NAME: _____

Problem solving with loops.
Reading: Gaddis Chap (4.2, 4.3 (review)), 4.4, 4.5, 4.6, 4.7, 4.1

Objectives

At the end of this exercise, students will:

- Practice with all three kinds of loops
- Practice tracing loop problems
- Practice creating loop solutions
- Reinforce method structure

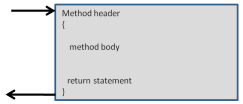
Getting ready

1. All team members - record your notes on your own response sheet.
2. Hand in for today is the exit pass and the notes sheet from one of the members of the team. It will be returned on Wednesday. List the names of all team members present on the back of the exit pass.

Manager – You will keep the group on track. Each person should do the exercise independently, then have the team compare notes. NOTE, for all but the tracing exercises, you may have different yet completely correct answers.

Part 1 - Reinforce method structure

A method has one entry point (its header that contains the return type, name, and parameter list). A method should have only one exit point. In structured programming all structures, loops, decisions and methods should have one entry point and one exit point. There are exceptions. One is with the switch statement which requires a break to break out of a particular case.



BOARD: Write the following method. Follow the documentation to decide what it is to do. Make sure that you have a single exit point (return) in your method.

```
/** isOdd determines if a number is an odd number
 *
 * @param number The number to test
 * @return true if the number is odd, false otherwise.
 ******************************************************/
public static boolean isOdd(int number)
{
```

}

Part 2 – Tracing a for loop

Given the code, trace the loop given dOne and dTwo values as shown. Use the worksheet provided in your packet to trace the changing variable values. Write your output in the lines below.

```
1.  /**
2.     This program demonstrates a method that
3.     accepts two arguments.
4.  */
5.  public class TwoArgs2
6.  {
7.     public static void main(String[] args)
8.     {
9.        double dOne;
10.       double dTwo;
11.       double total;
12.
13.       dOne = 1.5;
14.       dTwo = 3.1;
15.       total = 0;
16.       for (int ii = 0; ii <= 3; ii++)
17.       {
18.         if (ii <= dOne)
19.             total = total + getSum(dOne, dTwo);
20.          else
21.             total = total + getSum(total, dOne);
22.       }
23.       System.out.println("The total is: " + total);
24.    }
25.    /**
26.       The getSum method displays the sum of two numbers and returns the value.
27.       @param num1 The first number.
28.       @param num2 The second number.
29.       @return The sum of the two numbers.
30.    */
31.    public static double getSum(double num1, double num2)
32.    {
33.       double sum;    // To hold the sum
34.       System.out.println("The two values are: " + num1 + " and " + num2);
35.       sum = num1 + num2;
36.       return sum;
37.    }
38. }
```

BOARD Write your output here:

Practice writing looping methods. You may discuss the design with your team before writing the code.

For each of the following problems, design and write a method that will accomplish the task.

1. Write a method to accomplish Programming Challenges number 4 on page 242(puzzle) or 267(watermelon) in Gaddis. The parameter of your method, pennies4Pay, should be the number of days as an int and should return a double representing the number of dollars earned. (Example, if days is 1 the method should return .01) Don't worry about Java's approximation of floating point values.
Write three test cases here that you would use to test your method.

days	amount

```
public static double pennies4Pay(int days)
{
```

```
}
```

2. For the method in Step 1, use a different loop to accomplish the same task (in other words, if you used a for loop, use a while loop, if you used a while loop, use a for.

```
public static double pennies4Pay(int days)
{
```

```
}
```


3. Nested loops (use for loops). Write a method that will print the multiplication table from 0 to the number in the parameter, `limit`. Make it look like a multiplication table with appropriate row and column headings.

```
public static void printMultTable(int limit)
{
```

```
}
```

Tracing worksheet

main method variables					getSum method variables		
dOne	dTwo	total	ii		num1	num2	sum