**Algorithms:**

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)

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:]

left\_list = merge\_sort(left\_list)

right\_list = merge\_sort(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(merge\_sort(unsorted\_list))

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)

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))

**RANGES:**

Range() function it gives a sequence of numbers based on the start and stop index given.

for i in range(10):

print(i, end =" ")

for i in range(3, 10, 2):

print(i, end =" ")