1. Python Program for Recursive Insertion Sort

def insertionSortRecursive(arr,n):

if n<=1:

return

insertionSortRecursive(arr,n-1)

'''Insert last element at its correct position

in sorted array.'''

last = arr[n-1]

j = n-2

while (j>=0 and arr[j]>last):

arr[j+1] = arr[j]

j = j-1

arr[j+1]=last

def printArray(arr,n):

for i in range(n):

print(arr[i],end=" ")

arr = [12,11,13,5,6]

n = len(arr)

insertionSortRecursive(arr, n)

printArray(arr, n)

1. Python Program for QuickSort

def partition(array, low, high):

pivot = array[high]

i = low - 1

for j in range(low, high):

if array[j] <= pivot:

i = i + 1

(array[i], array[j]) = (array[j], array[i])

(array[i + 1], array[high]) = (array[high], array[i + 1])

return i + 1

def quickSort(array, low, high):

if low < high:

pi = partition(array, low, high)

quickSort(array, low, pi - 1)

quickSort(array, pi + 1, high)

data = [1, 7, 4, 1, 10, 9, -2]

print("Unsorted Array")

print(data)

size = len(data)

quickSort(data, 0, size - 1)

print('Sorted Array in Ascending Order:')

print(data)

1. Python Program for Iterative Quick Sort

def partition(arr,l,h):

i = ( l - 1 )

x = arr[h]

for j in range(l , h):

if arr[j] <= x:

i = i+1

arr[i],arr[j] = arr[j],arr[i]

arr[i+1],arr[h] = arr[h],arr[i+1]

return (i+1)

def quickSortIterative(arr,l,h):

size = h - l + 1

stack = [0] \* (size)

top = -1

top = top + 1

stack[top] = l

top = top + 1

stack[top] = h

while top >= 0:

h = stack[top]

top = top - 1

l = stack[top]

top = top - 1

p = partition( arr, l, h )

if p-1 > l:

top = top + 1

stack[top] = l

top = top + 1

stack[top] = p - 1

if p+1 < h:

top = top + 1

stack[top] = p + 1

top = top + 1

stack[top] = h

arr = [4, 3, 5, 2, 1, 3, 2, 3]

n = len(arr)

quickSortIterative(arr, 0, n-1)

print ("Sorted array is:")

for i in range(n):

print ("%d" %arr[i])

1. Python Program for Selection Sort

def selectionSort(array, size):

for ind in range(size):

min\_index = ind

for j in range(ind + 1, size):

if array[j] < array[min\_index]:

min\_index = j

(array[ind], array[min\_index]) = (array[min\_index], array[ind])

arr = [-2, 45, 0, 11, -9,88,-97,-202,747]

size = len(arr)

selectionSort(arr, size)

print('The array after sorting in Ascending Order by selection sort is:')

print(arr)

1. Python Program for Bubble Sort

def bubbleSort(arr):

n = len(arr)

swapped = False

for i in range(n-1):

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

if arr[j] > arr[j + 1]:

swapped = True

arr[j], arr[j + 1] = arr[j + 1], arr[j]

if not swapped:

return

arr = [64, 34, 25, 12, 22, 11, 90]

bubbleSort(arr)

print("Sorted array is:")

for i in range(len(arr)):

print("% d" % arr[i], end=" ")

1. Python Program for 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 = l+(r-l)//2

mergeSort(arr, l, m)

mergeSort(arr, m+1, r)

merge(arr, l, m, r)

arr = [12, 11, 13, 5, 6, 7]

n = len(arr)

print("Given array is")

for i in range(n):

print("%d" % arr[i],end=" ")

mergeSort(arr, 0, n-1)

print("\n\nSorted array is")

for i in range(n):

print("%d" % arr[i],end=" ")

1. Python Program for Iterative Merge Sort

def merge(S, temp, From, mid, to):

a = From

b = From

c = mid + 1

while b <= mid and c <= to:

if S[b] < S[c]:

temp[a] = S[b]

b = b + 1

else:

temp[a] = S[c]

c = c + 1

a = a + 1

while b < len(S) and b <= mid:

temp[a] = S[b]

a = a + 1

b = b + 1

for b in range(From, to + 1):

S[b] = temp[b]

1. Python Program for Heap Sort

def heapify(arr, n, i):

largest = i # Initialize largest as root

l = 2 \* i + 1 # left = 2\*i + 1

r = 2 \* i + 2 # right = 2\*i + 2

if l < n and arr[i] < arr[l]:

largest = l

if r < n and arr[largest] < arr[r]:

largest = r

if largest != i:

(arr[i], arr[largest]) = (arr[largest], arr[i]) # swap

heapify(arr, n, largest)

def heapSort(arr):

n = len(arr)

for i in range(n // 2 - 1, -1, -1):

heapify(arr, n, i)

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

(arr[i], arr[0]) = (arr[0], arr[i]) # swap

heapify(arr, i, 0)

arr = [12, 11, 13, 5, 6, 7, ]

heapSort(arr)

n = len(arr)

print('Sorted array is')

for i in range(n):

print(arr[i])

1. Python Program for Counting Sort

def countSort(arr):

output = [0 for i in range(256)]

count = [0 for i in range(256)]

ans = ["" for \_ in arr]

for i in arr:

count[ord(i)] += 1

for i in range(256):

count[i] += count[i-1]

for i in range(len(arr)):

output[count[ord(arr[i])]-1] = arr[i]

count[ord(arr[i])] -= 1

for i in range(len(arr)):

ans[i] = output[i]

return ans

arr = "geeksforgeeks"

ans = countSort(arr)

print ("Sorted character array is %s" %("".join(ans)))

1. Python Program for ShellSort

def shellSort(arr):

n = len(arr)

gap = n/2

while gap > 0:

for i in range(gap,n):

temp = arr[i]

j = i

while j >= gap and arr[j-gap] >temp:

arr[j] = arr[j-gap]

j -= gap

arr[j] = temp

gap /= 2

arr = [ 12, 34, 54, 2, 3]

n = len(arr)

print ("Array before sorting:")

for i in range(n):

print(arr[i]),

shellSort(arr)

print ("\nArray after sorting:")

for i in range(n):

print(arr[i]),