**PRACTICAL NO.8**

**Aim: -** W.A.P to implement various kinds of searching and sorting algorithm.

**Algorithm and Flowchart:**

**#Linear Search**

Linear Search ( Array A, Value x)

Step 1: Set i to 1

Step 2: if i > n then go to step 7

Step 3: if A[i] = x then go to step 6

Step 4: Set i to i + 1

Step 5: Go to Step 2

Step 6: Print Element x Found at index i and go to step 8

Step 7: Print element not found

Step 8: Exit

I:=LB And LOC:=0

I<=UB And A[I]!=ITEM

F F

ITEM not Found

ITEM found at LOC

LOC!=0

I:=I+1

A[I]=ITEM and LOC:=1

**#Binary Search**

Step 1 − Start searching **data** from middle of the list.

Step 2 − If it is a match, return the index of the item, and exit.

Step 3 − If it is not a match, probe position.

Step 4 − Divide the list using probing formula and find the new midle.

Step 5 − If data is greater than middle, search in higher sub-list.

Step 6 − If data is smaller than middle, search in lower sub-list.

Step 7 − Repeat until match

BEG :=LB , End:=UB and MID:=int[(lb+ub)/2]

IF ITEM<A[MID]

BEG<=END && ITEM!=A[MID]

SET LOC:=MID

LOC:=NULL

Set MID:=INT[(BEG+END)/2]

BEG:=MI+1

END :=MID-1

ITEM<A[MID]

**#Insertion sort**

**Step 1** − If it is the first element, it is already sorted. return 1;

**Step 2** − Pick next element

**Step 3** − Compare with all elements in the sorted sub-list

**Step 4** − Shift all the elements in the sorted sub-list that is greater than the

value to be sorted

**Step 5** − Insert the value

**Step 6** –

Repeat until list is sorted

I:=1

I<=n-1

J:=1

J<=n-k

J:=J-1

Interchange A[i] with A[i+1]

A[j]>A[j+1]

**#Bubble sort**

1. Compare the first and the second element of the list and swap them if they are in wrong order.
2. Compare the second and the third element of the list and swap them if they are in wrong order.
3. Proceed till the last element of the list in a similar fashion.
4. Repeat all of the above steps until the list is sorted.

I:=1

I<=n-1

J:=1

J<=n-k

Interchange A[i] with A[i+1]

A[j]>A[j+1]

**#Selection Sort**

**Step 1** − Set MIN to location 0

**Step 2** − Search the minimum element in the list

**Step 3** − Swap with value at location MIN

**Step 4** − Increment MIN to point to next element

**Step 5** − Repeat until list is sorted

K:=1

K<=n-1

MIN:=A[K] POS:=K

P

PP

I:=K+1

I<N

MIN:= A[i] POS:=I

IF(MIN>A[I])

### #Merge Sort

1. algorithm Merge\_Sort(list)
2. Pre: list 6= fi
3. Post: list has been sorted into values of ascending order
4. if list.Count = 1 // when already sorted
5. return list
6. end if
7. m <- list.Count = 2
8. left <- list(m)
9. right <- list(list.Count - m)
10. for i <- 0 to left.Count - 1
11. left[i] <- list[i]
12. end for
13. for i <- 0 to right.Count -1
14. right[i] <- list[i]
15. end for
16. left <- Merg\_Sort(left)
17. 17) right <- Merge\_Sort(right)
18. 18) return MergeOrdered(left, right)
19. 19) end Merge\_Sort

**Set p=beg, q=mid**

**R=mid+1, s=end and j=0**

**p<=q**

**A[p]=A[r]**

**A[p]>A[r]**

**A[p]<A[r]**

**B[j]=A[p]**

**P=p+1 and j=j+1**

**B[j]=A[r]**

**R=r+1 and j=j+1**

**B[j]=A[r]**

**R=r+1 and j=j+1**

**B[j]=A[p]**

**P=p+1 and j=j+1**

**R<S**

**P<Q**

**R<=s**

**P<=q**

**B[j]=A[r]**

**R=r+1 and j=j+1**

B**[j]=A[p]**

**P=p+1 and j=j+1**

**Source Code:-**

**#Linear and Binary search**

print("SEARCHINGGGG")

arr=[]

n=int(input('Enter the number of terms in the list : '))

z=0

for i in range(0,n):

print('Enter the item number : ',i+1)

z=int(input('item : '))

arr.append(z)

print("given array is : ",arr)

print("linear search result is : ")

x=int(input("the value to be searched : "))

for i in range(0,n):

if(arr[i]==x):

print('the entered number is present on position :',i+1)

print("binary search result is : ")

beg=0

end=len(arr)-1

loc=0

mid=int((beg+end)/2)

while(beg<=end and x!=arr[mid]):

if(x<arr[mid]):

end=mid-1

else:

beg=mid+1

mid=int((beg+end)/2)

if(x==arr[mid]):

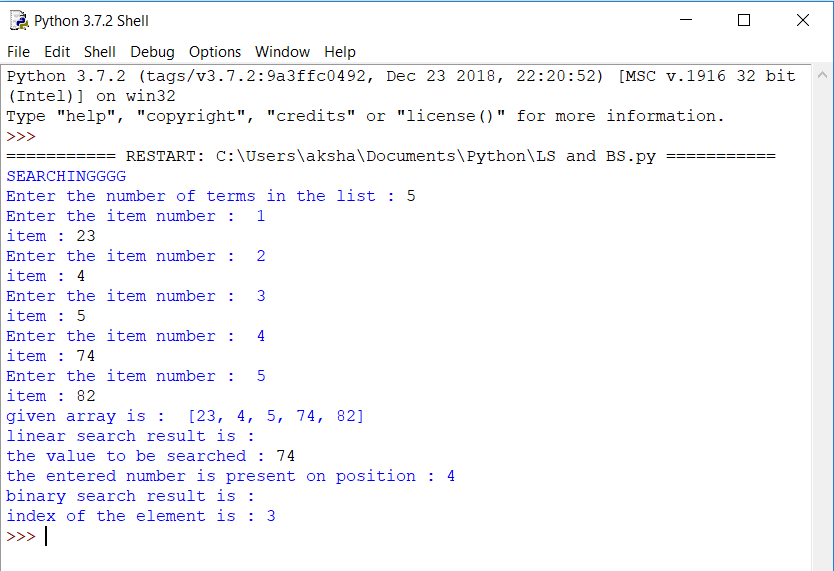
loc=mid

print("index of the element is :",loc)

else:

loc=0

print("Number doesn't Exist !")



**#Bubble Sort**

a=[]

n=int(input('Enter the total number of elements : '))

x=0

for i in range(0,n):

print('Enter the item number : ',i+1)

x=int(input('Enter item : '))

a.append(x)

print(a)

temp=0

for k in range(0,len(a)-1):

for i in range(0,len(a)-(k+1)):

if(a[i]>a[i+1]):

temp=a[i]

a[i]=a[i+1]

a[i+1]=temp

print(a)

A screenshot of a cell phone

Description automatically generated

**#Selection sort**

list=[]

n=int(input('Enter the total number of elements : '))

x=0

for i in range(0,n):

print('Enter the item number : ',i+1)

x=int(input('Enter item : '))

list.append(x)

print(list)

#-------------------------------------------------------------------

temp=0

for k in range(0,len(list)-1):

min=list[k]

pos=k

for i in range(k+1,len(list)):

if(min>list[i]):

min=list[i]

pos=i

if(pos!=k):

temp=list[k]

list[k]=list[pos]

list[pos]=temp

print(list)

A screenshot of a social media post

Description automatically generated

**#Insertion Sort**

a=[]

n=int(input('Enter the number of elements : '))

x=0

for i in range(0,n):

print('Enter the item number : ',i+1)

x=int(input('Enter item : '))

a.append(x)

print(a)

for k in range(1,len(a)):

temp= a[k]

i=k-1

while((temp<a[i]) and (i>=0)):

a[i+1]=a[i]

i-=1

a[i+1]=temp

print(a)

A screenshot of a cell phone

Description automatically generated

**#Merge Sort**

def merge(arr, l, m, r):

n1 = m - l + 1

n2 = r- m

L = [0] \* (n1)

R = [0] \* (n2)

for i in range(0 , n1):

L[i] = arr[l + i]

for j in range(0 , n2):

R[j] = arr[m + 1 + j]

i = 0

j = 0

k = l

while i < n1 and j < n2 :

if L[i] <= R[j]:

arr[k] = L[i]

i += 1

else:

arr[k] = R[j]

j += 1

k += 1

while i < n1:

arr[k] = L[i]

i += 1

k += 1

while j < n2:

arr[k] = R[j]

j += 1

k += 1

def mergeSort(arr,l,r):

if l < r:

m = int((l+(r-1))/2)

mergeSort(arr, l, m)

mergeSort(arr, m+1, r)

merge(arr, l, m, r)

arr=[]

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

print("Enter the elements of an list")

for i in range(n):

arr.append(input())

mergeSort(arr,0,n-1)

print("Sorted list is: ")

print(arr)

A screenshot of a social media post

Description automatically generated