IT 483
Lab Exercise 04

Due: In Lab, the week of Dec 2nd

Main topics: Programmer Defined Methods
Method Overloading

Exercise

This lab is designed to give you more practice working with programmer defined methods, specifically overloaded methods.

Getting Started To start this exercise, you should

1. Read and understand the Java code below
2. Convert the Java code to C# code

Lab04.Java

```
import java.util.Scanner;

public class Lab04
{
  public static void main(String[] args)
  {
    Scanner stdIn = new Scanner(System.in);

    int a = 1, b = 3, c = 5;
    double x = 2.2, y = 4.4, z = 6.6, ans;

    ans = average(a, b);
    System.out.println("\naverage(a, b) = " + ans);

    ans = average(a, b, c);
    System.out.println("\naverage(a, b, c) = " + ans);

    ans = average(x, y);
    System.out.println("\naverage(x, y) = " + ans);

    ans = average(x, y, z);
    System.out.println("\naverage(x, y, z) = " + ans);
  }

  public static double average(int n1, int n2)
  {
    return (n1 + n2) / 2.0;
  }
```

```
  // Overloaded methoid Definition(s) here ...
}
```

Problem Description

- Read through the existing code to understand what it does and what is left for you to do. Notice that it is incomplete and will not compile as is.
- Your job is to add as many *overloaded* method definitions, for the method `average`, as needed so that the program compiles and makes sense.
- Once you have written your *overloaded* method definitions:
  1. Make sure that your programs compile and run without errors or warnings.
  2. Run your program enough times to check all the choices for correctness.
- Next, answer the following questions, modifying your program file as needed:
  1. Do you really need the `int` *parameter* version(s) of `average`, and why?
  2. Do you really need the three *parameter* version of `average`, i.e. is `average(average(a, b), c) == average(a, b, c)`, and why?
  3. Is `average(1, 2.0, 3)` legal, and if so which version is used, and why?
-