



# Busqueda Binaria



90	30	50	42	67	23
23	30	42	50	67	90
0	1	2	3	4	5

Antes de realizar una búsqueda binaria sobre un arreglo, se necesita que el arreglo esté ordenado. Entonces va dividiendo el arreglo por la mitad hasta que encuentra el que está buscando.

# Binaria

<b>23</b>	<b>30</b>	<b>42</b>	<b>50</b>	<b>67</b>	<b>90</b>
0	1	2	3	4	5

Si suponemos que busca el 50 va a dividir el arreglo por la mitad

<b>23</b>	<b>30</b>	<b>42</b>	<b>50</b>	<b>67</b>	<b>90</b>
-----------	-----------	-----------	-----------	-----------	-----------

Descarta la primer mitad y vuelve a dividir el arreglo por la mitad

<b>50</b>	<b>67</b>	<b>90</b>
-----------	-----------	-----------

Descarta la segunda mitad y vuelve a dividir el arreglo por la mitad

<b>50</b>
-----------

# Binaria

23

30

42

50

67

90

**Binaria ( 50, L ) : Realiza 3 comparaciones para lograr encontrar al 50**

**Binaria ( 42, L ) : Realiza 1 comparación para lograr encontrar el 42**

**Binaria ( 30, L ) : Realiza 3 comparaciones para encontrar el 30**

# Binaria

```
int binaria (int numeros[], int tam, int x){
    int inicio = 0;
    int fin = tam-1;
    int medio;
    do {
        medio = ( inicio + fin ) / 2 ;
        if ( x == numeros [ medio ] ) {
            return medio;
        }
        else if (x > numeros[medio]) {
            inicio = medio + 1;
        }
        else if (x < numeros[medio]) {
            fin = medio - 1;
        }
    } while ( inicio <= fin );
    return -1;
}
```

# Binaria

23

30

42

50

67

90

**Binaria ( 42, L ) : Realiza 1 comparación para lograr encontrar el 42**

A la búsqueda del 42 es a lo que llamamos el **mejor de los casos** porque al primer intento lo encuentro

**Binaria ( 30, L ) : Realiza 3 comparaciones para encontrar el 30**

A la búsqueda del 30, 50 o el 90 es a lo que llamamos el **peor de los casos** porque requiere hacer el mayor número de comparaciones para encontrarlos