
	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 1

INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	ESTRUCTURA DE DATOS Y ALGORITMOS				
TÍTULO DE LA PRÁCTICA:	LABORATORIO 1: ESTRUCTURA DE DATOS Y ALGORITMOS				
NÚMERO DE PRÁCTICA:	1	AÑO LECTIVO:	2025 – A	NRO. SEMESTRE:	Tercero III
FECHA DE PRESENTACIÓN	17/05/2025	HORA DE PRESENTACIÓN	20:00		
INTEGRANTE (s): Santa Cruz Villa Diego Sebastián				NOTA:	
DOCENTE(s): <ul style="list-style-type: none"> Mg. Ing. Rene Alonso Nieto Valencia. 					

REPOSTORIO GITHUB: <https://github.com/DiegoSantaC/LaboratoriosEDA>

SOLUCIÓN Y RESULTADOS
<p>I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS</p> <p><u>EJERCICIOS RESUELTOS:</u></p> <p>1.- Edad promedio, mayor y menor de un grupo:</p> <pre> public class EjerciciosResueltos { public static void main(String[] args) { Scanner scanner = new Scanner(System.in); System.out.print("Ingrese el número de personas: "); int n = scanner.nextInt(); int[] edades = new int[n]; System.out.println("Ingrese las edades:"); for (int i = 0; i < n; i++) { edades[i] = scanner.nextInt(); } int suma = 0, mayor = edades[0], menor = edades[0]; for (int edad : edades) { suma += edad; if (edad > mayor) mayor = edad; if (edad < menor) menor = edad; } double promedio = (double) suma / n; System.out.println("Edad promedio: " + promedio); System.out.println("Edad mayor: " + mayor); System.out.println("Edad menor: " + menor); scanner.close(); //----Ejercicio Propuesto 1 } } </pre>

```
--- exec:3.1.0:exec (default-cli) @ Laboratorio1EDA ---
Ingrese el número de personas: 5
Ingrese las edades:
12
19
20
15
14
Edad promedio: 16.0
Edad mayor: 20
Edad menor: 12
-----
```

2.- Suma de los N numeros naturales:

```
public static void SumaNaturales() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Ingrese un número N: ");
    int n = scanner.nextInt();
    int suma = 0, contador = 1;
    while (contador <= n) {
        suma += contador;
        contador++;
    }
    System.out.println("La suma de los primeros " + n + " números naturales es: "
        + suma);
    scanner.close();
}
```

```
--- exec:3.1.0:exec (default-cli) @ Laboratorio1EDA ---
Ingrese un número N: 10
La suma de los primeros 10 números naturales es: 55
-----
```

3.- Lista ordenada (si o no):

```
48 public static void listaOrdenada() {
49     Scanner scanner = new Scanner(System.in);
50     System.out.print("Ingrese el número de elementos: ");
51     int n = scanner.nextInt();
52     int[] numeros = new int[n];
53     System.out.println("Ingrese los números:");
54     for (int i = 0; i < n; i++) {
55         numeros[i] = scanner.nextInt();
56     }
57     boolean estaOrdenada = true; // Invariante: Se supone que la lista está ordenada
58     for (int i = 1; i < n; i++) {
59         if (numeros[i] < numeros[i - 1]) {
60             estaOrdenada = false; // Se rompe la invariante si encontramos un desorden
61             break;
62         }
63     }
64     System.out.println("¿Está ordenada la lista?: " + (estaOrdenada ? "Sí" :
65         "No"));
66     scanner.close();
67 }
```

```

graph TD
    Inicio([Inicio]) --> Def1[Definir n, i Como Entero]
    Def1 --> Def2[Definir n, i Como Entero]
    Def2 --> Ingresa[Ingresar el número de...]
    Ingresa --> N[/n/]
    N --> Dim[Dimension lista(n)]
    Dim --> IngresaNota[Ingrese la nota de un...]
    IngresaNota --> ListaI[/lista(i)/]
    ListaI --> NotasIng[Notas ingresadas:]
    NotasIng --> ListaI_2[/lista(i), '/'/]
    ListaI_2 --> Ordenar[Ordenar(lista, n)]
    Ordenar --> NotasOrd[Notas ordenadas:]
    NotasOrd --> ListaI_3[/lista(i), '/'/]
    ListaI_3 --> EscribirMediana[Escribir La mediana es: ...]
    EscribirMediana --> EscribirMedia[Escribir La media es: ...]
    EscribirMedia --> EscribirDesviacion[Escribir La desviación es: ...]
    EscribirDesviacion --> Fin([FinProceso])

```

```
3 class Main {
4     Run | Debug
5     public static void main(String[] args) {
6         Scanner sc= new Scanner(System.in);
7         System.out.print(s:"Ingresar el numero de Estudiantes: ");
8         int [] lista= new int[sc.nextInt()];
9
10        for(int i=0;i<lista.length;i++){
11            System.out.print(s:"Ingrese la nota de un estudiante: ");
12            lista[i]=sc.nextInt();
13        }
14        for(int nota : lista){
15            System.out.print("[ "+nota+" " );
16        }
17        System.out.println("\nLa moda es: "+moda(lista));
18
19        for(int nota: ordenar(lista)){
20            System.out.print("[ "+nota+" " );
21        }
22        System.out.print("\nLa mediana es: "+mediana(ordenar(lista)));
23        System.out.print("\nLa media es: "+media(lista));
24        System.out.print("\nLa desviacion estandar es: "+desviacionEstandar(lista));
25    }
26 }
```

Moda:

```
28 public static String moda(int[] lista) {
29     int maxRepeticiones = 0;
30
31     for (int i = 0; i < lista.length; i++) {
32         int contador = 0;
33         for (int j = 0; j < lista.length; j++) {
34             if (lista[i] == lista[j]) {
35                 contador++;
36             }
37         }
38         if (contador > maxRepeticiones) {
39             maxRepeticiones = contador;
40         }
41     }
42
43     if (maxRepeticiones == 1) {
44         return "No hay moda";
45     }
46
47     String modas = "";
48     boolean[] yaContado = new boolean[lista.length];
49
50     for (int i = 0; i < lista.length; i++) {
51         if (yaContado[i]) continue;
52
53         int contador = 0;
54         for (int j = 0; j < lista.length; j++) {
55             if (lista[i] == lista[j]) {
56                 contador++;
57                 yaContado[j] = true;
58             }
59         }
60     }
```

```
61         if (contador == maxRepeticiones) {  
62             modas += lista[i] + " ";  
63         }  
64     }  
65  
66     return modas.trim();  
67 }
```

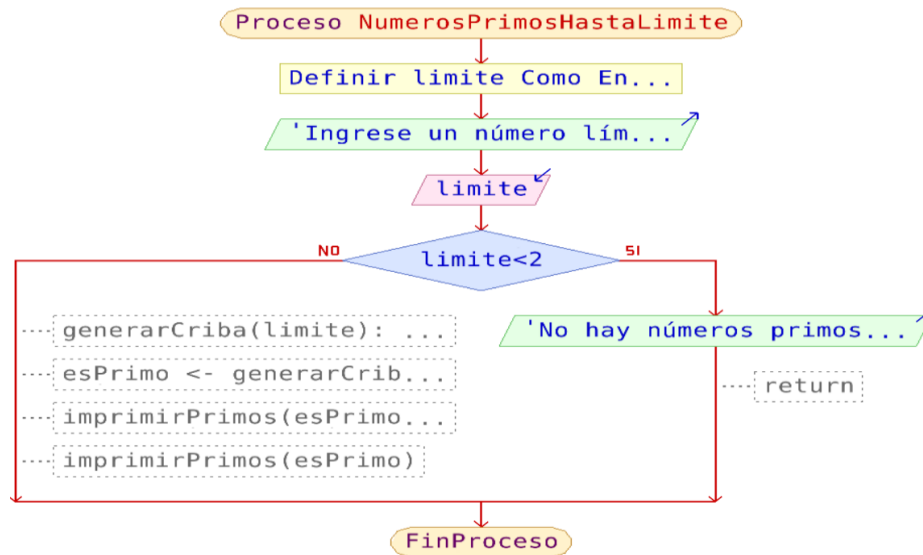
Ordenar --→ Mediana:

```
69     public static int[] ordenar(int[] lista) {  
70         int[] listaOrdenada = Arrays.copyOf(lista, lista.length);  
71         // Ordenamiento por burbuja  
72         for (int i = 0; i < listaOrdenada.length - 1; i++) {  
73             for (int j = 0; j < listaOrdenada.length - 1 - i; j++) {  
74                 if (listaOrdenada[j] > listaOrdenada[j + 1]) {  
75                     int temp = listaOrdenada[j];  
76                     listaOrdenada[j] = listaOrdenada[j + 1];  
77                     listaOrdenada[j + 1] = temp;  
78                 }  
79             }  
80         }  
81  
82         return listaOrdenada;  
83     }  
84  
85     public static double mediana(int[] lista) {  
86         int n = lista.length;  
87         if (n % 2 != 0) {  
88             return lista[n / 2];  
89         } else {  
90             return (lista[n / 2 - 1] + lista[n / 2]) / 2.0;  
91         }  
92     }
```

Media --→ Desviación estándar :

```
94     public static double desviacionEstandar(int[] lista) {  
95         double media = media(lista);  
96         double sumaDiferencias = 0;  
97  
98         for (int nota : lista) {  
99             sumaDiferencias += Math.pow(nota - media, 2);  
100         }  
101  
102         return Math.sqrt(sumaDiferencias / lista.length);  
103     }  
104  
105     public static double media(int[] lista) {  
106         double suma = 0;  
107         for (int nota : lista) {  
108             suma += nota;  
109         }  
110         return suma / lista.length;  
111     }  
112 }
```

2. Implementa un programa en Java que encuentre todos los números primos en un rango definido por el usuario utilizando el algoritmo de la Criba de Eratóstenes



Main:

```

1  import java.util.*;
2
3  class Main {
4      public static void main(String[] args) {
5          Scanner sc=new Scanner(System.in);
6          int limite = sc.nextInt();
7
8          if (limite < 2) {
9              System.out.println("No hay números primos menores que 2.");
10             return;
11         }
12
13         boolean[] esPrimo = generarCriba(limite);
14         imprimirPrimos(esPrimo);
15     }
  
```

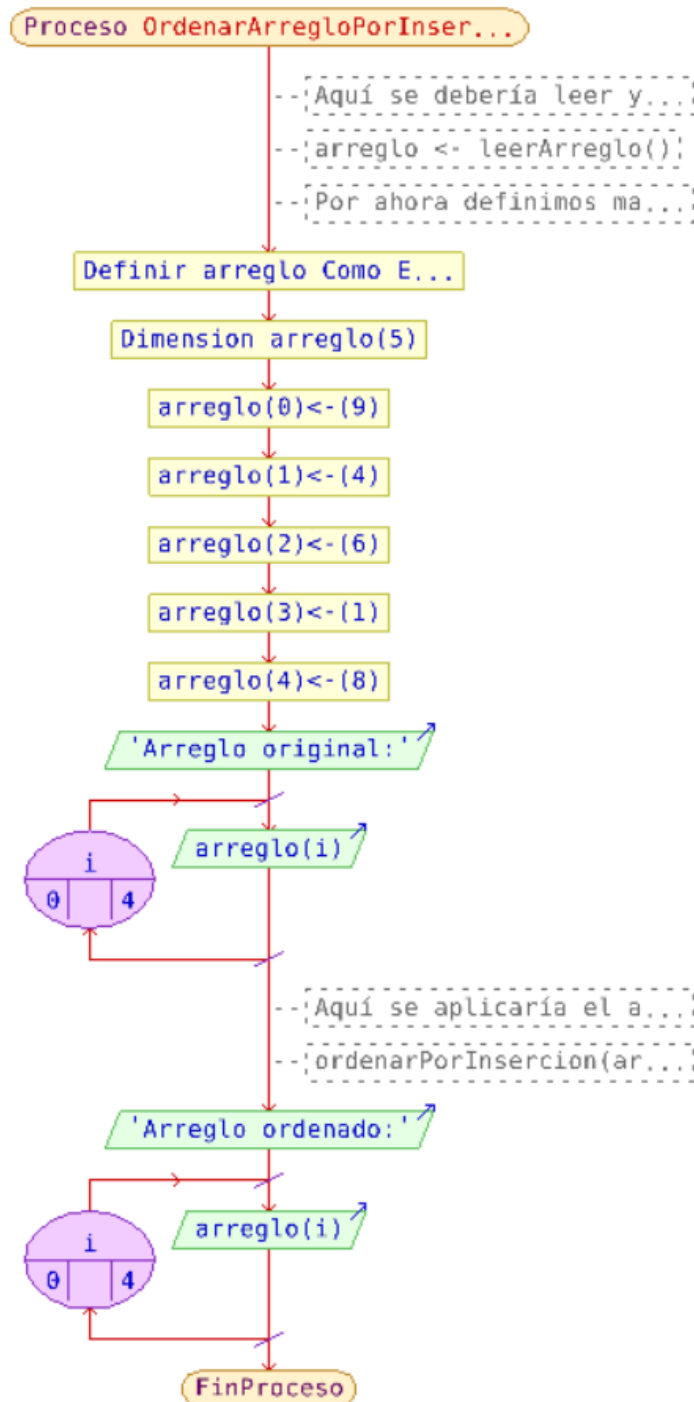
Generar Criba, Imprimir los Números Primos :

```
17 // Método que implementa la Criba de Eratóstenes
18 public static boolean[] generarCriba(int limite) {
19     boolean[] esPrimo = new boolean[limite + 1];
20
21     for (int i = 2; i <= limite; i++) {
22         esPrimo[i] = true;
23     }
24
25     // Aplicar criba
26     for (int i = 2; i * i <= limite; i++) {
27         if (esPrimo[i]) {
28             for (int j = i * i; j <= limite; j += i) {
29                 esPrimo[j] = false;
30             }
31         }
32     }
33
34     return esPrimo;
35 }
36
37 public static void imprimirPrimos(boolean[] esPrimo) {
38     System.out.println(x:"Números primos encontrados:");
39     for (int i = 2; i < esPrimo.length; i++) {
40         if (esPrimo[i]) {
41             System.out.print(i + " ");
42         }
43     }
44 }
45 }
46
```

3. Desarrolla un algoritmo que implemente el Ordenamiento por Inserción, asegurando que en cada paso del bucle el segmento procesado de la lista permanece ordenado (principio de invariante).

Main:

```
1 import java.util.*;
2
3 class Main {
4     Run | Debug
5     public static void main(String[] args) {
6         int[] arreglo = leerArreglo();
7         System.out.println(x:"Arreglo original:");
8         imprimirArreglo(arreglo);
9
10        ordenarPorInsercion(arreglo);
11
12        System.out.println(x:"Arreglo ordenado:");
13        imprimirArreglo(arreglo);
14    }
15 }
```





Leer e imprimir nuestro arreglo:

```
15 // Método para leer el arreglo desde consola
16 public static int[] leerArreglo() {
17     Scanner entrada = new Scanner(System.in);
18     System.out.print(s:"Ingrese la cantidad de elementos: ");
19     int tamaño = entrada.nextInt();
20
21     int[] arreglo = new int[tamaño];
22     for (int i = 0; i < tamaño; i++) {
23         System.out.print("Elemento [" + i + "]: ");
24         arreglo[i] = entrada.nextInt();
25     }
26     return arreglo;
27 }
28
29 // Método para imprimir el arreglo
30 public static void imprimirArreglo(int[] arreglo) {
31     for (int num : arreglo) {
32         System.out.print(num + " ");
33     }
34     System.out.println();
35 }
```

Ordenación por inserción:

```
37 // Método que implementa el ordenamiento por inserción
38 public static void ordenarPorInsercion(int[] arreglo) {
39     for (int i = 1; i < arreglo.length; i++) {
40         int actual = arreglo[i];
41         int j = i - 1;
42
43         while (j >= 0 && arreglo[j] > actual) {
44             arreglo[j + 1] = arreglo[j];
45             j--;
46         }
47
48         arreglo[j + 1] = actual;
49
50         System.out.print("Paso " + i + ": ");
51         imprimirArreglo(arreglo);
52     }
53 }
54
55 }
56
```

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 10</p>

II. SOLUCIÓN DEL CUESTIONARIO

1. ¿Cuáles fueron las dificultades que encontraste al desarrollar los ejercicios propuestos? por ejemplo, poca documentación, complejidad del lenguaje, etc.

Una dificultad fue la falta de información clara sobre algunos algoritmos, como en el caso de la Criba de Eratóstenes, donde fue necesario investigar para entender cómo funcionaba antes de implementarlo. También hubo un reto al calcular la moda, ya que el primer intento solo mostraba una, y se tuvo que adaptar el método para reconocer modas múltiples. Además, al aplicar el ordenamiento por inserción, fue importante mantener el código ordenado y modular, respetando el principio de invariante.

III. CONCLUSIONES

El desarrollo de los tres ejercicios permitió aplicar de forma práctica conceptos fundamentales de la programación en Java, como el manejo de arreglos, ciclos y métodos. Al resolver el ejercicio de moda, mediana y desviación estándar, se afianzaron habilidades para trabajar con datos numéricos, identificar patrones y aplicar operaciones matemáticas de forma lógica. El algoritmo de la Criba de Eratóstenes representó un reto adicional, ya que exigió comprender previamente su funcionamiento teórico. Finalmente, el ejercicio de ordenamiento por inserción permitió visualizar cómo un algoritmo puede mantener un segmento ordenado en cada paso, reforzando el principio de invariante. En conjunto, estas actividades fortalecieron la capacidad de análisis, resolución de problemas y estructuración lógica del código.

RETROALIMENTACIÓN GENERAL

Los ejercicios se resolvieron de manera ordenada y con buena lógica. Pude mejorar el código, como cuando se ajustó el método de la moda para que funcione con varios resultados. También se entendieron bien los pasos de los algoritmos, como la Criba de Eratóstenes y el ordenamiento por inserción. Para seguir mejorando, sería bueno seguir practicando cómo dividir bien el código en partes y buscar formas más fáciles o rápidas de resolver algunos problemas.

REFERENCIAS Y BIBLIOGRAFÍA