CIS 1068

Assignment 3

This assignment will demonstrate your knowledge of loops, classes, methods, and Java data types.

You will write a Java program to play the game of *Pico, Fermi, Bagel*. Here are the rules of the game:

The computer will generate a "secret" three digit number at random. The first number will not be 0, and all the digits will be different. The user tries to guess the number. If the user guesses correctly, then the game is over.

If not, the computer gives a hint and the player tries again.

The hints:

- for each digit that matches the secret number in the proper place, the computer prints "Fermi"
- for each digit that matches, but not in the proper place, the computer prints "Pico"
- if none of the digits match, the computer prints "Bagels"

Examples (If the program generated the secret number 482):

guess = 637, *Bagels*

guess = 381, *Fermi*

guess = 382, *Fermi Fermi*

guess = 832, *Fermi Pico*

guess = 328, *Pico Pico*

guess = 428, *Fermi Pico Pico*

guess = 482, *Winner!* (the game is over)

When the game is over, the results are printed: whether the user won or quit, and the number of guesses made

- To avoid making it too easy for the player, you should print all *Fermi*s first, and then the *Pico*s, for each guess
- To avoid making it too difficult, print all guesses and hints to *System.out* so the user can scroll through them

I. The Bagels Class

Your Bagels class will have only one *public* method, *playGame()*. This method will call three other methods to:

1. generate the secret number
2. determine whether the current guess is a winner
3. evaluate the current guess and print hints

Since these methods are called only by a method of the same class – *playGame()* - and not by the test class, it is customary to declare them **private**, instead of **public**. Such methods are sometimes called "utility" methods or "housekeeping" methods since they do "behind the scenes" chores.

OPTIONAL: You may have additional utility methods, also called from playGame(), to get the user's guess and print the game results.

II. High-Level Algorithm for the *playGame()* Method

Generate the Secret Number

do

Get User's Guess (including option to quit)
If (! userQuits)
    Evaluate User's Guess

while (not (winningGuess or userQuits))

Print results – did user win or quit, and number of guesses made

III. Your Main class

Your main class will create a Bagels object and call the method that plays the game. After each game, give the user the option of playing another. After the last game, thank the user. That's all, nothing else.

IV. Additional Specifications

1. The skeleton of the Bagels class you are to use is below. Write the methods bodies and declare instance and local variables as necessary.

2. DO NOT CHANGE THE METHOD DECLARATIONS IN ANY WAY!

3. The user guesses must be entered as a single, 3-digit string. Do not ask the user to enter three separate strings.

V. Upload

Upload a zip file containing both classes.

DUE DATE: October 19th, at 11:59 PM

Skeletal Bagel.java

```
public class Bagel {

    /*
    Starts a new round of the game by generating a new number and asking the user to guess until
    there is a match
    */
    public void playGame(){

    }

    /*
    Generates a secret 3-digit number that the user has to guess.
    */
    private void generateSecretNumber(){

    }

    /*
    Evaluate the user's guess. Return true if correct, false if not correct.
    */
    private boolean isGuessCorrect(String guess) {

    }

    /*
    Print hints to System.out to help the user guess the correct number
    */
    private void printHint(String guess){

    }

}
```