Reporte de Práctica #3

Loops

Ingeniería Informática

Joselyn Gallegos Abreo

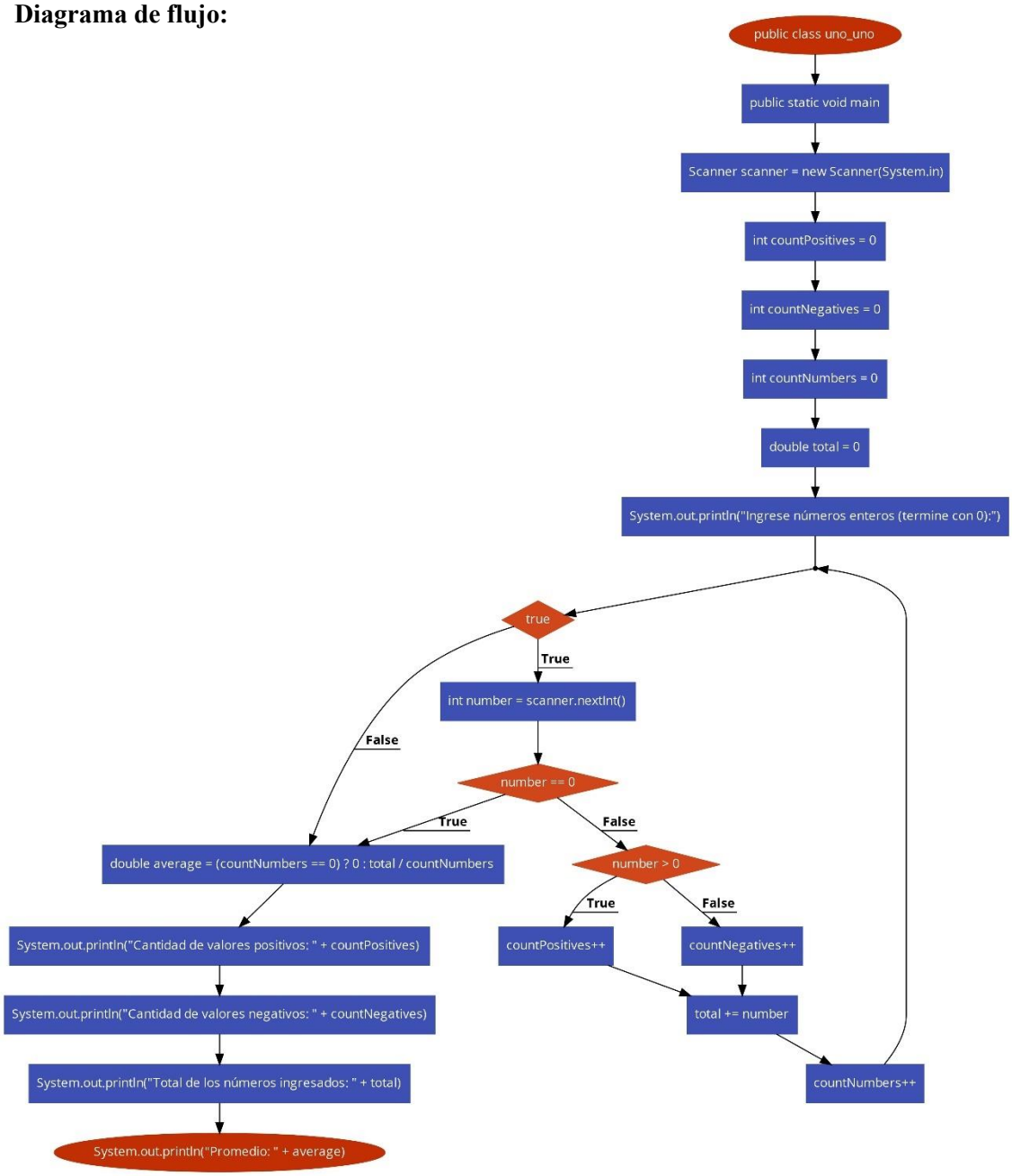
Expediente 285802

2025-1

- **Programming Exercises**

1. (Counting positive and negative numbers and computing the average of numbers) Write a program that reads an unspecified number of integers, determines how many positive and negative values have been read, and computes the total and average of the input values (not counting zeros). Your program ends with the input 0. Display the average as a floating-point number.

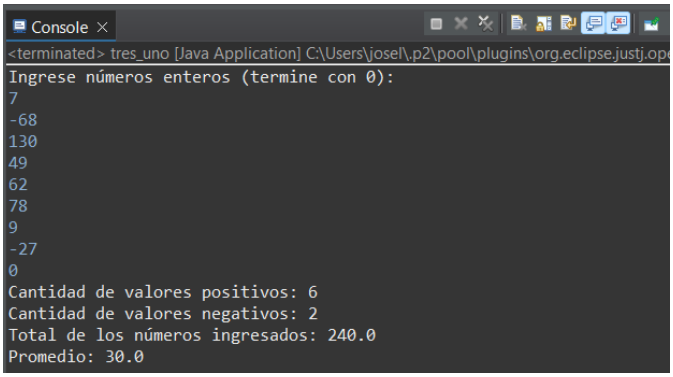
Diagrama de flujo:



Código:

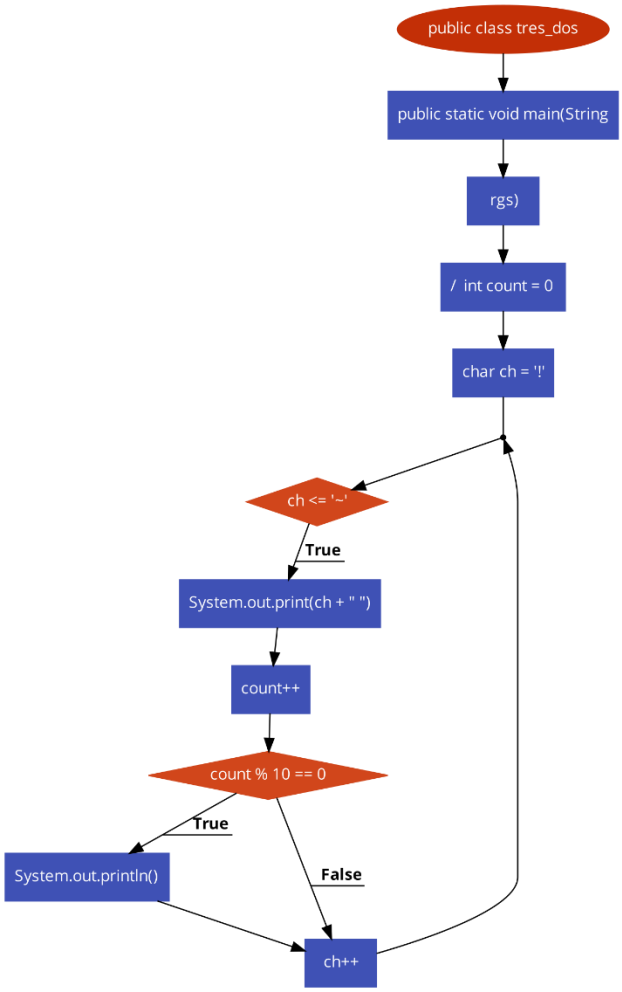
```
package prueba;
import java.util.Scanner;
public class tres_uno {
    public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner scanner = new Scanner(System.in);
     int countPositives = 0;
     int countNegatives = 0;
     int countNumbers = 0;
     double total = 0;
     System.out.println("Ingrese números enteros (termine con 0):");
     while (true) {
        int number = scanner.nextInt();
        if (number == 0) {
            break;
        }
        if (number > 0) {
            countPositives++;
        } else {
            countNegatives++;
        }

        total += number;
        countNumbers++;
     }
     double average = (countNumbers == 0) ? 0 : total / countNumbers;
     System.out.println("Cantidad de valores positivos: " + countPositives);
     System.out.println("Cantidad de valores negativos: " + countNegatives);
     System.out.println("Total de los números ingresados: " + total);
     System.out.println("Promedio: " + average);
    }
}
```

Pantalla:



2. (Displaying the ASCII character table) Write a program that prints the characters in the ASCII character table from '!' to '~'. Print then characters per line.

Diagrama de flujo:



Código:

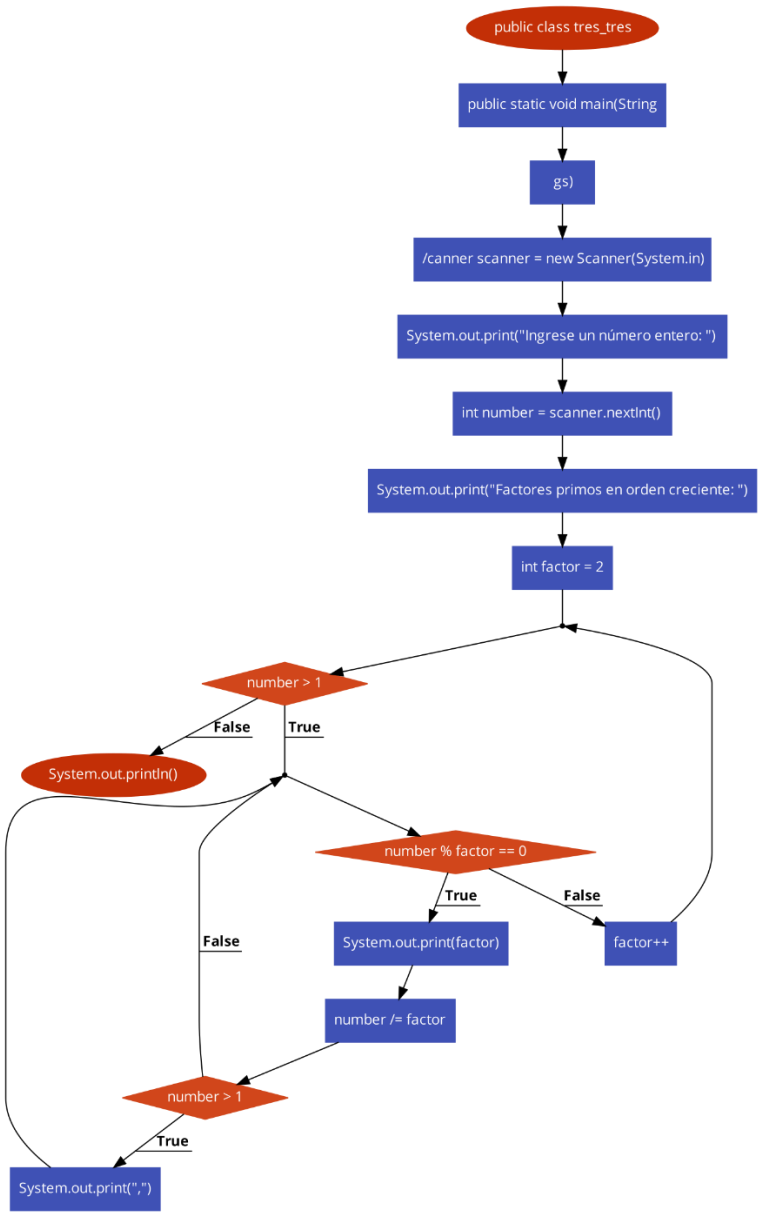
```
package prueba;
public class tres_dos {
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                  int count = 0;
            for (char ch = '!'; ch <= '~'; ch++) {
               System.out.print(ch + " ");
               count++;
               if (count % 10 == 0) {
                  System.out.println();
        }
            }
        }
}
```

Pantalla:

```
Console ×
<terminated> tres_dos [Java Application] C:\Users\josel\.
! " # $ % & ' ( ) *
+ , - . / 0 1 2 3 4
5 6 7 8 9 : ; < = >
? @ A B C D E F G H
I J K L M N O P Q R
S T U V W X Y Z [ \
] ^ _ ` a b c d e f
g h i j k l m n o p
q r s t u v w x y z
{ | } ~
```

3. (Finding the factors of an integer) Write a program that reads an integer and displays all its smallest factors in increasing order. For example, if the input integer is 120, the output should be as follows: 2,2,2,3,5.

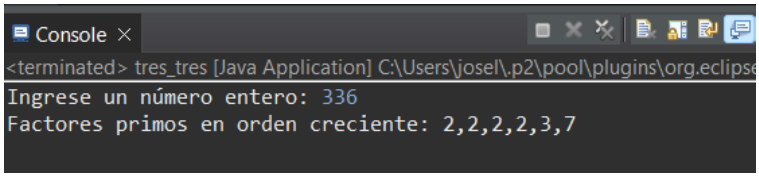
Diagrama de flujo:



Código:

```
package prueba;
import java.util.Scanner;
public class tres_tres {
    public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner scanner = new Scanner(System.in);
     System.out.print("Ingrese un número entero: ");
     int number = scanner.nextInt();
     System.out.print("Factores primos en orden creciente: ");
     int factor = 2;
     while (number > 1) {
        while (number % factor == 0) {
           System.out.print(factor);
           number /= factor;
           if (number > 1) {
              System.out.print(",");
           }
        }
        factor++;
     }

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

Pantalla:

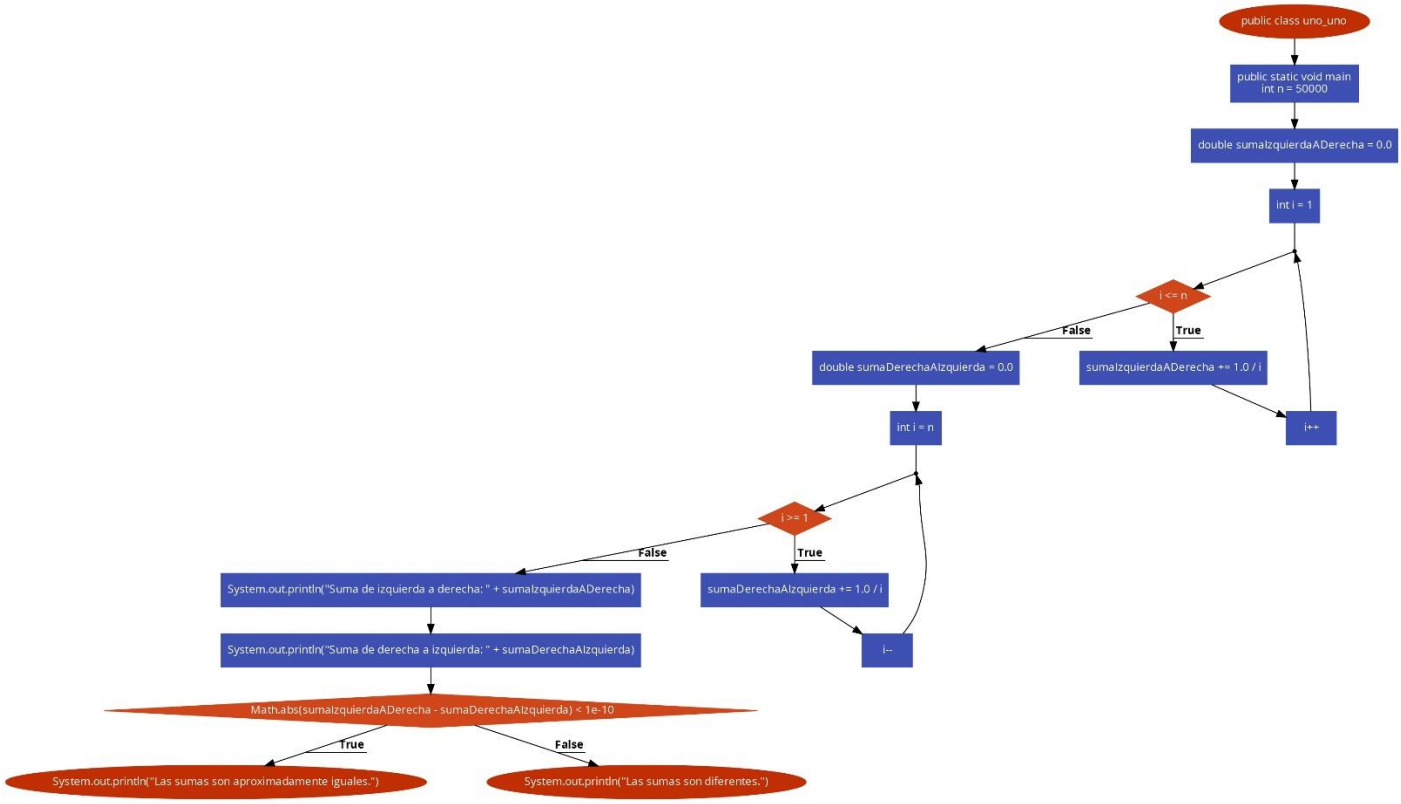


4. (Obtaining more accurate results) In computing the following series, you will obtain more accurate results by computing from right to left rather than from left to right:

$$1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

Write a program that compares the results of the summation of the preceding series, computing from left to right and from right to left with n=50000

Diagrama de flujo:



Código:

```
package prueba;
public class tres_cuatro {
    public static void main(String[] args) {
            // TODO Auto-generated method stub
            int n = 50000;
     double sumaIzquierdaADerecha = 0.0;
     for (int i = 1; i <= n; i++) {
        sumaIzquierdaADerecha += 1.0 / i;
     }
     double sumaDerechaAIzquierda = 0.0;
     for (int i = n; i >= 1; i--) {
        sumaDerechaAIzquierda += 1.0 / i;
     }
     System.out.println("Suma de izquierda a derecha: " + sumaIzquierdaADerecha);
     System.out.println("Suma de derecha a izquierda: " + sumaDerechaAIzquierda);

     if (Math.abs(sumaIzquierdaADerecha - sumaDerechaAIzquierda) < 1e-10) {
        System.out.println("Las sumas son aproximadamente iguales.");
     } else {
        System.out.println("Las sumas son diferentes.");
     }
     }
}
```

Pantalla:

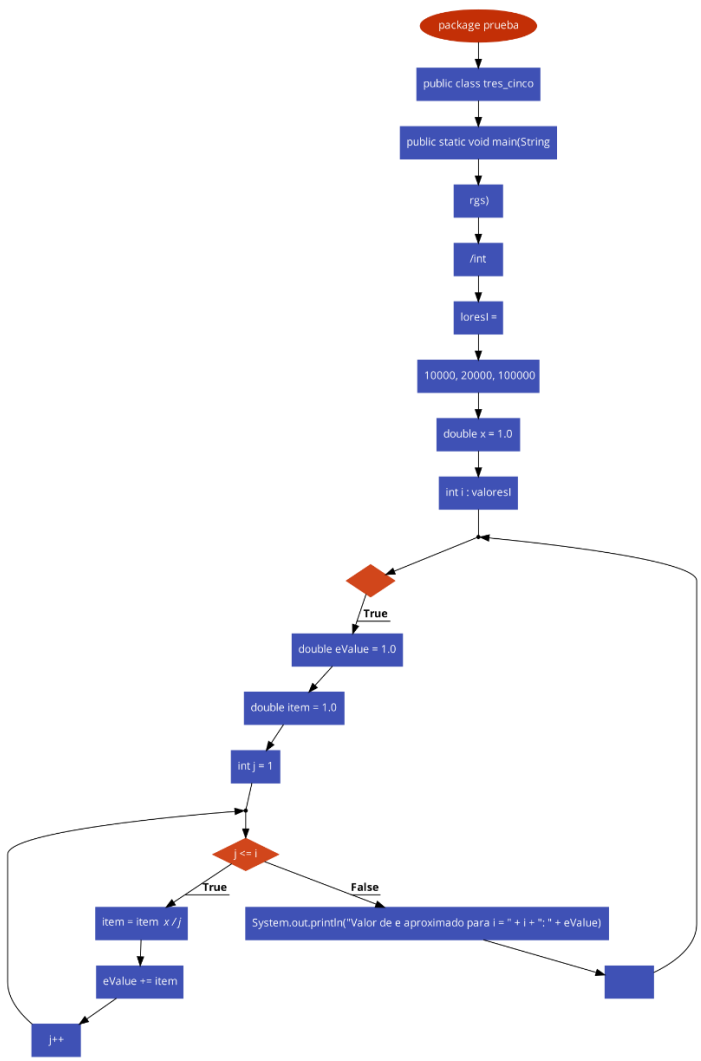
```
Console ×
<terminated> tres_cuatro [Java Application] C:\Users\josel\.p2\pool\plugins\org.ecli
Suma de izquierda a derecha: 11.397003949278504
Suma de derecha a izquierda: 11.397003949278519
Las sumas son aproximadamente iguales.
```

5. (Computing e) You can approximate e using the following series:

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^i}{i!}$$

Write a program that displays the e value for i=10000, 20000, and 100000
(Hint: since e and item to be 1 and keep adding a new item to e. The new item is the previous item divided by i for i=2,3,4…

Diagrama de flujo:



Código:

```
package prueba;
public class tres_cinco {
     public static void main(String[] args) {
                // TODO Auto-generated method stub
                int[] valoresI = {10000, 20000, 100000};
            double x = 1.0;
       for (int i : valoresI) {
          double eValue = 1.0;
          double item = 1.0;
     for (int j = 1; j <= i; j++) {
            item = item * x / j;
            eValue += item;
          }
          System.out.println("Valor de e aproximado para i = " + i + ": " + eValue);
       }
   }
     }
```

Pantalla: