COMP1006/1406 – Summer 2016
Updated - Problem 1

Submit a single file called `assignment1.zip` to cuLearn.
There are 105 marks possible marks. The assignment it out of 100.
(You can receive as much as 105/100.)

In assignment specifications, text appearing `like this` represents things that a user types when running or using a program. Text that appears `like this` represents output from your program.

0: Rubric [0 marks]

Grade your assignment using the provided rubric. Use integer or 1/2 integer grades for each component. This will not be graded.

Include your `rubric01.txt` file in your `assignment1.zip`.

1: Final Grade Computation [20 marks]

Write a program that computes a final grade, as specified in the course outline (available on the course webpage `http://people.scs.carleton.ca/~mjhinek/COMP1406/`) using the six term marks: assignments, project, tutorials, quizzes, midterm and final exam.

The six term marks will be provided to the program via command line arguments (values passed to the program when you run the program from the console window in DrJava) and will always be given in the same order: A, P, T, Q, M, F.

➡ Starting with the skeleton program provided, `Grades.java`, add code to the `computeGrade` and `roundGrade` methods so that the program outputs to standard output (`System.out`) the correct final grade. Do not change the `main` method. Your program will output a single line with the computed final grade rounded to exactly one decimal place. The grades provided from the command line can be integers or decimal numbers (all are given as a percentage). To allow for possible bonus marks, valid percentages can range from 0.0 to about 105.0. The roundGrade function will take a double and return a string that represents that number rounded to exactly one decimal place.

Add comments, whitespace, and indentation to the code so that the entire file (`Grades.java`), not just the functions you are completing, follows the style guide. Marks will be deducted if you do not follow the style guide.

Examples: From the console window in DrJava,

`java Grades 75.1 30 45.1 77 65.2 72.3` will output the single line `65.2`

`java Grades 99 90 95.1 97 51.1 37.2` will output the single line `41.4`

Testing: Create and submit another program called `GradesTesting`, that tests your `Grades` program. For each test case, your program should display the input (grades), the expected output of your program and the actual output. For example, a single test case might look like

```
String[] input = {"75.1", "30", "45.1", "77", "65.2", "72.3"};
System.out.println("Test 0:  " + input);
System.out.println(" Expected answer is 65.2");
System.out.print(" Answer obtained is ");                    ← note the change
Grades.main(input);                                          ← note the change
```

Your testing program should also follow the style guide. In particular, be sure you describe (in comments) each test case (or group of test cases). The expected and actual output should line up nicely on the screen.

Mark breakdown: 5 marks for Style, 5 marks for Testing, 10 marks for Correctness

> Include both your `Grades.java` and `GradesTesting.java` files in your submission `assignment1.zip` file.

2: Grade Conversion [20 marks]

➡ In the provided `Convert.java` file, complete the method called `convertToLetter`. The interface (specification) of the method is given below.

```
public static String convertToLetter(double grade)
 /* Purpose: converts a given numerical grade to a letter grade             *
  * Input  : a number                                                       *
  * output : the letter grade (F, D-, D, ..., A+) corresponding to the      *
  *          input grade if the input is valid, "Invalid" otherwise         */
```

The conversion table is given as follows

Output	Input Range
A+	[90, 100]
A	[85, 90)
A-	[80, 85)
B+	[77, 80)
B	[73, 77)
B-	[70, 73)
C+	[67, 70)
C	[63, 67)
C-	[60, 63)
D+	[57, 60)
D	[53, 57)
D-	[50, 53)
F	[0, 50)
Invalid	other

Note that the range $[a, b)$ means any number x such that $a \leq x < b$. That is, square brackets mean inclusive and parentheses mean exclusive.

Mark breakdown: 5 marks for Style, 15 marks for Correctness

Put your `Convert.java` file in your `assignment1.zip` file.

Examples:

`Convert.convertToLetter(60.0)` $\xrightarrow{\text{returns}}$ `C-`.

`Convert.convertToLetter(49.9)` $\xrightarrow{\text{returns}}$ `F`.

3: Longest Streak [20 marks]

➡ In the provided `Problem3.java` file, complete the method called `longestStreak`. There should be no other methods or attributes in your class. The contract (specification) of the method is given below.

```
public static int longestStreak(boolean[] values)
 /* Purpose: computes the length of a longest streak of consecutive         *
  *          true occurrences in the input argument values                  *
  * Input  : values is a non-null array of booleans with length at least 1  *
  * output : outputs the maximal number of consecutive trues found in       *
  *          the input array                                                */
```

Do not change the method signature (use the provided skeleton java file). Changing the method modifiers, return type, name or input arguments will result in zero correctness marks for this problem.

Mark breakdown: 5 for style. 15 for correctness

Put your `Problem3.java` file in your `assignment1.zip` file.

Examples: Your `longestStreak` method should behave as follows:

```
boolean[] test_case = new boolean[]{false, true, true, false};
Problem1.longestStreak( test_case ) returns 2

test_case = new boolean[]{false};
Problem1.longestStreak( test_case ) returns 0

test_case = new boolean[]{true, false, true};
Problem1.longestStreak( test_case ) returns 1

test_case = new boolean[]{false, true, true, true, false, true, true, true};
Problem1.longestStreak( test_case ) returns 3
```

Note: Examples are provided as a guide to help you get your programs/methods working as specified. However, having your code match the examples is not a guarantee that your code will receive high correctness marks. You must test your code more extensively.

4: Longest Streak Again [20 marks]

➡ In the provided `Problem4.java` file, complete the following method:

```
public static int[] longestStreak(boolean[] values)
 /* Purpose : computes the length and locations of all the  maximal           *
  *           sequences of consecutive true occurrences in values             *
  * Inputs  : values is a non-null array of booleans with length at least 1   *
  * Outputs : outputs an integer array with one or more elements              *
  *              - first element is the length of a maximal sequence of       *
  *                consecutive trues in the input array values                *
  *              - the next elements are the indexes of the starting points   *
  *                (in the input array values) of each of the maximal         *
  *                sequences of consecutive trues                             *
  *           if there are no true values in the input array, output [0]      */
```

Do not change the method signature (use the provided skeleton java file). Changing the method modifiers, return type, name or input arguments will result in zero correctness marks for this problem.

Mark breakdown: 5 for style, 15 for correctness

Put your `Problem4.java` file in your `assignment1.zip` file.

Examples: using the same examples as Problem 3, the output would be (in the same order) `[2,1]`, `[0]`, `[1,0,2]` and `[3,1,5]`. Other examples include

```
boolean[] test_case = new boolean[]{true, false, true, false, true};
Problem2.longestStreak( test_case ) returns [1,0,2,4]
```

```
test_case = new boolean[]{false, false, false, true, true};
Problem2.longestStreak( test_case ) returns [2,3]
```

5: Tic-Tac-Toe [25 marks]

In this problem you will write a program to play tic-tac-toe against a computer. Tic-tac-toe is a game that is played on a 3×3 grid. We will label each of the 9 grid elements using the following convention

1	2	3
4	5	6
7	8	9

See `http://en.wikipedia.org/wiki/Tic-tac-toe` if you are unfamiliar with the game.

➡ Write a program called `TicTacToe`. You will write this game using concepts of procedural programming (not object oriented programming). Your program should behave as follows:

- Your `main` method will drive your game by calling other static methods in your class. The overall control flow of your game will be in `main` but the details of the different actions will be in other static methods.
- Particular tasks should have their own method.
- You will use `Scanner` objects for user input. You may use `String` objects. You may use `array` objects. Your program should not use any other objects.
- Your program will let a user play as many games as they like, until they decide to stop playing. (You must ask the user if they wish to play another game after each game has ended. The expected answers to this question will be yes or no, and be case insensitive).
- Your game must be (relatively) user friendly. That it, it must provide useful prompts to the user during the game. It should say when a game has ended and who won, etc.

- You may assume that all input is valid. When asking for a position to play, the user will always enter a single number between 1 and 9 (inclusive). When asked to play another game, the user will always enter yes or no (or Yes, YEs, yES, nO, NO, etc.).
- Your game must play a valid game of tic-tac-toe. If the user specifies a location is that already taken, an appropriate message must be displayed and the user must be prompted to try again.
- The playing grid should be displayed showing the current state of the game. You should redraw the game after each computer move (so that the user can see what the board looks like).

Mark breakdown: 5 marks for Style, 5 marks for Design, 15 marks for Correctness

Put your `TicTacToe.java` file in your `assignment1.zip` file.

Submission Recap

A complete assignment will consist of a single file (`assignment1.zip`) with the following seven (7) files included: `rubric01.txt`, `Grades.java`, `GradesTesting.java`, `Convert.java`, `Problem3.java`, `Problem4.java` and `TicTacToe.java`.