	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	35/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Guía Práctica de Estudio 4


Algoritmos de búsqueda parte 1

Elaborado por:

M.I. Elba Karen Sáenz García

Revisión:

Ing. Laura Sandoval Montaña

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	36/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Guía Práctica 4

Estructura de datos y Algoritmos II

Algoritmos de Búsqueda. Parte 1.

Objetivo: El estudiante identificará el comportamiento y características de algunos algoritmos de búsqueda por comparación de llaves.

Actividades

- Implementar el algoritmo iterativo y recursivo de búsqueda lineal en algún lenguaje de programación para localizar alguna llave o valor en una secuencia de datos.
- Implementar el algoritmo iterativo y recursivo de búsqueda binaria en algún lenguaje de programación para localizar alguna llave o valor en una secuencia de datos.

Antecedentes

- Análisis previo de los algoritmos en clase teórica.
- Conocimiento de algún algoritmo de ordenamiento.
- Manejo de arreglos o listas, estructuras de control y funciones en Python 3.


Introducción

Recuperar información de una computadora es una de las actividades más útiles e importantes, muy comúnmente se tiene un nombre o llave que se quiere encontrar en una estructura de datos tales como una lista u arreglo.

Al proceso de encontrar un dato específico llamado llave en alguna estructura de datos, se denomina búsqueda. Este proceso termina exitosamente cuando se localiza el elemento que contienen la llave buscada, o termina sin éxito, cuando no aparece ningún elemento con esa llave.

Los algoritmos de búsqueda se pueden clasificar en:

- Búsqueda por comparación
- Búsqueda por transformación de llaves

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	37/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

En este documento se describen dos de los algoritmos de búsqueda por comparación: la búsqueda lineal o secuencial y la búsqueda binaria, que son dos de las técnicas más comunes; la primera por sencilla y la otra por eficiente.

Búsqueda Lineal

En este algoritmo se empieza a buscar desde la primera posición de la lista, comparando el elemento con la llave y si no se encuentra continúa verificando las siguientes posiciones hasta encontrarlo, entonces el algoritmo termina. Así el elemento a buscar puede estar en el primer elemento, el último o entre ellos.

A continuación, se muestra un algoritmo en pseudocódigo de una función que realiza la búsqueda secuencial del elemento x en una lista implementada como un arreglo lineal $A[0,1,2, \dots, n-1]$ de n elementos. El cual retorna la posición o índice del arreglo donde se encuentra el elemento x a buscar, y el valor de -1 si x no fue encontrado.

BúsquedaLineal (A,n,x)

Inicio

 encontrado=-1

 Para $k=0$ hasta $n-1$

 Si $x==A[k]$

 encontrado = k

 Fin Si


 Fin Para

 Retorna encontrado

Fin

Como se puede observar en la función BúsquedaLineal(), el número de iteraciones siempre es igual al tamaño del arreglo A , independientemente de dónde se encuentre el elemento a buscar y como todas las operaciones del interior de la estructura de repetición tienen una complejidad constante, la complejidad de este algoritmo de búsqueda secuencial es de $O(n)$.

Existen varias mejoras al algoritmo, una de ellas es no esperar a revisar todos los elementos del arreglo, si el buscado ya se encontró, entonces se tiene que terminar. Una forma de conseguirlo es revisar en cada iteración k , si el elemento a buscar x es ya igual al elemento $A[k]$, y si lo es, entonces retornar la posición del elemento encontrado y terminar. A continuación, se muestra una posible función en pseudocódigo donde se considera esta mejora [1].

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	38/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

BúsquedaLinealMejorado

Inicio

 encontrado=-1

 Para k=0 hasta n-1

 Si $A[k]==x$

 encontrado= k

 Salir de la estructura de repetición

 Fin Si

 Fin Para

 retorna encontrado

Fin

También se tiene la llamada búsqueda secuencial con centinela que asegura siempre encontrar el elemento a buscar. Lo que se hace es situar el valor que se está buscando al final del arreglo $A[n - 1]$ (centinela), guardando lo que se encuentra inicialmente ahí en algún lugar temporal. Así en la estructura de repetición que va revisando todos los elementos de A siempre se obtiene un valor del índice k donde se encuentra el elemento.

Al término de las iteraciones se regresa el valor inicial del elemento $A[n - 1]$ a su lugar y solo resta verificar si éste o el posicionado en $k < n - 1$ es el valor buscado, si no es ninguno de ellos, entonces, el elemento no está en la lista.

Lo anterior se puede representar en la siguiente función en pseudocódigo [1]

BusquedaLinealCentinela(A, n, x)

 tmp= $A[n-1]$

$A[n-1]=x$

 k=0

 Mientras $A[k]$ sea diferente de x

 k=k+1

 Fin Mientras

$A[n-1]=tmp$

 Si $k < n-1$ o $A[n-1]==x$


 retorna k

 En otro caso

 retorna -1

 Fin Si

Fin

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	39/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

En estos dos últimos algoritmos el tiempo de ejecución en el mejor de los casos se da cuando el valor a buscar se encuentra en el primer elemento de la lista y es $\theta(1)$. En el peor de los casos cuando está en la última posición y es $\theta(n)$.

Existe una versión recursiva del algoritmo de búsqueda lineal, donde se tendrá un caso base de fracaso que se da cuando se rebasa el número de elementos de la lista y un caso base de éxito cuando después de algunas llamadas recursivas el elemento analizado de la lista es igual a la llave. Un algoritmo en pseudocódigo se puede plantear como se muestra en la siguiente función.

BusquedaLinealRecursiva (A,x,ini,fin)

Inicio

Si ini > fin

 encontrado=-1

Si no

 Si A[ini]==x

 encontrado=ini

 Si no

 encontrado = BusquedaLinealRecursiva(A, x,ini+1,fin)

 Fin Si no

Fin Si no


 retorna encontrado

Fin

Búsqueda Binaria

El algoritmo de búsqueda binaria o también llamado dicotómica es eficiente para encontrar un elemento en una lista ya ordenada y es un ejemplo de la técnica divide y conquista. Es como buscar una palabra en un diccionario, donde dependiendo de la primera letra de la palabra a buscar se abre el libro cerca del principio, del centro o al final. Se puede tener suerte y haber abierto la página correcta o bien moverse a paginas posteriores o anteriores del libro.

Entonces en el algoritmo se divide repetidamente a la mitad la porción de la lista que podría contener al elemento, hasta reducir las ubicaciones posibles a solo una. La estrategia consiste en comparar una llave con el elemento de en medio de la lista, si es igual entonces se encontró el elemento, sino, cuando x es menor que el elemento del medio se aplica la misma estrategia a la lista de la izquierda y si x es mayor, a la lista de la derecha.

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	40/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Para describir el algoritmo con más detalle, se supone una lista ordenada de forma ascendente que está almacenada en un arreglo unidimensional $A[0, \dots, n - 1]$ de n elementos y se quiere encontrar la llave x .

- Primero se calcula el índice del punto medio del arreglo utilizando los índices de inicio (izquierdo) y fin (derecho) de la lista o sublista de búsqueda, $medio = \left\lfloor \frac{indiceIzquierdo + indiceDerecho}{2} \right\rfloor$, para poder dividir la lista en dos sub-listas Figura 4.1.

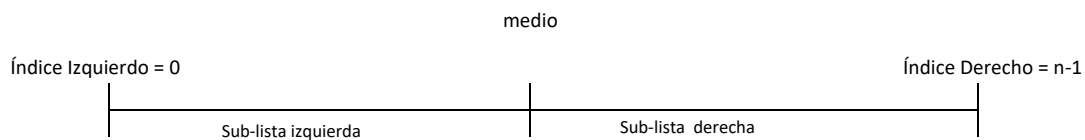


Figura 4.1

- Después se compara el punto *medio* con la llave a buscar x ,
 - Si son iguales, entonces la llave fue encontrada, y el algoritmo termina. Figura 4.2.

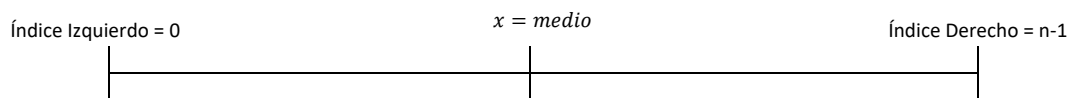


Figura 4.2

- Si el valor del arreglo A con índice *medio* es mayor a la llave x ($A[medio] > x$), se descarta la sub-lista de la derecha incluyendo el punto *medio* y se regresa al paso 1, donde la nueva sub-lista de búsqueda mantiene el índice izquierdo y el índice derecho cambia a *medio* - 1. Figura 4.3

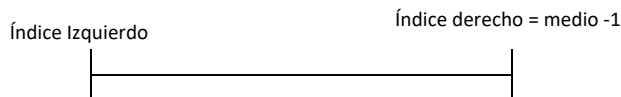


Figura 4.3

- Si el valor del arreglo A con índice *medio* es menor a la llave x ($A[medio] < x$), se descarta la sub-lista de la izquierda incluyendo el punto medio y se regresa al paso 1, donde la nueva sub-lista de búsqueda mantiene el índice derecho y el índice izquierdo cambia a *medio* + 1. Figura 4.4.

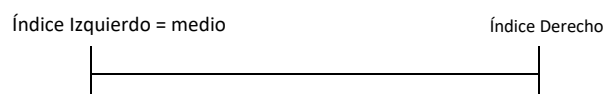



Figura 4.4

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	41/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Si en algún momento la sub-lista de búsqueda tiene longitud 0 o negativa significa que el valor buscado no se encuentra en la lista.

Ejemplo: Se quiere buscar el elemento $x = 18$ en un arreglo unidimensional de 13 elementos que contiene los siguientes datos, los cuales ya están ordenados:

0	1	2	3	4	5	6	7	8	9	10	11	12
10	12	13	14	18	20	25	27	30	35	40	45	47

Donde *índice izquierdo* = 0 e *índice derecho* = 12.

$medio = \frac{12}{2} = 6$, entonces $A[medio] = 25$.

Se compara 18 con 25, como $18 < 25$, ahora se trabaja con la sub-lista izquierda:

0	1	2	3	4	5
10	12	13	14	18	20

Donde *índice izquierdo* = 0 e *índice derecho* = $6 - 1 = 5$.

$medio = \frac{5}{2} = 2$, entonces $A[medio] = 13$.

Se compara 18 con 13, como $18 > 13$, ahora se trabaja con la sub-lista derecha:


3	4	5
14	18	20

Donde *índice izquierdo* = $medio + 1 = 3$ e *índice derecho* = 5.

$medio = \frac{8}{2} = 4$, entonces $A[medio] = 18$.

Se observa que $A[medio]$ es igual $x = 18$, entonces el algoritmo termina.

Una función del algoritmo en pseudocódigo de la búsqueda binaria descrita es:

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	42/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

BusquedaBinariaIterativa(A,x,indiceIzq,indiceDer)

Inicio

Encontrado= -1

Mientras indiceIzq <= indiceDer

Medio=[(indiceIzq + IndiceDer)/2]

Si x == A[Medio] entonces

Encontrado=Medio

Si no

Si A[medio] < x

indiceIzq=medio +1

Si no

indiceDer=medio-1

Fin Si no

Fin Si no

Fin Mientras

Retorna Encontrado

Fin

El algoritmo de búsqueda binaria también tiene un enfoque recursivo. Donde se tiene un caso base de fracaso, que sucede cuando se sobrepasa el número de elementos de la lista y un caso base de éxito, cuando después de algunas llamadas recursivas el elemento $A[\text{medio}]$ es igual a la llave x .

A continuación, se muestra una función en pseudocódigo que ilustra la forma recursiva:

BusquedaBinariaRecursiva(A,x,indiceIzq,indiceDer)

Inicio

Si indiceIzquierdo > indiceDerecho y x es diferente de A[indiceDerecho]

Retorna -1

Fin Si

medio = [(indiceIzq + IndiceDer)/2]

Si x==A[medio]

Retorna medio

En otro caso

Si x < A[medio]

Retorna BusquedaBinariaRecursiva(A,x,indiceIzquierdo,medio)


En otro caso

Retorna BusquedaBinariaRecursiva(A,x,medio, indiceDerecho)

Fin Si

Fin Si

Fin

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	43/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

En ambos algoritmos en pseudocódigo (iterativo y recursivo) se retorna la posición donde se encuentra el elemento o bien -1 si no se encuentra. También en ambos el tiempo de ejecución es $\theta(\log n)$.


Desarrollo

Actividad 1.1

A continuación, se proporciona la implementación en Python de los pseudocódigos de las funciones para búsqueda lineal presentados en este documento. Se pide realizar un programa que utilice las tres funciones para buscar en una lista dada por el profesor, un valor o llave proporcionado en la entrada estándar. Se debe mostrar el número de iteraciones que realiza cada función, así como el índice donde se localizó la llave. Las funciones en Python son las siguientes:

```
#búsqueda Lineal o secuencial
def busquedaLineal(A,n,x):
    encontrado=-1
    for k in range (n):
        if A[k] == x:
            encontrado=k
    return encontrado
```

```
#Búsqueda Lineal Mejorada
def busquedaLinealMejorada(A,n,x):
    encontrado=-1
    for k in range (n):
        if A[k] == x:
            encontrado=k
            break
    return encontrado
```

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	44/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

```

#Búsqueda Lineal con Centinela
def busquedaLinealCentinela (A, n, x) :
    tmp=A[n]
    A[n]=x
    k=0
    while A[k] != x:
        k=k+1
    print (k)
    A[n]=tmp
    if k < n or A[n]==x:
        return k
    else:
        return -1
    return encontrado

```


Probar su programa con diferentes llaves, y describir el porqué de los resultados obtenidos.

Actividad 1.2

Implementar la función recursiva de la búsqueda lineal utilizando el pseudocódigo proporcionado en este documento.

Actividad 2

Abajo se muestran las funciones en Python de los pseudocódigos para búsqueda binaria (solución iterativa y recursiva) mencionados en esta guía. Se requiere implementar un programa utilizando estas funciones y alguna de las de ordenamiento vistas en las guías anteriores para buscar un valor en la lista proporcionada en la actividad 1.

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	45/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

```
#Búsqueda Binaria Iterativa
import math
def BusquedaBinIter(A,x,izquierda,derecha):


    while izquierda <= derecha:
        medio = math.floor((izquierda+derecha)/2)
        if A[medio] == x:
            return medio
        elif A[medio] < x:
            izquierda = medio+1
        else:
            derecha = medio-1
    return -1
```

```
#Búsqueda binaria Recursiva
import math
def BusquedaBinRecursiva(A,x,izquierda,derecha):
    if izquierda > derecha :
        return -1
    medio = math.floor((izquierda+derecha)/2)
    print (medio)

    if A[medio] == x:
        return medio

    elif A[medio] < x:

        return BusquedaBinRecursiva(A,x,medio+1,derecha)
    else:
        return BusquedaBinRecursiva(A,x,izquierda,medio-1)
```

	Manual de prácticas del Laboratorio de Estructuras de datos y algoritmos II	Código:	MADO-20
		Versión:	01
		Página	46/180
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Actividad 3

Ejercicios propuestos por el profesor.

Referencias

[1] CORMEN, Thomas
Algorithms Unlocked
Cambridge MA, USA
The MIT Press, 2013