ALGORITMOS DE BUSQUEDA / BUSQUEDA BINARIA

GRUPO # 5

Juan David Jiménez Romero

QUE SON LOS ALGORITMOS DE BUSQUEDA?

Los procesos de búsqueda involucran recorrer un arreglo completo con el fin de encontrar algo. Lo más común es buscar el menor o mayor elemento (cuando es puede establecer un orden), o buscar el índice de un elemento determinado.

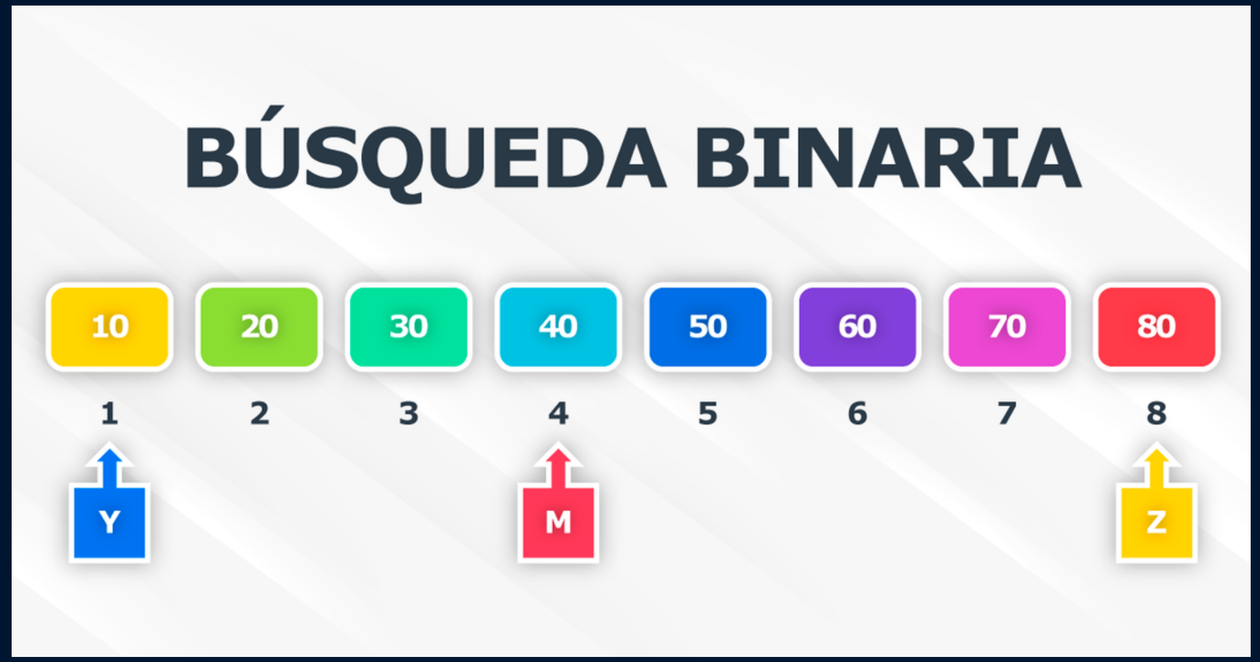
Para buscar el menor o mayor elemento de un arreglo, podemos usar la estrategia, de suponer que el primero o el último es el menor (mayor), para luego ir comparando con cada uno de los elementos, e ir actualizando el menor (mayor). A esto se le llama Búsqueda Lineal.

TIPOS DE ALGORITMOS DE BUSQUEDA

- Búsqueda secuencial: Es un método para encontrar un valor en una lista recorriéndola secuencialmente

- Busqueda Hash: también conocida como tabla hash, es una estructura de datos que implementa el tipo de dato "array" de forma que se pueda buscar un valor asociado a una clave.

- Búsqueda Binaria: Este será presentado mejor a continuacion.



¿QUÉ ES LA BUSQUEDA BINARIA?

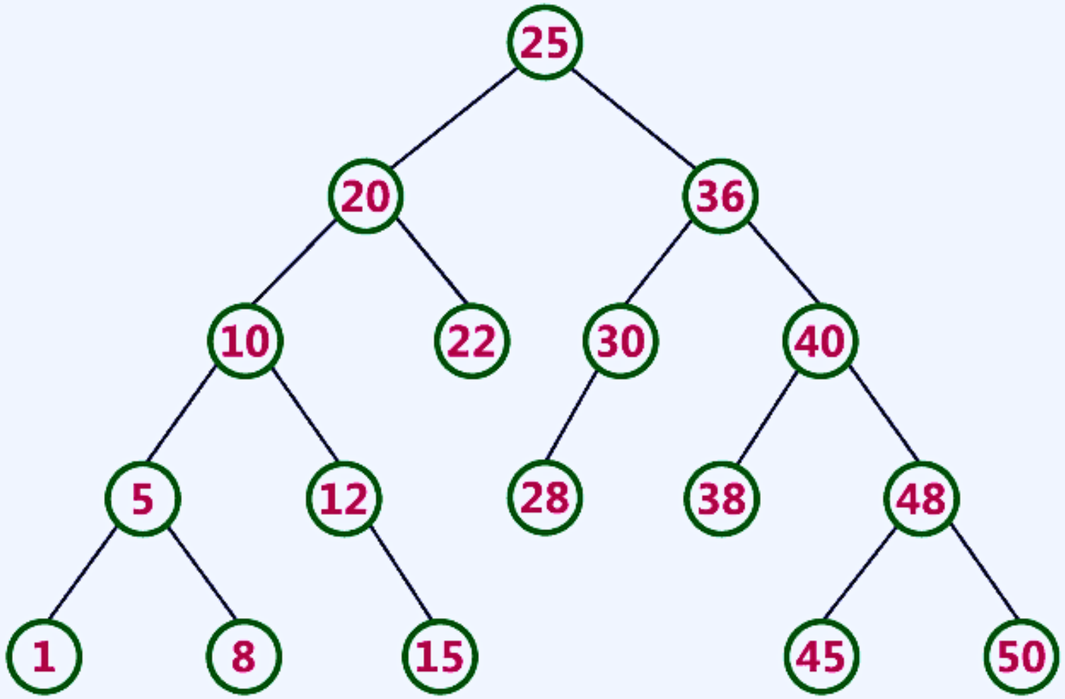
La búsqueda binaria es un algoritmo eficiente para encontrar un elemento en una lista ordenada de elementos. Funciona al dividir repetidamente a la mitad la porción de la lista que podría contener al elemento, hasta reducir las ubicaciones posibles a solo una.

CARACTERÍSTICAS DE LA BUSQUEDA BINARIA

- El algoritmo de búsqueda binaria es mucho mas rápido que el algoritmo de búsqueda lineal.
- A diferencia del algoritmo de búsqueda lineal que elimina un elemento por iteración, El algoritmo de búsqueda binaria elimina la mitad de elementos en cada una de las iteraciones.
- Este algoritmo tiene una desventaja, y es que solo funciona con arreglos que se encuentran ordenados.

¿COMO FUNCIONA LA BUSQUEDA BINARIA?

La búsqueda binaria utiliza Divide and Conquer osea “divide y conquistaras”, el cual consiste dividir el arreglo en 2 partes, tomar un punto medio, verificar si este punto medio es el valor que se busca o de lo contrario mover el inicio o el fin dependiendo de si el punto medio es mayor o menor que el valor buscado.



EL ALGORITMO DE BUSQUEDA BINARIA

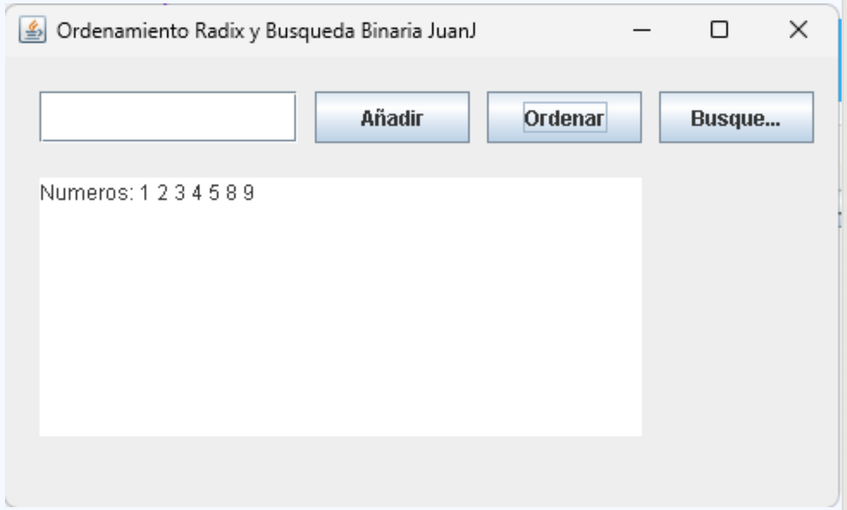
Start Page | BusquedaBinariaPresentacion.java | BusquedaBinaria.java

Source | History

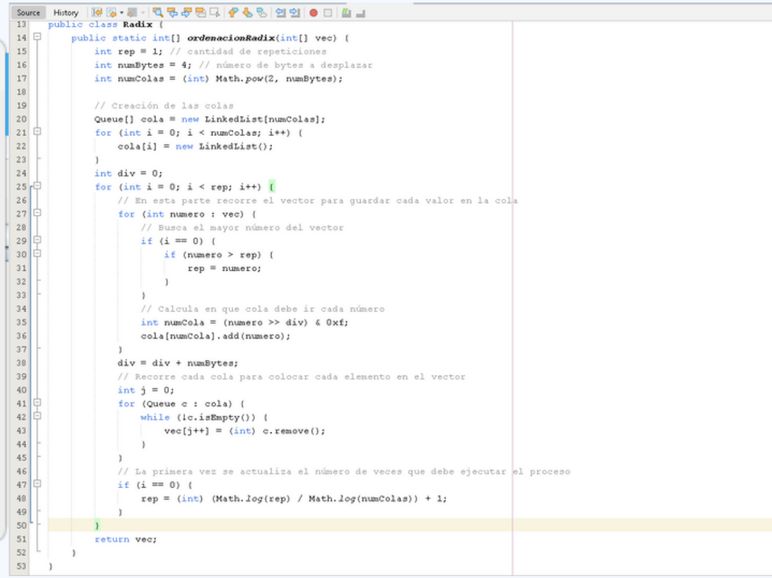
```java
//Paquete en el que se encuentra la clase
package busquedabinariapresentacion;
/**
 *
 * @author jrome
 */
//inicio de la clase BusquedaBinaria
public class BusquedaBinaria{
    //declaracion del metodo que realiza realiza la busqueda de los valores de tipo entero
    public int busquedaBinaria(int elementos[], int x){//Este método toma como parámetros un arreglo de elementos
           //ordenados y un valor a buscar, y devuelve la posición del valor en el arreglo o -1 si no se encuentra.
        //declaracion y inicializacion de las variables donde elementos[] es el array utilizado y x es el valor separado
        int l = 0, r = elementos.length - 1; //l inicializa en 0 y r en la cantidad de elementos del array menos uno
                                             //l significa Left (izquierda) osea el valor de la posicion inicial del array
                                             //r significa Right (derecha) osea el valor de la posicion del tamaño del array menos 1
        while (l <= r) {// buque while (mientras) que nos dice que va a seguir haciendose mientras l sea menor o igual a r
            //se declara un valor entero "m" que sera igual a la operacion l + (r - l) / 2
            int m = l + (r - l) / 2;
            //un if que nos dice que devolvera verdadero si la posicion m del array sea igual a x
            if (elementos[m] == x)
                //en caso de ser positivo el valor regresado sera m
                return m;
            //if que nos dice que en caso de que el valor de la posicion de m del array sea menor que x
            if (elementos[m] < x)
                //este hara que l sea igual al valor de m + 1
                l = m + 1;
            // else (demas) que nos dice que si el ultimo if que se realizo tambien debe realizar
            else
                //Que r tome el valor de m - 1
                r = m - 1;
        }
        //al cerrar el bucle while este retornara el valor de -1 y si ninguna de las otras funciones se cumplia se retorna
        return -1;
    }//fin del metodo busqueda binaria
}//fin de la clase BusquedaBinaria
```

EJEMPLO DE BUSQUEDA BINARIA:

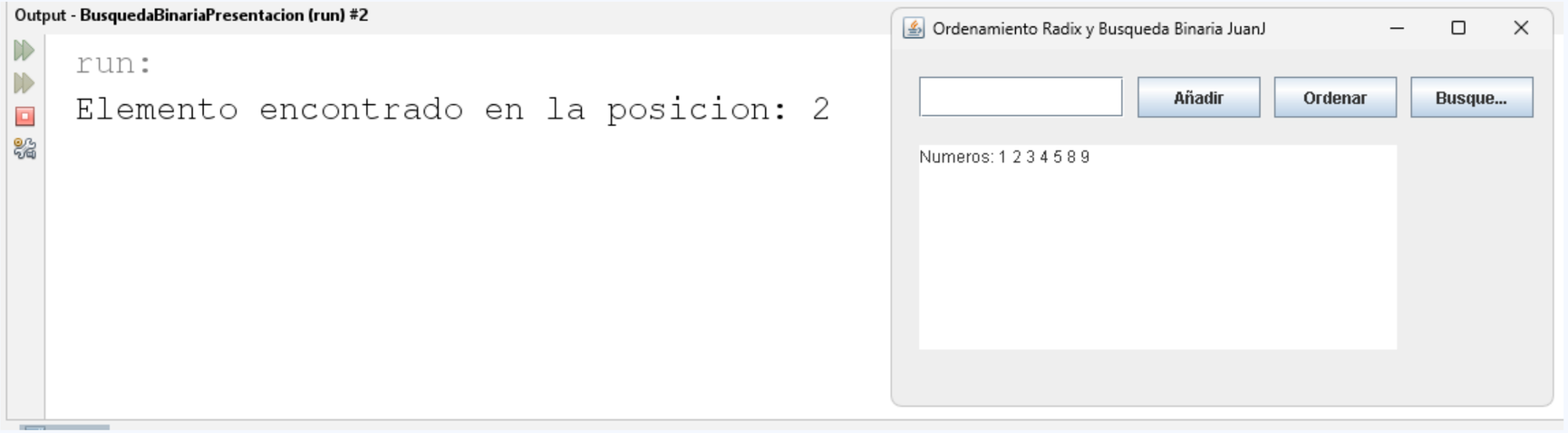
Primero se ingresan los datos al array



Segundo se ordena el array con la clase de ordenamiento por el metodo de Radixsort



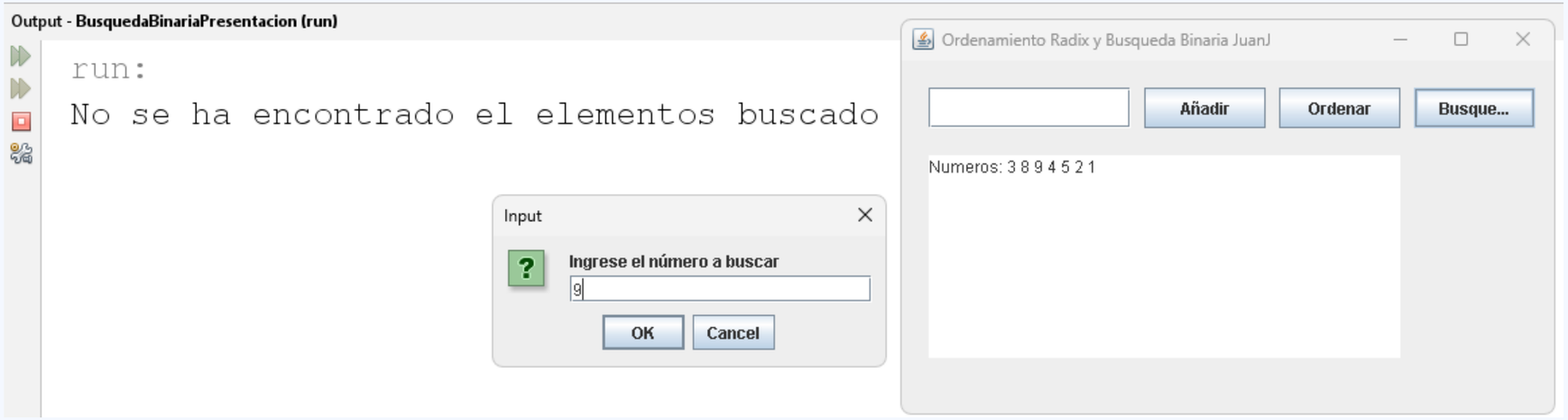
```
public class Radix {
    public static int[] ordenacionRadix(int[] vec) {
        int rep = 1; // cantidad de repeticiones
        int numBytes = 4; // número de bytes a desplazar
        int numColas = (int) Math.pow(2, numBytes);

        // Creación de las colas
        Queue[] cola = new LinkedList[numColas];
        for (int i = 0; i < numColas; i++) {
            cola[i] = new LinkedList();
        }
        int div = 0;
        for (int i = 0; i < rep; i++) {
            // En esta parte recorre el vector para guardar cada valor en la cola
            for (int numero : vec) {
                // Busca el mayor número del vector
                if (i == 0) {
                    if (numero > rep) {
                        rep = numero;
                    }
                }
                // Calcula en que cola debe ir cada número
                int numCola = (numero >> div) & 0xf;
                cola[numCola].add(numero);
            }
            div = div + numBytes;
            // Recorre cada cola para colocar cada elemento en el vector
            int j = 0;
            for (Queue c : cola) {
                while (!c.isEmpty()) {
                    vec[j++] = (int) c.remove();
                }
            }
            // La primera vez se actualiza el número de veces que debe ejecutar el proceso
            if (i == 0) {
                rep = (int) (Math.log(rep) / Math.log(numColas)) + 1;
            }
        }
        return vec;
    }
}
```



Por ultimo el usamos la búsqueda binaria para que nos muestre donde esta nuestro dato solicitado

EJECUCIÓN DEL PROGRAMA CUANDO EL ARRAY NO ESTA ORDENADO:

Primero se ingresan los datos al array



Se realiza la búsqueda pero como no se encuentra ordenada esta nos mostrara que el dato no se pudo encontrar.

Preguntas:

1. ¿Qué enfoque utiliza el algoritmo de búsqueda binaria?

a) Enfoque lineal

b) Enfoque exponencial

c) Enfoque "Divide and Conquer"

d) Enfoque aleatorio

Preguntas:

1. ¿Qué enfoque utiliza el algoritmo de búsqueda binaria?

a) Enfoque lineal

b) Enfoque exponencial

c) Enfoque "Divide and Conquer"

d) Enfoque aleatorio

Respuesta:

c) Enfoque “Divide and Conquer”

Preguntas:

2. ¿Cuál de las siguientes no es una caracteristica del algoritmo de búsqueda binaria?
a) El algoritmo de búsqueda binaria es mucho mas rápido que el algoritmo de búsqueda lineal.
b) El algoritmo de búsqueda binaria elimina un elemento por iteración.
c) solo funciona con arreglos que se encuentran ordenados.
d) El algoritmo de búsqueda binaria elimina la mitad de elementos en cada una de las iteraciones.

Preguntas:

2. ¿Cuál de las siguientes no es una caracteristica del algoritmo de búsqueda binaria?
a) El algoritmo de búsqueda binaria es mucho mas rápido que el algoritmo de búsqueda lineal.
b) El algoritmo de búsqueda binaria elimina un elemento por iteración.
c) solo funciona con arreglos que se encuentran ordenados.
d) El algoritmo de búsqueda binaria elimina la mitad de elementos en cada una de las iteraciones.

Respuesta:

b) El algoritmo de búsqueda binaria elimina un elemento por iteración.

Preguntas:

3.Qué condición debe cumplir el arreglo de la busqueda binaria es:

a) El arreglo debe estar ordenado

b) El arreglo debe estar ordenado de mayor a menor.

c) El arreglo puede estar ordenado de menor a mayor o de mayor a menor.

d) El arreglo no necesita estar ordenado.

Preguntas:

3.Qué condición debe cumplir el arreglo de la busqueda binaria es:

a) El arreglo debe estar ordenado

b) El arreglo debe estar ordenado de mayor a menor.

c) El arreglo puede estar ordenado de menor a mayor o de mayor a menor.

d) El arreglo no necesita estar ordenado.

Respuesta:

a) El arreglo debe estar ordenado

Preguntas:

4.¿Qué significa “L” que se encuentra en la clase BusquedaBinaria?

a) El índice del elemento.

b) 1.

c) La posición del tamaño del array menos 1.

d) La posición inicial del array

Preguntas:

4.¿Qué significa “L” que se encuentra en la clase BusquedaBinaria?

a) El índice del elemento.

b) 1.

c) La posición del tamaño del array menos 1.

d) La posición inicial del array

Respuesta:

d) La posición inicial del array

Preguntas:

5.¿Cuál es el valor que se devuelve si el elemento buscado se encuentra en el arreglo?
a) 0
b) 1
c) -1
d) El índice del elemento.

Preguntas:

5.¿Cuál es el valor que se devuelve si el elemento buscado se encuentra en el arreglo?

a) 0

b) 1

c) -1

d) El índice del elemento.

Respuesta:

d) El índice del elemento.

MUCHAS GRACIAS