Assignment 1

Learning Outcomes

In this assignment, you will get practice with:

- Creating classes and objects of those classes
- Overloading constructors
- Implementing equals(), toString(), getters, and other methods
- Working with arrays
- Using loops and conditionals

Introduction

Most of us are probably familiar with the beloved family game '**Scrabble**'. In Scrabble, players collect seven random tiles, each consisting of a letter A through Z (for the sake of simplicity, let us that assume the 'blank' tile does not exist in this version of the game). Players then use these tiles to create words that consist of one or more of the letters they've randomly selected.

When a word is created using one or more of the seven selected tiles, the *value* of each of the tiles is summed, and that score is added to the player's cumulative score. The individual value of each tile is presented below.

A_1 B_3 E_1 F_4 I_1 J_8
C_3 D_2 G_2 H_4 K_5 L_1
M_3 N_1 Q_{10} R_1 U_1 V_4
O_1 P_3 S_1 T_1 W_4 X_8
Y_4 Z_{10}

Assignment 1

In this assignment, you are provided with a text file consisting of all 279,496 words published in the 2019 version of the Collins Scrabble Word dictionary.

Given the letters of seven randomized Scrabble tiles, you must determine the set of scores that a player could possibly obtain by placing these tiles. We will be assuming three traits that differ slightly from the traditional game of Scrabble:

1. There is an **unlimited** amount of every letter within the scrabble bag, thus it is reasonable to assume (though incredibly unlikely) that you could obtain something like ['A', 'A', 'A', 'A', 'A', 'A', 'A'].
2. You will be placing these tiles during the **first turn** on the board. Thus, you do not need to worry about combining your word with any pre-existing letters that might have already been placed on the Scrabble board.
3. Assume that there exist no word or letter bonuses. We will simply be basing the score off the tile values.

As an example, assume that your seven randomized tiles read:

{'A', 'C', 'A', 'A', 'B', 'A', 'H'}

The output should be 1) a list of length n of words that can be created using these letters, and 2) a list of length n containing the scores (integers) for each of the words of length $1 \leq m \leq 7$ that can be created using these letters (like 'AA', or 'AAH', which are both words within the Collins Scrabble Word dictionary, surprisingly). So two of the scores that will be in your score set for the above tile set are **2** (A (1) + A (1) = 2), and **6** (A (1) + A (1) + H (4) = 6).

Provided Files

The following is a list of files provided to you for this assignment. **Do NOT alter these files**.

- CollinsScrabbleWords2019.txt
- TestGame.java

Assignment 1

CS 1027

Computer Science Fundamentals II

Classes to Implement

For this assignment, you must implement two java classes: Tile and Scrabble.
Follow the guidelines for each one below.

In these classes, you can implement more private (helper) methods if you want
to. You may not, however implement more public methods. You may not add
instance variables other than the ones specified below nor change the variable
types or accessibility (i.e. making a variable public when it should be private).
Penalties will be applied if you implement additional instance variables or change
the variable types of modifiers from what is described here.

Tile.java

This class represents a single Scrabble tile that will be used in the game.

The class must have the following private variables:

- value (char)

The class must have the following public methods:

- public Tile() [constructor]
    - Initialize value to ' '
- public Tile(char) [constructor]
    - Initialize value to the given argument
- public void pickup()
    - Generate a random character between A and Z (inclusive) and set the
      value to that letter.
    - Feel free to use 'java.util.random' for this method
- public char getValue()
    - Returns the tile value

Assignment 1

Scrabble.java

This class represents the Scrabble game in which there are seven randomly selected tiles, and scoring is performed for each possible word (this will be the tougher class to implement).

The class must have the following private variables:

- tiles (Tile[ ])

The class must have the following public methods:

- public Scrabble() [constructor]
  - Initialize the Tile array and 'pickup' for random values
- public Scrabble(Tile [ ]) [constructor]
  - Initialize the tile array with the given argument
- public String getLetters()
  - Return a string that is all of the tile characters (for example, "ABFEODL")
- public ArrayList<String> getWords()
  - Create an ArrayList of Strings with *n* elements. Each element should represent a word that can be created using the current tiles.
  - The algorithm for this method should reference the provided file **CollinsScrabbleWords2019.txt**
  - ** do NOT put this file somewhere on your local machine and hardcode the local directory. This will likely cause your tests to fail on GradeScope. Also, do not put it within a folder in the relative path.
- public int[ ] getScores()
  - Create an int array with *n* elements. Each element in this list should represent each individual score for each word that can be created using the current tiles. This should be returned **in ascending order**.
- public Boolean equals(Scrabble)
  - Compare the given Scrabble object from the argument with the 'this' object to see if they are equal (do they have the same tiles?).

Assignment 1

Marking Notes

Functional Specifications

- Does the program behave according to specifications?
- Does it produce the correct output and pass all tests?
- Are the classes implemented properly?
- Does the code run on Gradescope (even if it runs on Eclipse, it is up to you to ensure it works on Gradescope to get the test marks)
- Does the program produce compilation or run-time errors on Gradescope?
- Does the program fail to follow the instructions (i.e, changing variable types, etc)

Non-Functional Specifications

- Are there comments throughout the code? Remember that *if you run out of time/fail to finish parts of the implementation, comments explaining what you would do if you were able to get things working are where you're going to get part marks!*
- Are the variables and methods given appropriate, meaningful names? Avoid things like 'temp', 'myvar', and 'goodForNothingVariable'.
- Is the code clean and readable with proper indenting and white-space?
- Is the code consistent regarding formatting and naming conventions?
- Submission errors (i.e, missing files, too many files, etc.) will receive a penalty of 5%
- Including a "package" line at the top of a file will receive a penalty of 5%

Assignment 1

Computer Science Fundamentals II

** Remember you must do all the work on your own. Do not copy or even look at the work of another student. All submitted code will be run through similarity-detection software.

Submission (due Monday, July 4th, 2022 at 11:55pm ET)

Assignments must be submitted to Gradescope, **not on OWL**. If you are new to this platform, see <u>these instructions</u> on submitting on Gradescope.

Rules

- Please only submit the files specified below.
- Do not upload the .class files! Penalties will be applied for this.
- Submit the assignment on time. Late submissions will receive a penalty of 10% per day.
- Forgetting to submit is not a valid excuse for submitting late.
- Submissions must be done through Gradescope. **If your code runs on Eclipse but not on Gradescope, you will NOT get the marks! Make sure it works on Gradescope to get these marks.**
- You are expected to perform additional testing (create your own test harness class to do this) to ensure that your code works for other tile combinations as well. We are providing you with some tests but we may (definitely will) use additional tests that you haven't seen before for marking.
- Assignment files are NOT to be emailed to the instructor(s) or TA(s). They will not be marked if sent by email.
- You may re-submit code if your previous submission was not complete or correct, however, re-submissions after the regular assignment deadline will receive a penalty.

Assignment 1

Files to Submit

- Tile.java
- Scrabble.java

Grading Criteria

Total Marks: [20]

Functional Specifications

[1] Tile.java

[6] Scrabble.java

[10] Passing Tests

Non-Functional Specifications

[1] Meaningful variable names, private instance variables

[1] Code readability and indentation

[1] Code comments