START WRITING FROM HERE

10. Exercises on command line arguments, Recursion

10.1 Arithmetic (command line arguments)

Write a program called Arithmetic that takes three command line arguments: two integers followed by an arithmetic operator (+, -, *, /). The program shall perform the corresponding operation on the 2 integers and print the result.

```
public class Arithmetic {
    public static void main(String args[]){
        int operand1, operand2;
        char theOperator;

        if (args.length != 3) {
            System.err.println("Usage: java Arithmetic int1 int2
                                op");
            return;
        }

        operand1 = Integer.parseInt(args[0]);
        operand2 = Integer.parseInt(args[1]);
        theOperator = args[2].charAt(0);
        System.out.print(args[0] + args[2] + args[1] + "=");

        switch (theOperator) {
            case ('-'): System.out.print(operand1 - operand2); break;
            case ('+'): System.out.print(operand1 + operand2); break;
            case ('*'): System.out.print(operand1 * operand2); break;
            case ('/'): System.out.print(operand1 / operand2); break;
            default : System.err.println("Error: invalid operator!");
        }
    }
}
```

output:

```
java Arithmetic 3 2 +
3+2=5
```

ROLL NUMBER :

10.2 Factorial Recursive

write a recursive method called factorial() to compute the factorial of the given integer.

```
public class FactorialRecursive {
    public static void main (String[] args) {
        System.out.println (factorial (10));
    }
    public static int factorial (int n) {
        return (n==0) ? 1 : n * factorial (n-1);
    }
}
```

output : 3628800

10.3 Fibonacci (Recursive)

write a recursive method to compute Fibonacci num of n.

```
public class FibonacciRecursive {
    public static void main (String[] args) {
        System.out.println (fibonacci (10));
    }
    public static int fibonacci (int n) {
        if (n==0) { return 0; }
        else if (n==1) { return 1; }
        else {
            return fibonacci (n-1) + fibonacci (n-2);
        }
    }
}
```

output : 55

ROLL NUMBER :

10.4 Length of a Running Number Sequence (Recursive)

A special number sequence is defined as :

S(1) = 1

S(2) = 12

S(3) = 123

S(10) = 12345678910

S(11) = 1234567891011

write a recursive method to compute the length of S(n).

```
public class RunningNumber {
    public static void main(String[] args) {
        System.out.println(len(13));
    }

    public static int len(int n){
        if (n==1) { return 1; }
        else {
            return len(n-1) + numOfDigits(n);
        }
    }

    public static int numOfDigits(int num){
        int count = 0;
        for(int i = num; i>0 ; i/=10) { count++;}
        return count;
    }
}
```

output : 17

ROLL NUMBER :

10.5 GCD (Recursive)

Write a recursive method called gcd() to compute the greatest common divisor of two given integers.

```
public class GCD{
    public static void main(String[] args){
        System.out.println(gcd(40,16));
    }
    public static int gcd(int a, int b){
        if (b==0){
            return a;
        }
        else if (b>0){
            return gcd(b,a%b);
        }
        else{
            return -1;
        }
    }
}
```

output : 8