

Búsqueda Binaria

Computer Science

CS1100 - Introducción a Ciencia de la Computación

la aguja en el pajar?



Las búsquedas son parte de nuestra vida

The screenshot shows a web interface for flight reservations. On the left is a vertical sidebar with five menu items, each with a red icon: 'Reserva tu vuelo' (hand icon), 'Vuelo, hotel y auto' (car and plane icon), 'Check-in' (laptop with checkmark icon), 'Estado del vuelo' (clock icon), and 'Mis viajes' (calendar icon). The main area has a dark blue background. At the top, there are two white input fields: 'CÓDIGO DE RESERVA' containing 'UTECVL' and 'APELLIDO DEL PASAJERO' containing 'INTRODUCCIONCD'. Below these fields is a link: 'Si no tienes tu código de reserva, puedes ingresar con tu cuenta'. In the center of the main area is a large red button with the text 'Comienza tu Check-in'.

Reserva tu vuelo

Vuelo, hotel y auto

Check-in

Estado del vuelo

Mis viajes

CÓDIGO DE RESERVA
UTECVL

APELLIDO DEL PASAJERO
INTRODUCCIONCD

[Si no tienes tu código de reserva, puedes ingresar con tu cuenta](#)

Comienza tu Check-in

Logro de la Sesión

Al finalizar esta sesión, estarás en la capacidad de:

- Encontrar elementos dentro de una estructura de datos.

Logro de la Sesión

Al finalizar esta sesión, estarás en la capacidad de:

- Encontrar elementos dentro de una estructura de datos.
- Comparar el performance de una búsqueda lineal vs. búsqueda binaria.

Logro de la Sesión

Al finalizar esta sesión, estarás en la capacidad de:

- Encontrar elementos dentro de una estructura de datos.
- Comparar el performance de una búsqueda lineal vs. búsqueda binaria.
- Utilizar conceptos previos de ordenamiento.

Encontrar un elemento dentro de una Lista

```
1      #Hasta ahora hemos buscado elementos de una manera lineal;  
      generamos un lazo y atravesamos todos los elementos de  
      nuestra estructura de datos para encontrar el elemento  
      que buscamos.  
2      import random  
3      arr1 = []  
4      for i in range(0,10):  
5          arr1.append(random.randint(1,10))  
6      for i in range(0,10):  
7          if (arr1[i]==7):  
8              print ("El numero 7 se encuentra en pos.",i)
```

Encontrar un nombre dentro de una Lista

```
1  arr1 = []
2  arr1.append("Andres")
3  arr1.append("Diana")
4  arr1.append("Javier")
5  #arr1.append("Carlos")
6  for i in range (len(arr1)):
7      print(arr1[i])
8  encontro = False
9  #Deseamos encontrar a Carlos
10 for i in range (len(arr1)):
11     if (arr1[i]=="Carlos"):
12         encontro = True
13         posicion = i
14 if (encontro==True):
15     print("Carlos SI se encuentra en la lista y la posicion :
16         ",posicion)
17 else:
18     print("Carlos NO se encuentra en la lista")
```


Busqueda lineal

Busqueda lineal en estructuras de datos grandes

Qué ocurriría si la lista o estructura de datos contiene 1 millón de elementos o 10 millones, vamos a atravesar todo el arreglo? Qué pasa si el elemento que buscamos se encuentra en las primeras posiciones? Qué pasa si el elemento que buscamos se encuentra en las ultimas posiciones?

Búsqueda Binaria

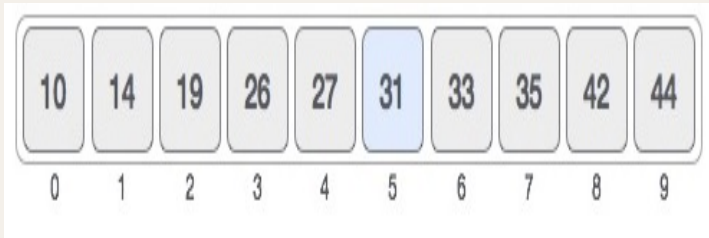
Definición

- Algoritmo que nos permite encontrar elementos de manera eficiente.
- Previamente nuestra estructura de datos tiene que estar ordenada
- Nos va a permitir mejorar el performance en la búsqueda

Primer Paso

Tener la lista ordenada

Antes de llamar a nuestra función de búsqueda binaria, la lista debe haber sido ordenado usando algoritmos eficientes como quicksort o bubblesort. En nuestro caso queremos encontrar el numero 31.



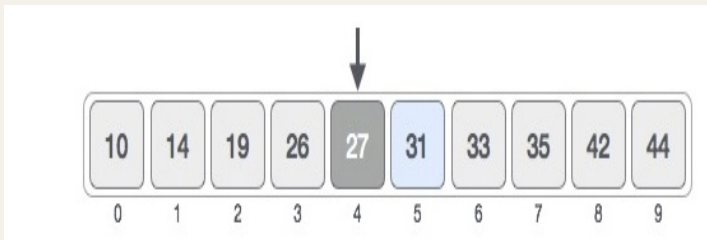
Segundo Paso

Encontrar la mitad

Se halla el elemento medio del arreglo mediante la formula:

$$\blacksquare \text{ MID} = \text{LOW} + (\text{HIGH} - \text{LOW}) / 2$$

Siendo $\text{MID} = 4 = 0 + (9 - 0) / 2$

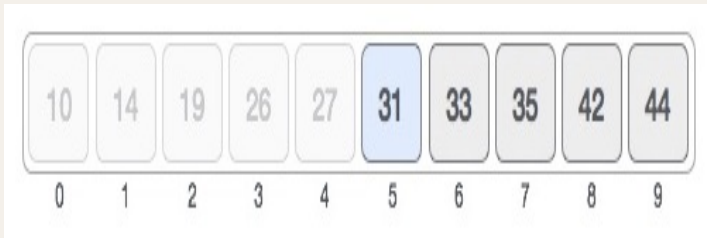


Tercer Paso

Encontrar el elemento

En esta caso el número 31 no se encuentra en la posición MID de la lista, lo cual quiere decir que debe estar en la mitad superior de la lista. Cambiamos LOW a MID + 1 y buscamos nuevamente el valor MID.

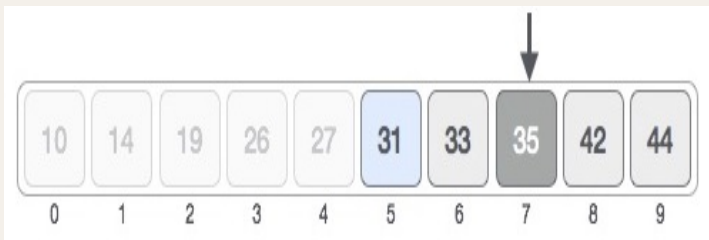
- $LOW = MID + 1$
- $MID = LOW + (HIGH - LOW) / 2$



Cuarto Paso

Encontrar la mitad

El nuevo valor de LOW es 5 y el valor de MID es 7. Comparamos el valor de la posición 7 (MID) con el número que estamos buscando, 31, y vemos que no coincide, lo cual nos indica que el valor debe hallarse en la parte inferior.

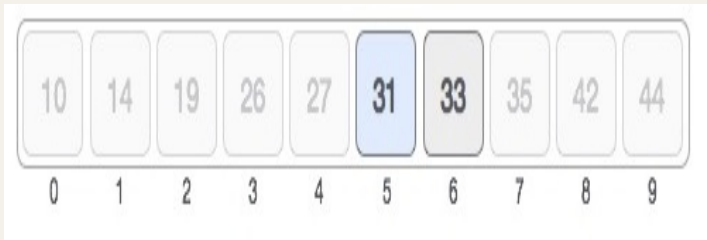


Quinto Paso

Encontrar nuevamente LOW y MID

LOW=5

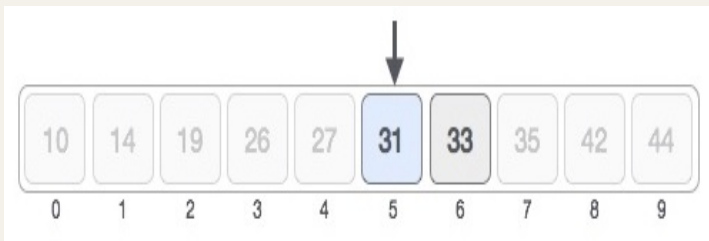
MID = $5 + (6-5)/2$, MID= 5



Sexto Paso

Encontrar la Mitad

En este caso ahora si la posicion MID (5), si contiene el numero 31 , el cual estabamos buscando.



Algoritmo Búsqueda Binaria

```
1  def busquedaBinaria(Lista,item):
2      low = 0
3      high = len(Lista)-1
4      encontrado= False
5      while (low<=high) and not encontrado:
6          mid = (low + high)//2
7          if Lista[mid]==item:
8              encontrado = True
9          else :
10             if item<Lista[mid]:
11                 high = mid-1
12             else:
13                 low = mid+1
14      return encontrado
15
16  lista1= [3,2,22,40,50,66,98]
17  print(busquedaBinaria(lista1,67))
```

Individual Work

Escriba un algoritmo que reciba del usuario una serie de nombres hasta que se ingrese "-1". Todos los nombres se deben almacenar en una lista. Adicionalmente, solicite al usuario un nombre que buscar. Luego con ayuda de la búsqueda binaria busque en que posición de la lista se encuentra el nombre a buscar ingresado por el usuario.

Cierre

En esta sesión aprendiste:

- Hallar elementos en estructuras de datos.

Cierre

En esta sesión aprendiste:

- Hallar elementos en estructuras de datos.
- Entender el algoritmo para búsqueda binaria

Cierre

En esta sesión aprendiste:

- Hallar elementos en estructuras de datos.
- Entender el algoritmo para búsqueda binaria
- Utilizar búsqueda binaria para tener mayor performance