CSc 110 Assignment 5

File I/O and 1D Arrays

Reminders:

Submit your java file name EXACTLY as specified (GradeCalculator.java) using the 'Assignments' link of the course conneX site. ** Don't forget to follow the Style Guidelines posted – this is part of the requirements of every assignment.

It is OK to talk about your assignment with your classmates, and you are encouraged to design solutions together, but each student must implement their own solution. We will be using plagiarism detection software on your assignment submissions.

Learning Outcomes:

When you have completed this assignment, you should understand:

- How to receive input from a file
- How to design and use indefinite loops to handle invalid inputs.
- How to handle File IO exceptions.
- How to print to an output file.
- How to create a one dimensional array.
- How to access and change values in an array.
- How to iterate through an array using a loop.

Assignment Description:

The teaching team needs some help calculating grades for CSC 110 at the end of term. This assignment requires you to implement a Grade Calculator that will take a raw set of student marks from the term and generate a final grade report.

The input file contains a unknown number of lines, where each line is a single student's set of marks.
Each line is comprised of the student's marks for the term (separated by white space) formatted as follows:
Labs A1 A2 A3 A4 A5 A6 A7 Midterm1 Midterm2 FinalExam

The corresponding max grade of each component of the course is:
5 10 10 10 10 10 10 10 20 35 110

The corresponding weight of each component of the course as a percentage is:
5 4 4 4 4 4 4 4 15 15 37

EXAMPLE:

the following sample line entry…
5 10 10 9 10 10 10 13 20.25 34.5 105

would calculate the following weighted values:
5.0 4.0 4.0 3.6 4.0 4.0 4.0 5.2 15.1875 14.785714285714286 35.31818181818182

and the sum of those weighted values would be: 99.0913961038961

if we cast this total sum to be an int (which will take the floor of the final result) the final grade percentage for this sample line would be: 99

The final grade percentage is >89 and therefore converts to: A+
NOTE: follow letter grade breakdown on your CSC110 course outline

Programming Problem Description:

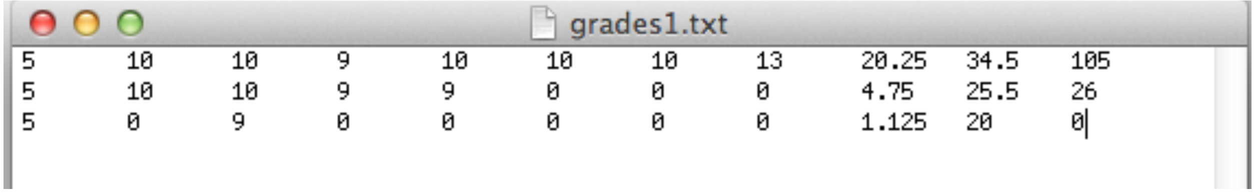
Your program must include the methods outlined on Page 4 of this document and your algorithm should include the following steps…

- Prompt the user to input the name of their file at the command line
  - If they enter the name of a file that does not exist you should keep prompting them until they enter the name of a file that exists.
- Find out how many grade entries are in the file (see required method list on page 4)
- Calculate the grade percentage for each line in the input file and store the final grades of each student into a one dimensional array of integers
- Generate a report into a file named gradeReport.txt which will contain:
  - The maximum grade percentage (see required method list on page 4)
  - The minimum grade percentage (see required method list on page 4)
  - The average grade percentage (see required method list on page 4)
  - Output of calculated percentage with corresponding Letter Grades (see required method list on page 4)
  - A count of Letter Grades occurrences (see required method list on page 4)

A Sample Run is provided on the following page.
Sample input and output files are also provided as attachments to the assignment in ***grades.txt*** and corresponding ***gradeReport.txt***

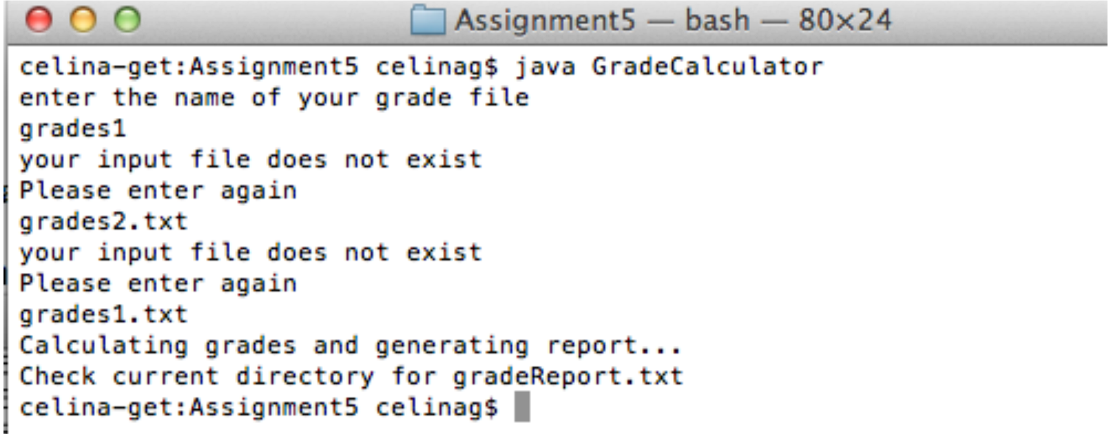
Sample Run

Input file opened in plain text editor :



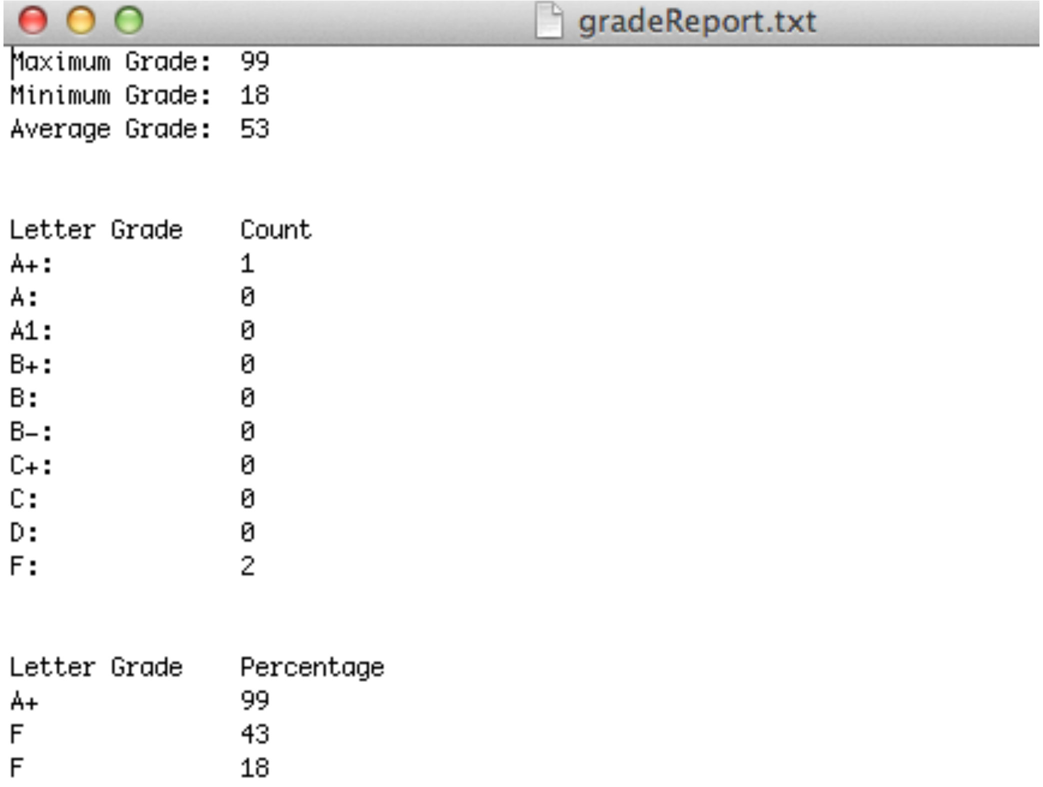
```
5       10      10      9       10      10      10      13      20.25   34.5    105
5       10      10      9       9       0       0       0       4.75    25.5    26
5       0       9       0       0       0       0       0       1.125   20      0
```

Console Output:



```
celina-get:Assignment5 celinag$ java GradeCalculator
enter the name of your grade file
grades1
your input file does not exist
Please enter again
grades2.txt
your input file does not exist
Please enter again
grades1.txt
Calculating grades and generating report...
Check current directory for gradeReport.txt
celina-get:Assignment5 celinag$ 
```

Output file opened in plain text editor:



```
Maximum Grade:  99
Minimum Grade:  18
Average Grade:  53


Letter Grade    Count
A+:             1
A:              0
A1:             0
B+:             0
B:              0
B-:             0
C+:             0
C:              0
D:              0
F:              2


Letter Grade    Percentage
A+              99
F               43
F               18
```

Required Methods

You may and SHOULD add additional methods to the following but your implementation **MUST** include the following method signatures **EXACTLY** in order to be graded.

```
/*
 *    PURPOSE:    gets the number of line entries in a file
 *    INPUT:      File fIn, file with text
 *    OUTPUT:     returns number of lines in the file as an integer
 */
 public static int getNumEntries(File fIn)

/*
 *    PURPOSE:    prints an array of integers to the console
 *    INPUT:      int[] values, array of integers
 *    OUTPUT:     none
 */
public static void print(int[] values)

/*
 *    PURPOSE:    identifies the min value in an array of integers
 *    INPUT:      int[] values, array of integers
 *    OUTPUT:     returns the min value as an integer
 */
public static int min(int[] values)

/*
 *    PURPOSE:    identifies the max value of an array of integers
 *    INPUT:      int[] values, array of integers
 *    OUTPUT:     returns the max value as an integer
 */
public static int max(int[] values)

/*
 *    PURPOSE:    calculates the average of an array of integers
 *    INPUT:      int[] values, array of integers
 *    OUTPUT:     returns the average as an integer
 */
public static int avg(int[] values)

/*
 *    PURPOSE:    gets the corresponding letter grades
 *                for an array of integers
 *    INPUT:      int[] grades, array of grades as integers
 *    OUTPUT:     returns the letter grades in a String array
 */
public static String[] getLetterGrades(int[] grades)

/*
 *    PURPOSE: counts the number grades matching each letter grade
 *          in the array of Strings passed in as a parameter.
 *          The counts will be stored in an array of 10 integers where,
 *          index 0 holds the number of "A+"'s in grades array
 *          index 1 holds the number of "A"'s in grades array
 *          ...
 *          index 9 holds the number of "F"'s in grades array
 *    INPUT:  String[] grades, array of letter grades as Strings
 *    OUTPUT: returns an integer array holding the letter grade counts
 */
public static int[] countGrades(String[] grades )
```