1. Read the following code and answer the questions.

```java
public class Question1 {
    static String s1, s2, s3, s4;
    public static void main(String[] args){
        s1 = new String("Rose");
        s2 = new String("Jasmine");
        s3 = new String("Tulip");
        s4 = new String("Lilac");
        flowers(s1, s3);
        shuffles(s2, s4);
    }
    private static void flowers(String a1, String a2){
        System.out.println(a1.equals(a2)); //---------(a)
        if(a1.compareTo(a2)<0)
            System.out.println(a2); //---------------(b)
        else
            System.out.println(a1);
    }
    private static void shuffle(String str1, String str2){
        String b1 = str1;
        str2 = str1;
        str1 = new String("Dahlia");
        System.out.println(b1 == str2); //-------------(c)
        System.out.println(str2); //------------------(d)
        System.out.println(b1); //--------------------(e)
    }
}
```

(a) What is the output at line (a)?
false (s1 and s2 have different values stored in them)

(b) What is the output at line (b)?
Tulip ("Rose" is lexicographically smaller than "Tulip")

(c) What is the output at line (c)?
true (b1 = s2; s4 = s2; Since both b1 and s4 are assigned s2, they are aliases of s2. So they are equal by ==)

(d) What is the output at line (d)?
Jasmine (It is s2 that is assigned a new value. Not s4)

(e) What is the output at line (e)?
Jasmine (Again, it is s2 that is assigned a new value, not b1)
2. Complete each part before moving on to the next part.

(a) Write a class Fraction with the following specifications:
   Two instance variables: numerator and denominator that can be stored as integers.
   A no argument constructor.
   ```java
   public class Fraction{
       private int numerator;
       private int denominator;

       public Fraction(){
           numerator = 0;
           denominator = 1;
       }
   }
   ```

(b) Add a two argument constructor that will accept two positive integers, one for numerator, and one
   for denominator, and initialize the instance variables. Check that denominator is not zero. If it is,
   print an error statement and exit the program.
   Write two accessor (get) methods for each of the instance variables.
   ```java
   public Fraction(int numer, int denom){
       numerator = numer;
       if(denom == 0){
           System.out.println("Error: zero for demon is not valid! ");
           System.exit(1);
       }
       denominator = denom;
   }

   public int getNumerator(){
       return numerator;
   }

   public int denominator(){
       return denominator;
   }
   ```

(c) Write a method method "add" that accepts another fraction object and adds to the current object
   and returns a new fraction. a is numerator of this object
   b is denominator of this object c is numerator of the other object
   d is denominator of the other object
   New fraction’s numerator is: a * d + b * c
   New fraction’s denominator is: b * d
   ```java
   public Fraction add(Fraction f){
       a = this.numerator;
       b = this.denominator;
       c = f.numerator;
       d = f.denominator;
       if(d == 0){
           System.out.println("Error: f.denominator is invalid!");
       }
       ```
System.exit(1);
}
/* You could also do this way: */
/* int numer = a*d + b*c; */
/* int denom = b*d;*/
/* return new Fraction(numer, denom); */
return new Fraction(a*d + b*c, b*d);

(d) Write a toString method which is a String representation of this class.

public String toString(){
    return "Numerator = " + numerator + "\n denominator = " + denominator;
}

(e) Write an equals method that compares this fraction with the other fraction

public boolean equals(Fraction otherFraction){
    return (this.numerator == otherFraction.numerator &&
            this.denominator == otherFraction.denominator);
}
3. Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title(header) lines. Each block should be a short function of only a few lines.

```java
public class P3{
    public static void main(String args[]){
        String s = "Hello";
        int[][] a = {{1, 2, 3, 4}, {0, 1, 2, 3}};
        int[] b = {3, 1, 4, 1}
        int x = 1, y = 2;
        // (a) Return true if at least one of x and y is positive. Here yes is printed.
        if(positive(x, y)) System.out.println("Yes");
        // (b) Return the sum of the first row. Here 10 is printed.
        System.out.println(rowSum(a));
        // (c) Return the smallest element. Here 1 is printed.
        System.out.println(smallest(b));
        // (d) Remove the first letter; Here ello is printed.
        System.out.println(removeFirst(s));
        // (e) Insert an X at the specified position. Here heXllo is printed.
        s = addX(s, 2);
        System.out.println(s);
    }}

(a) private static boolean positive(int x, int y)
{
    return (x > 0 || y > 0);
}

(b) private static int rowSum(int[][] m)
{
    int sum = 0;
    for(int i = 0; i < m[0].length; i++) {
        sum += m[0][i]
    }
    return sum;
}

(c) private static int smallest(int[] n)
{
    int min = n[0];
    for(int i = 1; i < n.length; i++){
        if(min > n[i])
            min = n[i];
    }
    return min;
}
```
if(n[i] < min)
    min = n[i];
}
return min;
}

(d) private static String removeFirst(String str)
{
    return str.substring(1);
}

(e) private static String addX(String str, int n)
{
    return str.substring(0,n) + "X" + str.substring(n);
}
4. Suppose there are $n$ people attending a party. If each person shakes hands with everyone else, how many handshakes occur in the party? Write a complete Java application to find the total number of handshakes. Your program can have either a single main method or a main method and a small private method.

```java
import java.util.Scanner;
public class P4{
    public static void main(String[] args){
        int n;
        Scanner input = new Scanner(System.in);
        System.out.println("Enter number of people at the party: ");
        n = input.nextInt();
        System.out.println(handShakes(n));
        input.close();
    }

    private static int handShakes(int x){
        if (x == 2) return 1;
        return x-1 + handshakes(x-1);
    }
}
```

You could also write function without recursion.