marks by *fill_marks* function which functions by again iterating through all the files and substituting the 'a' for the mark obtained by the student in that particular exam using the following sed command

```
sed -i -E "/^$roll_number/s/^(([^,]*,){$((col_index - 1))})([^,]*)/\1$marks/" main.csv
```

where column for the exam is extratced in the *col_index* variable using the following awk script

```
awk -F',' -v exam_name="$exam_name" 'NR==1 { for (i=1; i<=NF; i++) if ($i == exam_name)
{ print i; break } }' main.csv
```

This now finally creates the main.csv which we'll need

2.1.1 Flags

Here are some flags in the combine function which covers some edge/faulty cases

- It checks for the presence of a total column in the main.csv file, and if the total column is already present after running the script combine it again runs the *total* function which adds the total column back to the main.csv
- If grades.csv is present in the directory it doesn't consider it and doesn't include it for the combine function as it is not supposed to be added. (The grades.csv is generated in the grade functionality added later.)

2.2 Upload

```
bash submission.sh upload ~/Desktop/project.csv
```

Imports new CSV files into the script's directory for future processing during combining.Basically what it does is copy's uploaded files to the script's directory.

2.2.1 Flags

Here are some flags in the uplaod function which covers some edge/faulty cases

- It checks whether a file with the same name already exists in the current directory, if it does then we don't upload the file.
- It checks whethere the file whose path is given whether it actually exists or not,if it doesn't exist we throw an error.

2.3 Total

```
bash submission.sh total
```

Adds a "total" column to main.csv, calculating the total marks for each student based on existing marks, treating absentees with marks labelled as 'a' as 0.

2.3.1 Flags

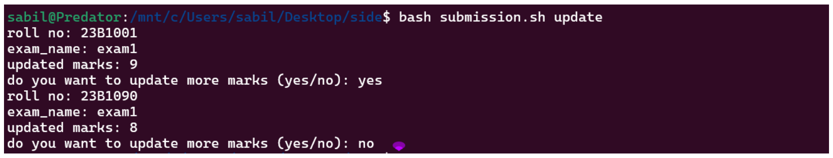
Here are some flags in the total function which covers some edge/faulty cases

- It checks whether the total column is already present in the *main.csv* if it does,we don't perform the total again.
- It checks whether the *main.csv* file exists in the current directory if it doesn't exist we throw an error.

2.4 Update

```
bash submission.sh update
```

This function updates the marks of a student in a particular exam and updates the marks for the student in the particular exam's *exam.csv* file as well as in the *main.csv* file by taking in the roll number and exam name as input and keeps updating marks until we stop and say no. After saying no,it re-combines all the files by calling the combine function and creates a new *main.csv* file which contains the updated marks for the student.
Here's a picture showing example of using the update function-



```
sabil@Predator:/mnt/c/Users/sabil/Desktop/side$ bash submission.sh update
roll no: 23B1001
exam_name: exam1
updated marks: 9
do you want to update more marks (yes/no): yes
roll no: 23B1090
exam_name: exam1
updated marks: 8
do you want to update more marks (yes/no): no
```

Figure 1: Example of running update function.

2.4.1 Flags

Here are some flags in the update function which covers some edge/faulty cases

- If we try to update the marks of a student who was absent in a given exam we won't be able to update the marks of the student for that exam,and it'll throw an error.
- It checks if the given roll number and exam names are valid , if they're not it throws an error.

3 Git Functionalities

3.1 git_init

`bash submission.sh git_init ./remote_repo`

This function initializes a new git repository of the files , it marks a directory as the remote repository for storing the commits file's the git log and other files required for the git functionalities.

It also creates a *.git_init.csv* which is a hidden file in the current directory, which stores the file path to the remote repository and can be used to access the file path for the remote repository.

3.1.1 Flags

Here are some flags in the git_init function which covers some edge/faulty cases

- It just checks if the given directory path is valid or not,if it is not valid it throws an error.

3.2 git_commit (using diff patch)

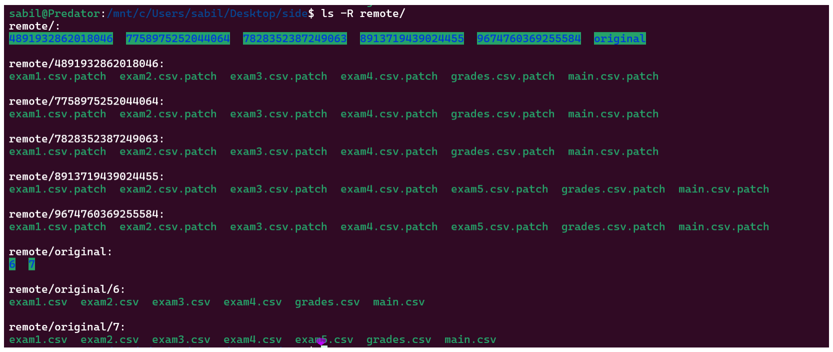
```
bash submission.sh git_commit -m "Example message for a commit"
```

Commits the changes to the remote repository with a unique hash ID,and shows which files are modified or if any new files are added compared to the previously made commit version of files. The commit files stored in the repository in two ways:-

- **original files**
  In in the intial commit, or in a commit at which new file is added the $.csvfiles$ are as it is copied from the working directory to the remote repository in a directory called original as they are the initial version of files which we will be referring to while patching and creating diff files for the later commits.

- **diff-patch files**
  For every commit we store a *.patch* file for every file stored in the directory for that commit .The diff files are generated with respect to the files which are stored in the original files directory that were committed at an initial commit,or at a commit where a new file was added as said above.

Here's an example showing a git remote repository file structure named as $./remote$



```
sabil@Predator:/mnt/c/Users/sabil/Desktop/side$ ls -R remote/
remote/:
4891932862018046  7758975252044064  7828352387249063  8913719439024455  9674760369255584  original

remote/4891932862018046:
exam1.csv.patch  exam2.csv.patch  exam3.csv.patch  exam4.csv.patch  grades.csv.patch  main.csv.patch

remote/7758975252044064:
exam1.csv.patch  exam2.csv.patch  exam3.csv.patch  exam4.csv.patch  grades.csv.patch  main.csv.patch

remote/7828352387249063:
exam1.csv.patch  exam2.csv.patch  exam3.csv.patch  exam4.csv.patch  grades.csv.patch  main.csv.patch

remote/8913719439024455:
exam1.csv.patch  exam2.csv.patch  exam3.csv.patch  exam4.csv.patch  exam5.csv.patch  grades.csv.patch  main.csv.patch

remote/9674760369255584:
exam1.csv.patch  exam2.csv.patch  exam3.csv.patch  exam4.csv.patch  exam5.csv.patch  grades.csv.patch  main.csv.patch

remote/original:
6  7

remote/original/6:
exam1.csv  exam2.csv  exam3.csv  exam4.csv  grades.csv  main.csv

remote/original/7:
exam1.csv  exam2.csv  exam3.csv  exam4.csv  exam5.csv  grades.csv  main.csv
```

Figure 2: Example of a git repository storing original and patch files.

After committing the files it adds all the info of the commit to the *.git_log* file,and tells the user which files are modified and tells whether any new file have been added or not.

3.2.1 Flags

Here are some flags in the git_commit function which covers some edge/faulty cases

- It just checks if the git_init has been initialized yet or not if the git init has not been done yet,it throws an error and asks the user to do initialization before committing any files.

3.3 git_checkout (using diff patch)

```
bash submission.sh git_checkout -m "Example message for a commit"

bash submission.sh git_checkout 12343    #random prefix value
```

The git checkout function can be called in two ways, one is with the -m flag and the other is without it. With the -m flag it searches through the git_log file trying to find an ID with the given matching message, If it gets the ID it then proceeds as described further.

After getting the ID from $git\_checkout - m$ or as provided by the prefix in normal *git_checkout* it searches for IDs with matching prefix and once it finds the exact ID which was required. It takes in the .patch files as stored in the commit and replaces all the .csv files in the current working directory with the patched files, made after patching the diff files from their corresponding original files.

3.3.1 Flags

Here are some flags in the git_checkout function which covers some edge/faulty cases

- If the message given in checkout -m doesn't match any commit's message it thrown an error and asks the user to enter a correct message.
- If the given prefix doesn't matches or the prefix is a prefix of multiple commits it throws respective conflicts or errors to the user,so the user can enter correct prefix value to checkout to a commit.
- It just checks if the git_init has been initialized yet or not if the git init has not been done yet,it throws an error and asks the user to do initialization before checking out to any previous commit.

3.4 git_log

```
bash submission.sh git_log
```

It just shows the git log file stored in the git remote repository.

3.4.1 Flags

Here are some flags in the git_checkout function which covers some edge/faulty cases

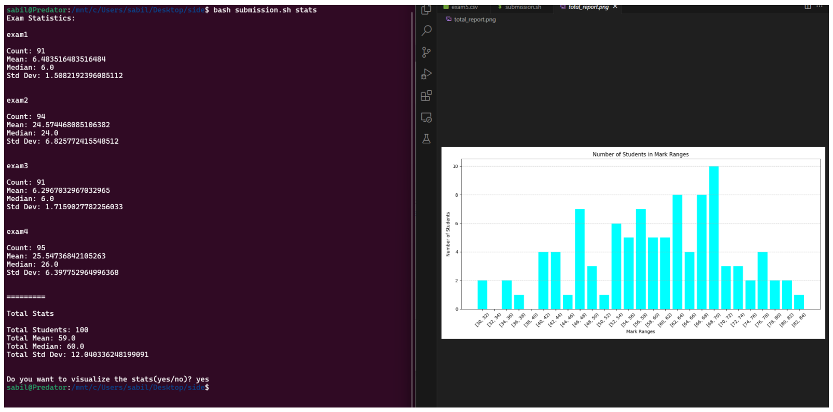
- It just checks if the git_init has been initialized yet or not if the git init has not been done yet,it throws an error and asks the user to do initialization before asking to display the git log file.

4 Additional Features

4.1 Stats

```
bash submission.sh stats
```

Additional Functionality to show the complete stats of the complete class for all each exams and on the total field which include mean, median,max,min etc with additional option which prompts the user whether to generate a bar graph showing visual distribution of marks for the total marks with functionalities taken from [3] ,[2] , [1]. Here's an example of the stats shown and the bar graph generated on using the function.



```
sabil@Predator:/mnt/c/Users/sabil/Desktop/side$ bash submission.sh stats
Exam Statistics:

exam1

Count: 91
Mean: 6.483516483516484
Median: 6.0
Std Dev: 1.5082192396085112


exam2

Count: 94
Mean: 24.574468085106382
Median: 24.0
Std Dev: 6.825772415548512


exam3

Count: 91
Mean: 6.2967032967032965
Median: 6.0
Std Dev: 1.7159027782256033


exam4

Count: 95
Mean: 25.54736842105263
Median: 26.0
Std Dev: 6.397752964996368


=========

Total Stats

Total Students: 100
Total Mean: 59.0
Total Median: 60.0
Total Std Dev: 12.040336248199091


Do you want to visualize the stats(yes/no)? yes
sabil@Predator:/mnt/c/Users/sabil/Desktop/side$
```

Figure 3: Example of the stats shown and the bar graph generated.

4.2 Grading

`bash submission.sh grade`

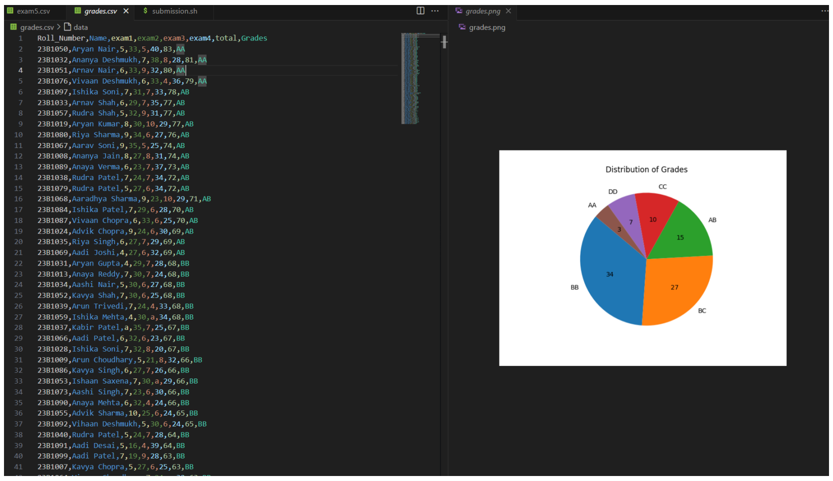
Additional Functionality to Grade the realatively grade the whole class based on a guassian distribution and generate a grades.csv which contains all the students with grades assigned in ascending order from AP to FF ,

It also generates a grades.png to show the grade distribution. The relative grading is based on the following policy:-

Grade	**Min Mark for the grade**
AP	$\mu + 2.2\delta$
AA	$\mu + 1.6\delta$
AB	$\mu + 0.8\delta$
BB	$\mu - 0.1$
BC	$\mu - 1\delta$
CC	$\mu - 1.5\delta$
DD	min_mark
FF	0

Where μ is the mean of the total marks and δ is the standard deviation of the total marks data.

Here's an example of the grades.csv and grades.png generated on running the command:-



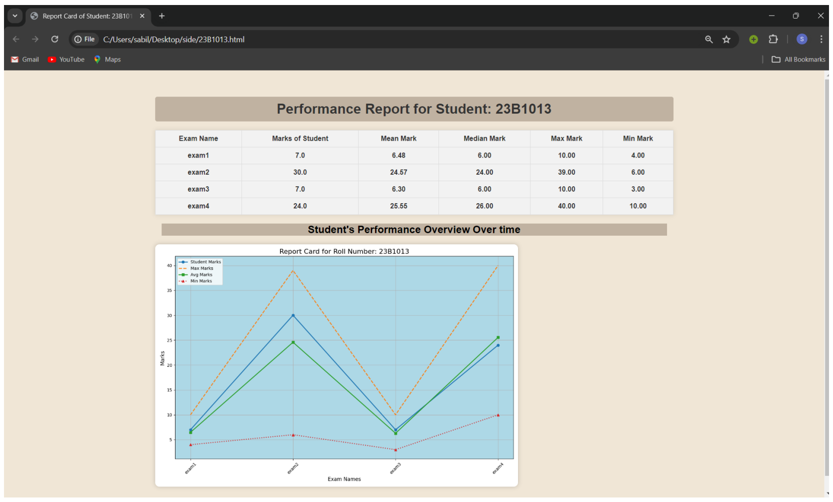
```
Roll_Number,Name,exam1,exam2,exam3,exam4,total,Grades
23B1050,Aryan Nair,5,33,5,40,83,AA
23B1032,Ananya Deshmukh,7,38,8,28,81,AA
23B1051,Arnav Nair,6,33,9,32,80,AA
23B1076,Vivaan Deshmukh,6,33,4,36,79,AA
23B1097,Ishika Soni,7,31,7,33,78,AB
23B1033,Arnav Shah,6,29,7,35,77,AB
23B1057,Rudra Shah,5,32,9,31,77,AB
23B1019,Aryan Kumar,8,30,10,29,77,AB
23B1080,Riya Sharma,9,34,6,27,76,AB
23B1067,Aarav Soni,9,35,5,25,74,AB
23B1008,Ananya Jain,8,27,8,31,74,AB
23B1089,Anaya Verma,6,23,7,37,73,AB
23B1038,Rudra Patel,7,24,7,34,72,AB
23B1079,Rudra Patel,5,27,6,34,72,AB
23B1068,Aaradhya Sharma,9,23,10,29,71,AB
23B1084,Ishika Patel,7,29,6,28,70,AB
23B1087,Vivaan Chopra,6,33,6,25,70,AB
23B1024,Advik Chopra,9,24,6,30,69,AB
23B1035,Riya Singh,6,27,7,29,69,AB
23B1069,Aadi Joshi,4,27,6,32,69,AB
23B1031,Aryan Gupta,4,29,7,28,68,BB
23B1013,Anaya Reddy,7,30,7,24,68,BB
23B1034,Aashi Nair,5,30,6,27,68,BB
23B1052,Kavya Shah,7,30,6,25,68,BB
23B1039,Arun Trivedi,7,24,4,33,68,BB
23B1059,Ishika Mehta,4,30,a,34,68,BB
23B1037,Kabir Patel,a,35,7,25,67,BB
23B1066,Aadi Patel,6,32,6,23,67,BB
23B1028,Ishika Soni,7,32,8,20,67,BB
23B1009,Arun Choudhary,5,21,8,32,66,BB
23B1086,Kavya Singh,6,27,7,26,66,BB
23B1053,Ishaan Saxena,7,30,a,29,66,BB
23B1073,Aashi Singh,7,23,6,30,66,BB
23B1090,Anaya Mehta,6,32,4,24,66,BB
23B1055,Advik Sharma,10,25,6,24,65,BB
23B1092,Vihaan Deshmukh,5,30,6,24,65,BB
23B1040,Rudra Patel,5,24,7,28,64,BB
23B1091,Aadi Desai,5,16,4,39,64,BB
23B1099,Aadi Patel,7,19,9,28,63,BB
23B1007,Kavya Chopra,5,27,6,25,63,BB
```

Figure 4: Example of the grades.csv and grades.png generated.

4.3 Generate Report Card Using HTML

```
bash submission.sh report_card 23B1013
```

Additional Functionality to generate a report card nad shows it in a web browse of a student in the form of a .html file comparing student's marks in each exam with various stats in a table and showing the performance line chart of the student over time.
Here's an example of the .html file generated on running the command:-


Report Card of Student: 23B101
C:/Users/sabil/Desktop/side/23B1013.html
Gmail
YouTube
Maps
All Bookmarks

Performance Report for Student: 23B1013

Exam Name	Marks of Student	Mean Mark	Median Mark	Max Mark	Min Mark
exam1	7.0	6.48	6.00	10.00	4.00
exam2	30.0	24.57	24.00	39.00	6.00
exam3	7.0	6.30	6.00	10.00	3.00
exam4	24.0	25.55	26.00	40.00	10.00

Student's Performance Overview Over time

Report Card for Roll Number: 23B1013
Student Marks
Max Marks
Avg Marks
Min Marks
Marks
Exam Names
exam1
exam2
exam3
exam4

Figure 5: Example webpage generated on running the generate report card command.

4.4 Efficient commits and checkout using diff and patch

All the committing and the checkout process is done through diff files stored at every commit rather than simply copying the files from the current working directory and storing them as it is there.

Storing only the changes made (using diff output) instead of entire files saves significant storage space, especially for large projects with frequent updates. This efficiency is crucial for repositories with limited storage capacity.

Also for checking out to a previous commit we just patch the diff files with one original version of files stored making the process much more efficient.

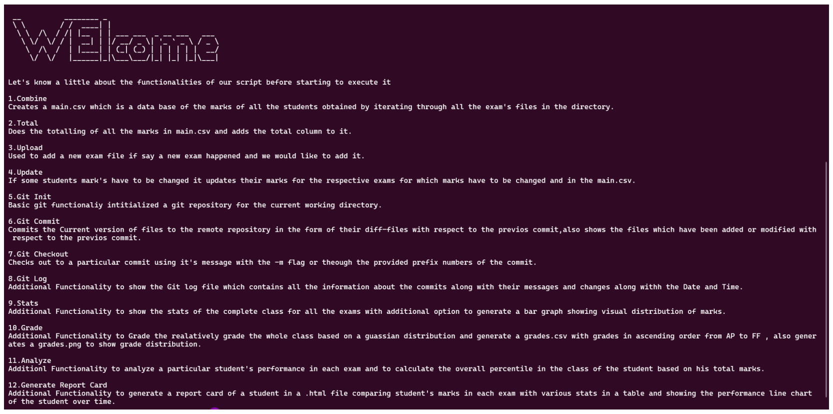
4.5 git_log

It adds an extra functionality in git to show the git_log file with information contained of every commit,that is the commit history in a chronological order in which commits are made with their unique commit hash that is the commit ID and also contains the message given along with the commit.

4.6 welcome

```
bash submission.sh welcome
```

When running the script initially, it displays the user with all the available functions and given the user a brief about their usage and functionalities so that the user has a good idea of what the scriptis capable of along with a welcome ASCII art as shown below :) .



```
Let's know a little about the functionalities of our script before starting to execute it

1.Combine
Creates a main.csv which is a data base of the marks of all the students obtained by iterating through all the exam's files in the directory.

2.Total
Does the totalling of all the marks in main.csv and adds the total column to it.

3.Upload
Used to add a new exam file if say a new exam happened and we would like to add it.

4.Update
If some students mark's have to be changed it updates their marks for the respective exams for which marks have to be changed and in the main.csv.

5.Git Init
Basic git functionaliy intitialized a git repository for the current working directory.

6.Git Commit
Commits the Current version of files to the remote repository in the form of their diff-files with respect to the previos commit,also shows the files which have been added or modified with
 respect to the previos commit.

7.Git Checkout
Checks out to a particular commit using it's message with the -m flag or theough the provided prefix numbers of the commit.

8.Git Log
Additional Functionality to show the Git log file which contains all the information about the commits along with their messages and changes along withh the Date and Time.

9.Stats
Additional Functionality to show the stats of the complete class for all the exams with additional option to generate a bar graph showing visual distribution of marks.

10.Grade
Additional Functionality to Grade the realatively grade the whole class based on a guassian distribution and generate a grades.csv with grades in ascending order from AP to FF , also gener
ates a grades.png to show grade distribution.

11.Analyze
Additionl Functionality to analyze a particular student's performance in each exam and to calculate the overall percentile in the class of the student based on his total marks.

12.Generate Report Card
Additional Functionality to generate a report card of a student in a .html file comparing student's marks in each exam with various stats in a table and showing the performance line chart
of the student over time.
```

Figure 6: Example showing all the functions being defined by welcome.

4.7 Error Flags

Every function present is been guarded by some flags which prevents the user from doing something which is out of order from the scheme of things and warns it of the possible errors which it might be making, so that he/she can make them right.

References

[1] Kameshwari Chebrolu. *CS108 Notes Bash,Sed,Awk,Numpy,HTML,Matplotlib and Python Course slides.* 2024.

[2] Geeks for Geeks. Geeks for geeks - pandas tutorial, 2024.

[3] Matplotlib Development Team. Matplotlib documentation, 2024.