**SORTING ALGORITHMS SET 2**

**RIZWANA RASHEED**

**AM.SC.P2CSC19026**

**a)MERGE SORT**

def mergeSort(list):

#print("Splitting ",list)

if len(list)>1:

mid = len(list)//2

lefthalf = list[:mid]

righthalf = list[mid:]

mergeSort(lefthalf)

mergeSort(righthalf)

i=j=k=0

while i < len(lefthalf) and j < len(righthalf):

if lefthalf[i] < righthalf[j]:

list[k]=lefthalf[i]

i=i+1

else:

list[k]=righthalf[j]

j=j+1

k=k+1

while i < len(lefthalf):

list[k]=lefthalf[i]

i=i+1

k=k+1

while j < len(righthalf):

list[k]=righthalf[j]

j=j+1

k=k+1

#print("Merging ",list)

list = []

size=int(input("ENTER THE SIZE "))

for x in range(size):

elements=int(input())

list.append(elements)

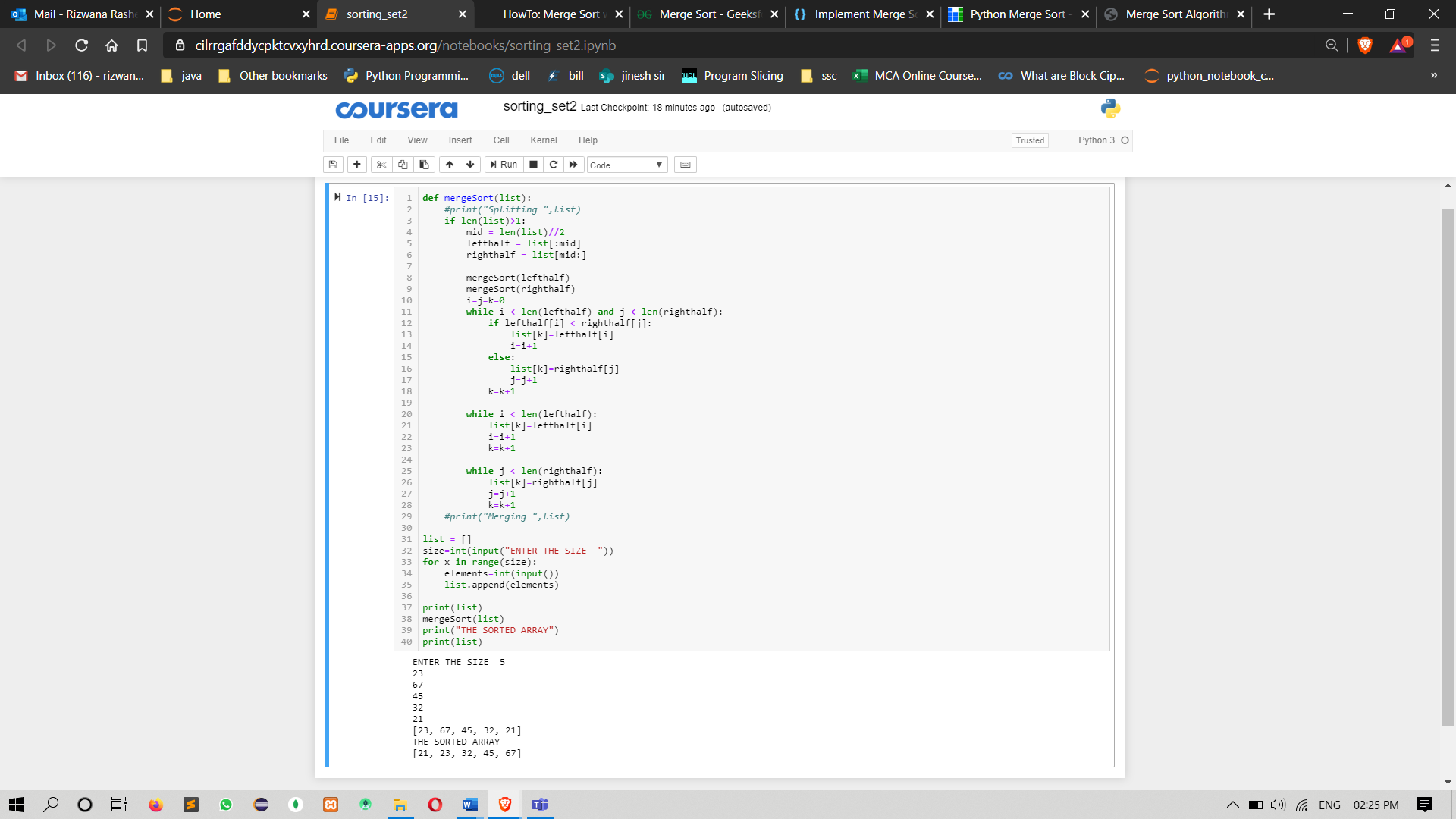
print(list)

mergeSort(list)

print("THE SORTED ARRAY")

print(list)

**OUTPUT**



**a)QUICK SORT**

def partition(sort\_list, low, high):

i = (low -1)

pivot = sort\_list[high]

for j in range(low, high):

if sort\_list[j] <= pivot:

i += 1

sort\_list[i], sort\_list[j] = sort\_list[j], sort\_list[i]

sort\_list[i+1],sort\_list[high] = sort\_list[high], sort\_list[i+1]

return (i+1)

def quick\_sort(sort\_list, low, high):

if low < high:

pi = partition(sort\_list, low, high)

quick\_sort(sort\_list, low, pi-1)

quick\_sort(sort\_list, pi+1, high)

lst = []

size = int(input("Enter size of the list: "))

for i in range(size):

elements = int(input())

lst.append(elements)

print(lst)

low = 0

high = len(lst) - 1

quick\_sort(lst, low, high)

print("The sorted array is:")

print(lst)

**OUTPUT**

