CSCI 1101 - 2017
Assignment 1

Complete the assigned tasks and submit a ZIP file containing your finished source code (these are the .JAVA files) on Brightspace.

Your assignment **must be your own work**. You may ask questions of your instructor, course TAs, or Learning Centre TAs. If you discuss the ideas in this assignment with your classmates, **it is your responsibility to make sure you are not plagiarizing** – take no notes during the discussion, wait 30 minutes, and whatever you remember is safe to be used in your own code.

Submission Deadlines (firm):
Due: Friday July 7, by 12:00pm (noon)
Late submission (10% penalty): Saturday July 8, by 12:00pm (noon)

Submissions outside of these given deadlines are **not** accepted.

Before submitting, check that:

1. Your code is properly formatted, includes reasonable comments (including the header comment), and is properly tested for the given problem.
2. Your submission ZIP file contains your **source code** .JAVA files, not your compiled .CLASS files.
3. Your code **compiles** and can be run – even if your program does not do everything perfectly, make sure it compiles before you submit.

Header Comments

Your code should now include header comments for **all** of your class (.java) files. The comment should include the lab/assignment number, the course (CSCI 1101), the name of your program and a short description of the entire class, the date, your name and Banner ID, and a declaration that matches the first page of this document (e.g., whether you received help). See the example below for what a header comment should include:

```
/*Lab1, Question 1 CSCI 1101
   Student.java holds information about a student at Dalhousie in CSCI1101
and
   their grades
   June 29, 2015
   John Smith B00112345
   This is entirely my own work. */

public class Student {
//rest of Code
```

If applicable, your demo class should then also have a similar header:

```
/*Lab1, Question 1  - demo class CSCI 1101
   StudentDemo.java is a demo program for the Student class. It creates
student
   objects, and compares different students.
   June 29, 2015
   John Smith B00112345
   I received help with creating Student objects from my TA but the rest is
my
   own work. */

public class StudentDemo {
//rest of Code
```

Question 1
Consider a class that could be used to play a computer version of the long running TV game *Wheel of Fortune*.
http://www.wheeloffortune.com/

On the TV game, there are three players who try and guess a phrase. They get to spin a wheel with different dollar amounts (e.g., $200, $500, $1000, etc). Then they guess a letter. If the letter is in the hidden phrase they get the value that they spun multiplied by the number of the guessed letter in the phrase. For example, if the person spun $200 and guessed 'S' for the hidden phrase "Star Wars", the person would win $400 because there were two letters 'S' in the phrase. The winner of the game is the person who guesses the phrase first and they win all the money that they accumulated during the game.

You are going to write a very simple text version of this game. There will only be one player and three dollar amounts ($200, $500 and $1000) that the player will 'spin' before guessing a letter. The player's jackpot (the amount of money they win on each guess) will keep increasing while they guess letters up to a maximum of 6 letter guesses. If they select a letter that is not in the phrase they will not win any money on that turn (but it counts as a turn) and they cannot try to guess the phrase. The game continues until one of the following occurs:

1) the player correctly guesses the phrase and they win their 'jackpot', or
2) the player incorrectly guesses the phrase in which case the game is over, the phrase will be revealed and the player does not win any money, or
3) the player uses their last of 6 letter guesses - the player gets to select no more than 6 letters, and if they cannot guess the phrase after the 6th letter guess then the game is over, the phrase is revealed, and the player does not win any money.

This guessing game is also similar to what is commonly known as hangman:
http://www.printactivities.com/Paper-Games/Rules-For-Hangman.shtml

The `Game` class will have the following attributes:

- The secret phrase (an array of char)
- The disguised phrase (another array of char), in which each unknown letter in the secret phrase is replaced with a blank line (_). For example, if the secret phrase is computer science and the letters c and s have been guessed, the disguised phrase would print as

  C_ _ _ _ _ _ _ SC _ _ _ C_

- The number of guesses made
- The number of incorrect guesses
- The jackpot amount

The `Game` class will have the following methods:

- `spinWheel( )` which returns a random int of 100, 200 or 500
- `guessLetter(c)` guesses that character c (letter c) in the phrase and returns the number of times that letter is in the phrase.
- `GetDisguisedPhrase( )` returns a String containing correctly guessed letters in the correct positions and unknown letters replaced with _.
- `getSecretPhrase( )` returns the secret phrase.
- `increaseJackpot(int amt)` which increases the jackpot by the amount that turn (number of times the guessed letter is in the phrase x the money amount from the spin)
- `getGuessCount( )` returns the number of guesses made.
- `isFound( )` returns true if the hidden phrase has been discovered.
- Appropriate constructor and get and set methods (and any others that make sense).

Implement the `Game` class.

Next write a demo class that demonstrates the game. For this class, first create an array of song titles (these titles can be hardcoded as a String array). Select one song title at random from the array as the secret phrase. The following program shows you how to use java.util.Random to select a phrase at random from an array of song titles.

```java
import java.util.Scanner;
import java.util.Random;
public class RandomWord
{
  public static void main(String[] args)
   {
      String[] songs = {"shake it off", "satisfaction", "mr jones", "uptown
funk"}; //note this is a very short list as an example
      Random rand = new Random();
      int i = rand.nextInt(songs.length);
      System.out.println(songs[i]);
   }
}
```

Note: You can also use the Random class to determine the money value from the spin (200, 500, or 1000).

Some pointers:

1. Keep track of the letters that were guessed (correctly or incorrectly). If the user guesses a letter that they have already guessed, it still decreases their letter guesses. Also, make sure that your program can handle both upper and lower case letters.
2. You should keep a limit (6) on the number of letter guesses, and quit the game if the user doesn't guess the phrase within the limit. After the sixth letter guess, if the user has not able to guess the phrase the game is over and the user loses (wins no money).
3. Recall the conversion options between a char array and a String:

   ```
   String s = new String(phrase);
   will convert a char array phrase into a String s.
   ```

   Similarly,

   ```
       char[] phrase = s.toCharArray();
   ```

   will convert a String s into a char array phrase.

You can use the following template for your main method. You should replace the phrase array of songs to a set of song titles of your choice (use at least 8).

```java
import java.util.Scanner;
import java.util.Random;
public class GameDemo
{
  public static void main(String[] args)
   {
      String[] songs = {"shake it off ", "satisfaction", "mr jones",
     "uptown funk"}; //note this is a very short list as an example
     Random rand = new Random();
      int i = rand.nextInt(songs.length);
      String secretword = songs[i];
```

```
        //continue the code here.



    }
}
```

Here’s how a sample run of what the program would look like:

```
    Welcome to the Guessing Game. The topic is song titles.

    You have 6 guesses. Your puzzle has the following letters:


    _ _   _ _ _ _ _

    Round 1:
    After spinning the wheel, you got $200.
    Please guess a letter: e
    There is/are 1 e.

    _ _   _ _ _ E _

    Do you know the song? (y or n): n

    Your jackpot is $200
    You have 5 guesses left.

    Round 2:
    After spinning the wheel, you got $500.
    Please guess a letter: a
    There is no a.

    _ _   _ _ _ E _

    Do you know the song? (y or n): n

    Your jackpot is $200
    You have 4 guesses left.


    Round 3:
    After spinning the wheel, you got $1000.
    Please guess a letter: s
    There is/are 1 s.

    _ _   _ _ _ E S

    Do you know the song? (y or n): n

    Your jackpot is $1200
    You have 3 guesses left.

    Round 4:
    After spinning the wheel, you got $1000.
    Please guess a letter: a
    Too bad – you already guessed a and there is no a.
```

```
_ _  _ _ _ E S

Do you know the song? (y or n): n

Your jackpot is $1200
You have 2 guesses left.

Round 4:
After spinning the wheel, you got $200.
Please guess a letter: M
There is/are 1 M.

M _  _ _ _ E S

Do you know the song? (y or n): y
What is it: MR JONES

Great Guess! You win $1400!

 Play again (Y/N): Y


Welcome to the Guessing Game. The topic is song titles.

You have 6 guesses. Your puzzle has the following letters:


_ _ _ _ _ _   _ _ _ _

Round 1:
After spinning the wheel, you got $200.
Please guess a letter: n
There is/are 2 n.

_ _ _ _ _ N   _ _ N _

Do you know the song? (y or n): n

Your jackpot is $400
You have 5 guesses left.

Round 2:
After spinning the wheel, you got $500.
Please guess a letter: t
There is/are 1 t.

_ _ T _ _ N   _ _ N _

Do you know the song? (y or n): y

What is it: LOTOWN FUNK

Too bad – wrong song! It is UPTOWN FUNK. You lost $900! Game over.

 Play again (Y/N): N

 Thanks for playing!
```

Question 2
For this question, use the below UML diagram as the basis of a class description. You should implement the class and then create a demo (tester) class to test your program. Make sure you show that all the methods work the way they should.

Movie

- **title : String**
- **publisher : String**
- **year : int**
- **runningTime : double**
- **budget : double**

+ Movie ( ) :
+ Movie (t : String, p : String, y : int, rt : double, b : double) :
+ get methods for all the attributes
+ set methods for all the attributes
+ isLonger (m : Movie) : boolean   //compares this Movie's running time with another Movie's running time
+ isEqual (m : Movie) : boolean   //checks to see if this Movie's attributes are the same as another Movie's
+ copy ( ) : Movie   //creates a copy of this Movie and returns it
+ toString ( ) : String   //returns the title, publisher, and year of the Movie