1. def binary\_search(arr,target):

    start=0

    end=len(arr)-1

    result=-1

    while(start<=end):

#You update start and end on each iteration based on comparisons.

#Since mid depends on both start and end, it must be recalculated every time they change.

        mid=(start+end)//2

        if arr[mid]==target:

            result=mid

            end=mid-1 # But if the array has multiple occurrences of k, we want the first (smallest) index, not just any one.

        elif target>arr[mid]:

            start=mid+1

        else:

            end=mid-1

        mid=(start+end)//2

    return result

arr=[1,1,1,1,2,2,2,5,6,7,8,9,10,11]

target=2

result=binary\_search(arr,target)

print(result)

1. def insertion\_sort(arr):

    for i in range(1,len(arr)):

        key=arr[i]

        j=i-1

        while j>=0 and arr[j]>key:#if this fails then arr[j+1]=key this is executed directly

            arr[j+1]=arr[j]#responsible for shifting the array's

            j-=1 #move to the previous element

        arr[j+1]=key

    return arr

arr=[23,63,44,57]

ans=insertion\_sort(arr)

print(ans)

1. def merge\_sort(arr,start=0,end=None):

    if end is None:

        end=len(arr)-1

    if start<end: #When to start and where to stop

        mid=(start+end)//2

        #Recursively sort the first half

        merge\_sort(arr,start,mid)

        #Recursively sort the second half

        merge\_sort(arr,mid+1,end)

        #Merge the srted array

        merge(arr,start,mid,end)

def merge(arr,start,mid,end):

    left = arr[start:mid+1]#arr[0:2]

    right = arr[mid+1:end+1]#arr[3:4]

    i=j=0

    k=start

    while i<len(left) and j<len(right):#this loop will occur until it reaches the end of the array

        if left[i]<=right[j]:# arr[0]<=arr[0] 30<=49 so this 30 is stored in the arr[k]

            arr[k]=left[i]

            i+=1

        else:

            arr[k] = right[j]

            j+=1

        k+=1

#this is written to take care of the edge elements.

#Bcoz when both satisfies in the above while loop will be exexuted in the form of recusiion.

#when i=2 and j=2 then that while loop works after that when i=3 and j=2 it dosent work.

    while i<len(left):

        arr[k]=left[i]

        i+=1

        k+=1

    while j<len(right):

        arr[k]=right[j]

        j+=1

        k+=1

arr=[8, 4, 5, 2, 9, 1]

merge\_sort(arr)

print(arr)

'''

[1, 2, 4, 5, 8, 9]

'''

1. def bubble\_sort(arr):

    n=len(arr)

    for i in range(n):

        for j in range(0,n-i-1):#(0,5)

            if arr[j]>arr[j+1]:#arr[0]>arr[1],45>8(as this condition is true it enters into the loop)

                arr[j],arr[j+1]=arr[j+1],arr[j]#arr[0](45)=(8)arr[1](so here equals too means it swaps with each other)

    return arr

arr=[45,8,9,1,78,45]

ans=bubble\_sort(arr)

print(ans)

'''

[1, 8, 9, 45, 45, 78]

'''