BUBBLE SORT

In [12]:

```
#l=[2,1,4,5,6]
def bubblesort(l):
    for i in range(len(l)-1):
        for j in range(i+1,len(l)):
            if l[i]>l[j]:
                temp=l[i]
                l[i]=l[j]
                l[j]=temp
l=[2,1,4,5,6]
bubblesort(l)
print(l)
```

```
[1, 2, 4, 5, 6]
```

In [17]:

```
def bubblesort(list):

# Swap the elements to arrange in order
   for iter_num in range(len(list)-1,0,-1):
        for idx in range(iter_num):
            if list[idx]>list[idx+1]:
                temp = list[idx]
                list[idx] = list[idx+1]
                list[idx+1] = temp
list = [19,2,31,45,6,11,121,27]
bubblesort(list)
print(list)
```

```
[2, 6, 11, 19, 27, 31, 45, 121]
```

MERGE SORT

In [18]:

```
def merge_sort(unsorted_list):
    if len(unsorted_list) <= 1:
        return unsorted_list
# Find the middle point and devide it
    middle = len(unsorted_list) // 2
    left_list = unsorted_list[:middle]
    right_list = unsorted_list[middle:]
   # print(left_list,right_list)

    left_list = merge_sort(left_list)
    right_list = merge_sort(right_list)
    print(left_list,right_list)
    return list(merge(left_list, right_list))

# Merge the sorted halves
def merge(left_half,right_half):
    res = []
    while len(left_half) != 0 and len(right_half) != 0:
        if left_half[0] < right_half[0]:
            res.append(left_half[0])
            left_half.remove(left_half[0])
        else:
            res.append(right_half[0])
            right_half.remove(right_half[0])
    if len(left_half) == 0:
        res = res + right_half
    else:
        res = res + left_half
    return res
unsorted_list = [64, 34, 25, 12, 22, 11, 90]
print("sorted list",merge_sort(unsorted_list))
```

```
[34] [25]
[64] [25, 34]
[12] [22]
[11] [90]
[12, 22] [11, 90]
[25, 34, 64] [11, 12, 22, 90]
sorted list [11, 12, 22, 25, 34, 64, 90]
```

INSERTION SORT

In [17]:

```
def insertionSort(arr):

    # Traverse through 1 to len(arr)
    for i in range(1, len(arr)):

        key = arr[i]

        # Move elements of arr[0..i-1], that are
        # greater than key, to one position ahead
        # of their current position
        j = i-1
        while j >=0 and key < arr[j] :
                arr[j+1] = arr[j]
                j -= 1
        arr[j+1] = key
arr=[1,7,62,44]
insertionSort(arr)
print ("Sorted array is:",arr)
```

```
Sorted array is: [1, 7, 44, 62]
```

LINEAR SEARCH

In [23]:

```
def linear_search(values, search_for):
    search_at = 0
    search_res = False
# Match the value with each data element
    while search_at < len(values) and search_res is False:
        if values[search_at] == search_for:
            search_res = True
        else:
            search_at = search_at + 1
        return search_res
l = [64, 34, 25, 12, 22, 11, 90]
print(linear_search(l, 12))
print(linear_search(l, 91))
```

```
False
False
```

In [ ]:

In [15]:

```
#l=[2,1,4,5,6]
def bubblesort(l):
    for i in range(0,len(l)):
        for j in range(i+1,len(l)):
            if l[i]>l[j]:
                temp=l[i]
                l[i]=l[j]
                l[j]=temp
l=[2,1,4,5,6,0]
bubblesort(l)
print(l)
```

```
[0, 1, 2, 4, 5, 6]
```

In [1]:

```
def insertion_sort(InputList):
   for i in range(1, len(InputList)):
      j = i-1
      nxt_element = InputList[i]
# Compare the current element with next one
   while (InputList[j] > nxt_element) and (j >= 0):
      InputList[j+1] = InputList[j]
      j=j-1
   InputList[j+1] = nxt_element
list = [19,2,31,45,30,11,121,27]
insertion_sort(list)
print(list)
```

```
[19, 2, 31, 45, 30, 11, 27, 121]
```

In [ ]: