**1)**

Get two numbers that represent a given range + (Pattern) Output max five numbers per line [c]

(The five numbers are smallest → largest. Normally done with nested for loops but not in this case.)

```
static void calcOptionC() {

    int m = 0;
    int n = 0;
    int largeNum = 0;
    int smallNum = 0;
    int total = 0;
    int counter = 1;

    Scanner keyboard = new Scanner(System.in);

    System.out.print("Enter the m value: "); //user enters the m value
    m = keyboard.nextInt();
    System.out.print("Enter the n value: "); //user enters n value
    n = keyboard.nextInt();

    if (m > n) { //assigns the entered m/n values to smallest number or largest number
        largeNum = m;
        smallNum = n;
    } else {
        largeNum = n;
        smallNum = m;
    }
                small → Large

    while (smallNum <= largeNum) { //Calculate and display numbers inbetween

        if (counter % 5 == 0) { //Work out if there are 5 numbers are in one line yet
            System.out.println(" " + smallNum);
        } else {

            System.out.print(" " + smallNum);
        }

        if (smallNum % 2 == 0) { //Find even number
            total = total + 0; //Does nothing but need it in order to find odd number

        } else { //Find odd number
            total = total + smallNum;
        }

        smallNum++; //Iteration to find the next to output
        counter++; //Keep track of numbers per line
    }

    System.out.println();
    System.out.print("Total odd number is: ");
    System.out.print(total);
```

Get a single number + Determine prime number [E]

```
static void calcOptionE() {

    int num = 0;
    boolean primeNumber = true;

    Scanner keyboard = new Scanner(System.in);

    System.out.print("Enter a number: "); //Get user input
    num = keyboard.nextInt();

    for(int counter = 2; counter < num; counter++) { //Work out whether or not input is prime number

        if(num % counter == 0) {
            primeNumber = false;
        }
    }

    if(primeNumber) { //Check if prime number and display message accordingly
        System.out.printf("The number: %d is a prime number", num);
    } else {
        System.out.printf("The number: %d is NOT a prime number", num);
    }

    System.out.println();
}
```

Get only 3 integers + Determine the smallest number + largest number [b]

```
static void calcOptionB() {

        double x = 0;
        double y = 0;
        double z = 0;
        double largeNum = 0;
        double smallNum = 0;

        Scanner keyboard = new Scanner(System.in);

        System.out.println("Enter first number: "); //user enters first number
        x = keyboard.nextDouble();

        System.out.println("Enter second number: "); //user enters second num
        y = keyboard.nextDouble();

        System.out.println("Enter third number: "); //user enters third num
        z = keyboard.nextDouble();

        largeNum = x; //NOTE: WHY I didn't put double in front of largeNum to declare? Because conventionally in Java do it this way
        smallNum = x; //assign the inital smallest number and largest number to compare

       if (y > largeNum) {
            largeNum = y;
       }
       if (z > largeNum) {
            largeNum = z;
       }


       if (y < smallNum ) {
            smallNum = y;
       }
       if (z < smallNum) {
            smallNum = z;
       }



        System.out.printf("Largest num: %f Smallest num: %f", largeNum, smallNum); //output largest number and smallest number
        System.out.println();
    }
```

How to use switch cases??

**3)**

Get a string + determine if there are repeat characters in the string + Exit loop without Break OR Return


```
start

Scanner keyboard = new Scanner(System.in);

System.out.print("Enter a string: ");
userString = keyboard.nextLine();

userStringLength = userString.length(); //Get string length

    for (int i = 0; i < userStringLength && repeatedChar == false; i++) { //Do the loop until fullString length or a repeated char is found

        userStringChar = userString.charAt(i); //Get the initial character
        newI = i + 1; //Have to do it //Get the second initial character

        for (int j = newI; j < userStringLength && repeatedChar == false; j++) { //Loop compare current character with all the following character
            charLoop = userString.charAt(j);

            if (userStringChar == charLoop) { //If current character matches one of the following characters stop the loop
                repeatedChar = true;
            }
        }
    }


if (repeatedChars) {
        System.out.println("There are repeated characters"); //Display result
    } else {
        System.out.println("There are NO repeated characters");
    }

End
```