UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGON

TAREA

P R E S E N T A

Alexis Hernández Zamudio

APROFESOR

Jesús Hernández Cabrera

Gpo:1558

URL del repositorio:

https://github.com/AlextyrB/Analisis_Algoritmos/tree/main

Ciudad Nezahualcóyotl, EDOMEX. 2 de SEPTIEMBRE del 2025

CODIGO

BusquedaLineal.java

```java
public class BusquedaLineal {

    public static int busquedaLineal(int[] arreglo, int valorBuscado) {  5 usages
        int comparaciones = 0;

        for (int i = 0; i < arreglo.length; i++) {
            comparaciones++;
            if (arreglo[i] == valorBuscado) {
                System.out.println("Elemento " + valorBuscado + " encontrado en la posición: " + i);
                return comparaciones;
            }
        }
        System.out.println("Elemento " + valorBuscado + " no encontrado en el arreglo");
        return comparaciones;
    }
    public static ResultadoBusqueda busquedaLinealCompleta(int[] arreglo, int valorBuscado) {  2 usages
        int comparaciones = 0;

        for (int i = 0; i < arreglo.length; i++) {
            comparaciones++;
            if (arreglo[i] == valorBuscado) {
                return new ResultadoBusqueda(true, i, comparaciones);
            }
        }

        return new ResultadoBusqueda(false, -1, comparaciones);
    }

    static class ResultadoBusqueda {  5 usages
        boolean encontrado;  2 usages
        int posicion;  2 usages
        int comparaciones;  3 usages

        public ResultadoBusqueda(boolean encontrado, int posicion, int comparaciones) {  2 usages
            this.encontrado = encontrado;
            this.posicion = posicion;
            this.comparaciones = comparaciones;
        }
        @Override
        public String toString() {
            if (encontrado) {
                return String.format("Elemento encontrado en posición %d con %d comparaciones",
                        posicion, comparaciones);
            } else {
                return String.format("Elemento no encontrado después de %d comparaciones",
                        comparaciones);
            }
        }
    }

    public static void main(String[] args) {
        int[] arreglo1 = {10, 25, 3, 47, 15, 8, 92, 33};
        int[] arreglo2 = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int[] arregloVacio = {};

        System.out.println("=== PRUEBAS DE BÚSQUEDA LINEAL ===\n");

        System.out.println("Prueba 1 - Elemento al inicio:");
        System.out.println("Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]");
        System.out.println("Buscar: 10");
        int comp1 = busquedaLineal(arreglo1, 10);
        System.out.println("Comparaciones realizadas: " + comp1);
        System.out.println("Análisis: MEJOR CASO - T(n) = 5 = O(1), S(n) = 3 = O(1)\n");

        System.out.println("Prueba 2 - Elemento al final:");
        System.out.println("Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]");
        System.out.println("Buscar: 33");
```

```java
        int comp2 = busquedaLineal(arreglo1, 33);
        System.out.println("Comparaciones realizadas: " + comp2);
        System.out.println("Análisis: PEOR CASO - T(n) = 3 + 3n = 3 + 3(8) = 27 = O(n), S(n) = 3 = O(1)\n");

        System.out.println("Prueba 3 - Elemento en el medio:");
        System.out.println("Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]");
        System.out.println("Buscar: 47");
        int comp3 = busquedaLineal(arreglo1, 47);
        System.out.println("Comparaciones realizadas: " + comp3);
        System.out.println("Análisis: CASO PROMEDIO - T(n) = 3 + 3n/2 = 3 + 3(8)/2 = 15 = O(n), S(n) = 3 = O(1)\n");

        System.out.println("Prueba 4 - Elemento no existe:");
        System.out.println("Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]");
        System.out.println("Buscar: 100");
        int comp4 = busquedaLineal(arreglo1, 100);
        System.out.println("Comparaciones realizadas: " + comp4);
        System.out.println("Análisis: PEOR CASO - T(n) = 3 + 3n = 3 + 3(8) = 27 = O(n), S(n) = 3 = O(1)\n");

        System.out.println("Prueba 5 - Arreglo vacío:");
        System.out.println("Arreglo: []");
        System.out.println("Buscar: 5");
        int comp5 = busquedaLineal(arregloVacio, 5);
        System.out.println("Comparaciones realizadas: " + comp5);
        System.out.println("Análisis: CASO ESPECIAL - T(n) = 2 = O(1), S(n) = 3 = O(1)\n");

        System.out.println("=== PRUEBAS CON FUNCIÓN COMPLETA ===\n");

        ResultadoBusqueda resultado1 = busquedaLinealCompleta(arreglo2, 7);
        System.out.println("Buscar 7 en [1,2,3,4,5,6,7,8,9,10]: " + resultado1);

        ResultadoBusqueda resultado2 = busquedaLinealCompleta(arreglo2, 15);
```

```java
        ResultadoBusqueda resultado2 = busquedaLinealCompleta(arreglo2, 15);
        System.out.println("Buscar 15 en [1,2,3,4,5,6,7,8,9,10]: " + resultado2);

        System.out.println("\n=== ANÁLISIS DE COMPLEJIDAD ===");
        analizarComplejidad(arreglo1.length);
    }

    public static void analizarComplejidad(int n) {  1 usage
        System.out.println("\n--- RESULTADOS DEL ANÁLISIS DE COMPLEJIDAD ---");

        System.out.println("\nComplejidad Temporal T(n):");
        System.out.println("  • Mejor caso: T(n) = 5 = O(1)");
        System.out.println("  • Caso promedio: T(n) = 3 + 3n/2 = O(n)");
        System.out.println("  • Peor caso: T(n) = 3 + 3n = O(n)");

        System.out.println("\nComplejidad Espacial S(n):");
        System.out.println("  • S(n) = 3 = O(1)");

        System.out.println("\nTipo de algoritmo: Búsqueda lineal secuencial");
    }
}
```

Compilación

```
=== PRUEBAS DE BÚSQUEDA LINEAL ===

Prueba 1 - Elemento al inicio:
Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]
Buscar: 10
Elemento 10 encontrado en la posición: 0
Comparaciones realizadas: 1
Análisis: MEJOR CASO - T(n) = 5 = O(1), S(n) = 3 = O(1)

Prueba 2 - Elemento al final:
Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]
Buscar: 33
Elemento 33 encontrado en la posición: 7
Comparaciones realizadas: 8
Análisis: PEOR CASO - T(n) = 3 + 3n = 3 + 3(8) = 27 = O(n), S(n) = 3 = O(1)

Prueba 3 - Elemento en el medio:
Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]
Buscar: 47
Elemento 47 encontrado en la posición: 3
Comparaciones realizadas: 4
Análisis: CASO PROMEDIO - T(n) = 3 + 3n/2 = 3 + 3(8)/2 = 15 = O(n), S(n) = 3 = O(1)

Prueba 4 - Elemento no existe:
Arreglo: [10, 25, 3, 47, 15, 8, 92, 33]
Buscar: 100
Elemento 100 no encontrado en el arreglo
Comparaciones realizadas: 8
Análisis: PEOR CASO - T(n) = 3 + 3n = 3 + 3(8) = 27 = O(n), S(n) = 3 = O(1)
```

```
Prueba 5 - Arreglo vacío:
Arreglo: []
Buscar: 5
Elemento 5 no encontrado en el arreglo
Comparaciones realizadas: 0
Análisis: CASO ESPECIAL - T(n) = 2 = O(1), S(n) = 3 = O(1)

=== PRUEBAS CON FUNCIÓN COMPLETA ===

Buscar 7 en [1,2,3,4,5,6,7,8,9,10]: Elemento encontrado en posición 6 con 7 comparaciones
Buscar 15 en [1,2,3,4,5,6,7,8,9,10]: Elemento no encontrado después de 10 comparaciones

=== ANÁLISIS DE COMPLEJIDAD ===

--- RESULTADOS DEL ANÁLISIS DE COMPLEJIDAD ---

Complejidad Temporal T(n):
  • Mejor caso: T(n) = 5 = O(1)
  • Caso promedio: T(n) = 3 + 3n/2 = O(n)
  • Peor caso: T(n) = 3 + 3n = O(n)

Complejidad Espacial S(n):
  • S(n) = 3 = O(1)

Tipo de algoritmo: Búsqueda lineal secuencial
```