CIS 1068: Quiz 7 **Name(print)_____Solution_____**

1. **Tracing programs (1 point each value)**: For each snippet of Java code on the left, write down the value of the variable *x* after the code is finished executing. *(Hint: none of these contains an error.)*

Body of `main` method	Method definition	Value of x at end of `main` in LEFT COLUMN

int x = 25;
updateN(x);
	public static void updateN(int n){
	n = n + 2;
	}
		x = 25

int x = 5;
x = cube(x);
	public static int cube(int x) {
	int y = x * x * x;
	return y;
	}
		x =125

int [] x =
{2, -5, 7, 9};
updateX(x);
	public static void updateX(int [] x){
	x[2] = 3;
	x[3] = x[3] - 7;
	}
		x ={2,-5, 3, 2}

Mystery m1 =
new Mystery(2,7);
 	
int d1 = m1.getDist(9);
 	
int d2 = m1.getDist(3);
 	
Mystery m2 =
new Mystery(6,4);
 	
int d3 = m2.getDist(1);
	public class Mystery {
	private int n, x;
	public Mystery (int a,int b) {
	n = Math.min(a, b);
	x = Math.max(a, b);
	}
	public int getDist(int num) {
	if(num > x) {
	return x + num;
	} else if(num >= n) {
	return (n+x) / 2;
	}
	else {
	return Math.min(-1,-num);
	}
	}
	}
		m1 = (2,7)
		
		d1=16
		
		d2=4
		
		m2 = (4,6)
		
		d3 = -1

2. **Short program (12 points)**: Write a definition of the Adder class below so that the code in the Arithmetic class displays the correct sums.

```
-----------------------------------------------------------------
public class Arithmetic {
        public static void main(String [] args)
        {
                Adder a1 = new Adder(3);
                System.out.println("3 + 4 = " + a1.add(4));
                System.out.println("3 + 5 = " + a1.add(5));

                Adder a2 = new Adder(7);
                System.out.println("7 + 8 = " + a2.add(8));
                System.out.println("7 + 2 = " + a2.add(2));
        }
}
-----------------------------------------------------------------
// define your Adder class here
// include: a field for an int, a constructor and an add method
```

```
public class Adder
{
private int value;
public Adder(int n){ value = n;}
public int add(int n){return value + n;}
}
```

3. **Writing a short method (10 points)**: Consider the following Date class. Each Date object represents a calendar date such as September 19th. Write a method named compareTo method that accepts another Date as a parameter and compares them to see which comes first in chronological order. It returns an integer with the following value:
   a) -1 if the date represented by this Date comes before that of the parameter
   b) 0 if the two Date objects represent the same month and day
   c) 1 if the date represented by this Date comes after that of the parameter

For example, if the following Date objects are declared in client code:

```
Date sep19 = new Date(9, 19);
Date temp = new Date(9, 19);
Date sep11 = new Date(9, 11);
```

The following boolean expressions should all produce a true result.

```
sep19.compareTo(sep11) > 0
sep11.compareTo(sep19) < 0
temp.compareTo(sep19) == 0
```

```
public class Date {
private int month;
private int day;
public Date(int m, int d) {
        month = m;
        day = d;
}
public int getDay() {
        return day;
}
public int getMonth() {
        return month;
}
public int numDays(int month) {
        if (month == 2) {
        return 28; // ignore leap years
        } else if (month == 4 || month == 6 || month == 9 || month == 11) {
        return 30;
        } else {
        return 31;
        }
}
// your method would go here
public int compareTo(Date x){
        if(month>x.getMonth()|| month==x.getMonth()&&day>x.getDay())
                return 1;
        else if (month==x.getMonth() && day==x.getDay())
                return 0;
        else
                return -1;
}
}
```