N.NIRESH

19ITR054

IT-A

Python programming laboratory

Experiment.no:1

LINEAR SEARCH

AIM:

To perform linear search using python.

SOURCE CODE:

n=int(input("enter the size of list"))

a=[]

print("enter the element:")

for i in range(n):

a.append(int(input()))

print(a)

b=int(input("enter the value to search"))

count=0

for i in range(0,n):

if(b==a[i]):

count=1

if(count==1):

print("\nthe element is present")

else:

print("\nthe element is not present")

OUTPUT:

enter the size of list5

enter the element:

234

987

456

109

56

[234, 987, 456, 109, 56]

enter the value to search456

the element is present

RESULT:

The given program to perform linear search using python was compiled and executed successfully.

BINARY SEARCH

AIM:

To perform binary search using python.

SOURCE CODE:

lst=[]

count=0

n=int(input("Enter the size of the list:"))

for i in range (0,n):

a=int(input("Enter the elements to be inserted"))

lst.append(a)

searchele=int(input("Enter the element to be searched"))

for i in range (0,n):

mid=int(n/2);

if(lst[mid]==searchele):

count=1

j=mid

break

elif(lst[mid]>searchele):

for j in range(0,mid):

if(lst[j]==searchele):

count=1

break

elif(lst[mid]<searchele):

for j in range(mid,n):

if(lst[j]==searchele):

count=1

break

if(count==1):

print("The element is present")

else:

print("The element not found")

OUTPUT:

Enter the size of the list:5

Enter the element to insert:45

Enter the element to insert:987

Enter the element to insert:654

Enter the element to insert:234

Enter the element to insert:567

Enter the element to be searched654

The element is present

RESULT:

The given program to perform binary search using python was compiled and executed successfully.

QUICK SORT

AIM:

To perform quick sort using python.

SOURCE CODE:

def partition(lst,lb,ub):

start=lb

pivot=lst[lb]

end=ub

while start<end:

while start<len-1 and lst[start]<=pivot:

start+=1

while end>=0 and lst[end]>pivot :

end-=1

if start<end:

temp=lst[start]

lst[start]=lst[end]

lst[end]=temp

temp2=lst[lb]

lst[lb]=lst[end]

lst[end]=temp2

return end

def quick(lst,lb,ub):

if lb<ub:

loc=partition(lst,lb,ub)

quick(lst,lb,loc-1)

quick(lst,loc+1,ub)

if lb==0 and ub==len-1:

print("QUICK SORT:\n")

print(lst)

lst=[]

n=int(input("enter the size of the list:"))

for i in range(0,n):

lst.append(int(input(f"enter the element{i+1}:")))

len=len(lst)

quick(lst,0,len-1)

OUTPUT:

enter the size of the list:7

enter the element1:239

enter the element2:563

enter the element3:678

enter the element4:488

enter the element5:542

enter the element6:809

enter the element7:900

QUICK SORT:

[239, 488, 542, 563, 678, 809, 900]

RESULT:

The given program to perform quick sort using python was compiled and executed successfully.

MERGE SORT

AIM:

To perform merge sort using python.

SOURCE CODE:

def mergeSort(lst1):

if len(lst1)>1:

mid = len(lst1)//2

lefthalf = lst1[:mid]

righthalf = lst1[mid:]

mergeSort(lefthalf)

mergeSort(righthalf)

i=j=k=0

while i < len(lefthalf) and j < len(righthalf):

if lefthalf[i] < righthalf[j]:

lst1[k]=lefthalf[i]

i=i+1

else:

lst1[k]=righthalf[j]

j=j+1

k=k+1

while i < len(lefthalf):

lst1[k]=lefthalf[i]

i=i+1

k=k+1

while j < len(righthalf):

lst1[k]=righthalf[j]

j=j+1

k=k+1

lst1=[]

num=int(input("Enter the size of the list"))

for n in range(0,num):

ele=int(input("Enter the element to insert:"))

lst1.append(ele)

mergeSort(lst1)

print("MERGE SORT")

print(lst1)

OUTPUT:

Enter the size of the list5

Enter the element to insert:324

Enter the element to insert:986

Enter the element to insert:452

Enter the element to insert:123

Enter the element to insert:564

MERGE SORT

[123, 324, 452, 564, 986]

RESULT:

The given program to perform merge sort using python was compiled and executed successfully.