Project 2

Exam project

Program for grading students

Introduction

In this project you must develop, test, and document a program for processing grades for students. You must implement the functions and main script described in the following according to the specifications.

Grade rounding function

Interface

```
function gradesRounded = roundGrade(grades)
% Insert your code here
```

Input arguments   grades: A vector (each element is a number between -3 and 12).

Output arguments   gradesRounded: A vector (each element is a number on the 7-step-scale).

User input   No.

Screen output   No.

Description   The function must round off each element in the vector grades and return the nearest grade on the 7-step-scale:

7-step-scale: Grades	12	10	7	4	02	00	−3

For example, if the function gets the vector [8.2, -0.5] as input, it must return the rounded grades [7, 0] which are the closest numbers on the grading scale.

Final grade function

Interface

```
function gradesFinal = computeFinalGrades(grades)
% Insert your code here
```

Input arguments   grades: An $N \times M$ matrix containing grades on the 7-step-scale given to N students on M different assignments.

Output arguments   gradesFinal: A vector of length n containing the final grade for each of the N students.

User input   No.

Screen output   No.

Description   For each student, the final grade must be computed in the following way:

1. If there is only one assignment ($M = 1$) the final grade is equal to the grade of that assignment.

2. If there are two or more assignments ($M > 1$) the lowest grade is discarded. The final grade is computed as the mean of $M - 1$ highest grades rounded to the nearest grade on the scale (using the function roundGrade).

3. Irrespective of the above, if a student has received the grade -3 in one or more assignments, the final grade must always be -3.

Grades plot function

Interface	`function gradesPlot(grades)` `% Insert your code here`
Input arguments	`grades`: An $N \times M$ matrix containing grades on the 7-step-scale given to N students on M different assignments.
Output arguments	No.
User input	No.
Screen output	Yes (plots, see specifications below.)
Description	This function must display two plots:

1. "Final grades": A bar plot of the number of students who have received each of possible final grades on the 7-step-scale (computed using the function `computeFinalGrades`).

2. "Grades per assignment": A plot with the assignments on the x-axis and the grades on the y-axis. The x-axis must show all assignments from 1 to M, and the y-axis must show all grade -3 to 12. The plot must contain:

   1. Each of the given grades marked by a dot. You must add a small random number (between -0.1 and 0.1) to the x- and y-coordinates of each dot, to be able tell apart the different dots which otherwise would be on top of each other when more than one student has received the same grade in the same assignment.
   2. The average grade of each of the assignments plotted as a line

The plots should include a suitable title, descriptive axis labels, and a data legend where appropriate. You are allowed to present the plots in separate figure windows or as sub-plots in a single figure window.

Main script

Interface	Must be implemented as a script.
Input arguments	No.
Output arguments	No.
User input	Yes (see specifications below.)
Screen output	Yes (see specifications below.)
Description	When the user runs the main script he/she must first be asked to enter the name of a comma-separated-values (CSV) file containing grades given to a number of students for a number of assignments (see description of file format below). Remember to check if the file name is valid. After reading in the data, you must display some information about the loaded data, including at least the number of students and the number of assignments.

Next, the user must have at least the following options:

1. Load new data.
2. Check for data errors.
3. Generate plots.
4. Display list of grades.
5. Quit.

The user must be allowed to perform these actions (see specifications below) in any order as long as he/she chooses, until the user decides to quit the program. The details of how the main script is designed and implemented is for you to determine. It is a requirement that the program is interactive, and you should strive to make it user friendly by providing sensible dialogue options. Consider what you would expect if you were to use such a script, and what you think would be fun. Play around and make a cool script.

File format

The first row in the CSV file will contain the column headings. Each of the following rows will contain a student id, a name, and a number of grades for a student. An example of a data file with four students and three assignments is given below:

```
StudentID,Name,Assignment1,Assignment2,Assignment3
s123456,Michael Andersen,7,7,4
s123789,Bettina Petersen,12,10,10
s123468,Thomas Nielsen,-3,7,2
s123579,Marie Hansen,10,12,12
```

Remember that your program should work for any number of students and any number of assignments. You are encouraged to make your own test data file in order to validate that your program functions correctly.

Error handling

You must test that all input provided by the user are valid. If the user gives invalid input, you must provide informative feedback to the user and allow the user to provide the correct input.

1. Load new data

If the user chooses to load new data, the user must be asked to input a valid filename of a data file, and data must be loaded in the same way as in the beginning of the script.

2. Check for data errors

If the user chooses to check for data errors, you must display a report of errors (if any) in the loaded data file. Your program must at least detect and display information about the following possible errors:

1. If two students in the data have the same student id.
2. If a grade in the data set is not one of the possible grades on the 7-step-scale.

3. Generate plots

If the user chooses to generate plots, call the `gradesPlot` function to display the plots.

4. Display list of grades

If the user chooses to display the list of grades, you must display the grades for each assignment as well as the final grade for all of the students in alphabetical order by their name. The displayed list must be formatted in a way so that it is easy to read.

5. Quit

If the user chooses to quit the program, the main script should stop.