Homework 3

Due: February 27, 11:59 PM

Name: Daniel Carpenter

Student ID: 113009743

Submission Instructions: This Word document contains boxes where you can input your answers. (For example, you can input your name and ID in the fields above.) Please save the document regularly so your answers are not lost. After you complete the assignment, convert it to PDF (File -> Save as Adobe PDF is one way to do this), and upload the PDF file to Gradescope. A link to Gradescope can be found on the left side of the course Canvas page.

Important Note on Academic Integrity: This assignment should be completed individually. In recent semesters, improper collaboration on homework has led to multiple cases of plagiarism, where we receive identical or nearly identical submissions from two or more students. If you decide to discuss this assignment with other students before the deadline, make sure you first read the section of the syllabus on proper and improper collaboration. Additionally, you must include the names of these students below.

Names of Collaborators (if any): None

Question 1 (15 points; 3 per part): Trace the execution of the loops below by filling in the table to the right of each code fragment. If a loop executes infinitely, trace at least three iterations and then write "infinite loop" on the next row. Be sure to store the initial and final values in the table. Note that the tables may contain more rows than you need.

Example:
```
int count = 1;
while (count > 0)
{
    count = count + 1;
}
```

count
1
2
3
4
infinite loop

a)
```
int total = 0;
int count = 0;
while (count <= 2)
{
    total = total + 3 * count;
    count = count + 1;
}
```

total	count
0	0
0	1
3	2
9	3

b)
```
int total = 0;
int count1 = 7;
int count2 = 0;
while (count1 > count2)
{
    total = count1 + count2;
    count1 = count1 - 2;
    count2 = count2 + 1;
}
```

total	count1	count2
0	7	0
7	5	1
6	3	2
5	1	3

c)
```
int total = 0;
for (int x = 1; x != 10; x = x + 5)
{
    total = total - x;
}
```

total	x
0	0
0	5
-5	0

d)
```
int cycle = 0;
for (int idx = 6; idx > 0; --idx)
{
    cycle = (cycle + 1) % 4;
}
```

cycle	idx
0	6
2	5
2	4
2	3
2	2
2	1
2	0

e)
```
int huge = 92486;
int count = 0;
while (count < 0)
{
    huge = huge * 2;
}
```

huge	count
92486	0

Question 2 (10 points): The program shown below has comments at the end of some lines. On these lines, there is a variable that also appears in a row of the table under the program. For each row, fill in the empty columns. See the first row of the table for an example.

In the first empty column, write the role that the variable plays in the program. There are three possible answers: local variable, parameter, or argument. Note that any variable that is passed to a method as an argument will also be either a local variable or parameter of the calling method. For example, consider the following method:

```
public static int exampleMethod(int param)                    // Line 1
{
    int localVar = 42;                                        // Line 2
    return Math.max(param, localVar);                         // Line 3
}
```

On line 1, the variable param is a parameter of the method exampleMethod. On line 2, the variable localVar is a local variable. On line 3, both param and localVar are passed to the method Math.max as arguments. Thus, although the variables param and localVar are a parameter and local variable respectively, on line 3 they both play the role of arguments.

In the second empty column, give the scope of the variable. The scope is the part of the program where the variable can be used. If the variable can only be used in a loop, write the type of loop (e.g., “for loop”). If it can only be used in a conditional statement, write “conditional.” Otherwise, write the name of the method where the variable can be used. In the method above, for example, the scope of both param and localVar are the method exampleMethod.

```
import java.util.Scanner;

public class NumberGame
{

	public static void main(String[] args)
	{
		Scanner keyboard = new Scanner(System.in); // Line 1

		System.out.println("How many times would you like to play?");
		int times = keyboard.nextInt(); //Line 2

		int wins = playMultipleGames(keyboard, times); // Line 3

		System.out.println("You won " + wins * 100 / (double) times
			+ "% of the time");

	}

	// Will return true if the game is won
	public static boolean playOneGame(Scanner input) // Line 4
	{
		int number = (int) (Math.random() * 100) + 1; // a random number between
```

```
                                                        // 1 and 100

        System.out.println("Guess the number between 1 and 100 (inclusive)");
        int guess = input.nextInt(); // Line 5
        int tries = 1;  // because one guess has already occurred
        final int MAX_TRIES = 7;  // You can always do it in this many tries if
                                  //you're
        // Using the right strategy.  That strategy is called binary search and
        // we'll talk about it later in the semester.

        while (guess != number && tries < MAX_TRIES)
        {
              if (guess < number)
              {
                    System.out.println("Your guess was too small");
                    System.out.println("Guess again");
                    guess = input.nextInt();
              }
              else
              {
                    System.out.println("Your guess was too big");
                    System.out.println("Guess again");
                    guess = input.nextInt();
              }

              ++tries;
        }

        // There are two ways to win. You can win by guessing the number
        // in less than MAX_TRIES. Or you can win by guessing the number
        // On the last try
        if ((tries == MAX_TRIES && guess == number) || tries < MAX_TRIES)
        {
              System.out.println("You win!");
              return true;
        }
        else
        {
              System.out.println("The number was " + number);
              System.out.println("You lose.");
              return false;
        }
  }

  // Returns the number of games won
  public static int playMultipleGames(Scanner input, int gameCount) //Line 6, 10
  {
        int wins = 0; // The number of times a game was won--this is returned
        int games = 0;  // The number of games that were played
        while (games < gameCount)
        {
              if (playOneGame(input))   // Line 7
                    // notice this doesn't say == true,
                    // although that's what it means
              {
```

```
                        ++wins; // Line 8
                    }

                    ++games;
                }

                return wins; // Line 9
            }
}
```

Line #	Identifier	Role	Scope
1	System.in	Parameter	Scanner
2	times	Local Variable	Main
3	times	Argument	playMultipleGames
4	input	Parameter	playOneGame
5	guess	Local variable	Conditional
6	input	Parameter	playMultipleGames
7	input	Argument	playOneGame
8	wins	Local Variable	While
9	wins	Parameter	playMultipleGames
10	gameCount	Parameter	playMultipleGames

Question 3 (10 points; 2 per part): The first table below shows the signatures of an overloaded method. For each row in the second table, determine which of the methods in the first table is actually called. If it is impossible to determine the method, write "impossible." If the assignment is illegal, write "illegal."

Hint: You can use eclipse to help you figure this out by creating fake methods with the given signatures and seeing what happens when you try to use them. If you choose to do this, make sure that the rules that Java uses make sense. Remember that you won't have access to eclipse on the exam, so you'll need to know the rules.

Label	Method Signature
A	`void blend(int color1, int color2)`
B	`void blend(int color1, int color2, boolean opaque)`
C	`int blend(int color1, int color2)`
D	`int blend(int color1, double color2)`
E	`double blend(double color1, int color2)`
F	`int blend(double color1, double color2)`

Method Call	Label (A to F) or Impossible or Illegal
`blend(14, 29, false);`	B
`int result = blend(14, 29.3);`	Impossible
`double result = blend(127.0, 63.0);`	F
`blend(14, 29);`	C
`int result = blend(14, 29, true);`	Illegal

Question 4 (15 points; 3 per part): Write the signature for a method that will perform each of the computations described below. **You do not need to write the body of the method.**

Example: Find the average of two grades given as integers.

```
double average(int grade1, int grade2)
```

a) Generate a random double that is between (inclusive) two given integers.

Int randVars(int a, int b)

b) Determine if three characters were given in sorted order (i.e. the first before the second and both the first and second before the third).

char sortChar(char a, char b, char c)

c) Find the maximum of four double values (this is a method that does not exist in the Math class, BTW).

double max(int a, int b, int c, int d)

d) Print a message with a given greeting, name, and punctuation to the console.

String greetingMessage(String greeting, String name, String punctuation)

e) Find the concatenation of String values entered from the keyboard that ends with a given sentinel value. The sentinel value should not be included in the output. The Scanner used to read the values is not constructed in the method.

String concatStrings(String a, String b, String c, SENTINAL)