AOA Assignment 2

Keegan Vaz
192120
SECMPNB
42

Aim: Implement **Merge Sort** and **Quick Sort** algorithms and derive its time complexity.

Problem statement: Write a menu driven program to implement the Mergesort and Quicksort algorithm. Input at least 10 values separately to each algorithm.

For Quick sort: Display the array after every partition. And final sorted array
For Mergesort: Display the array after every merge operation, and the final sorted array.

Code:

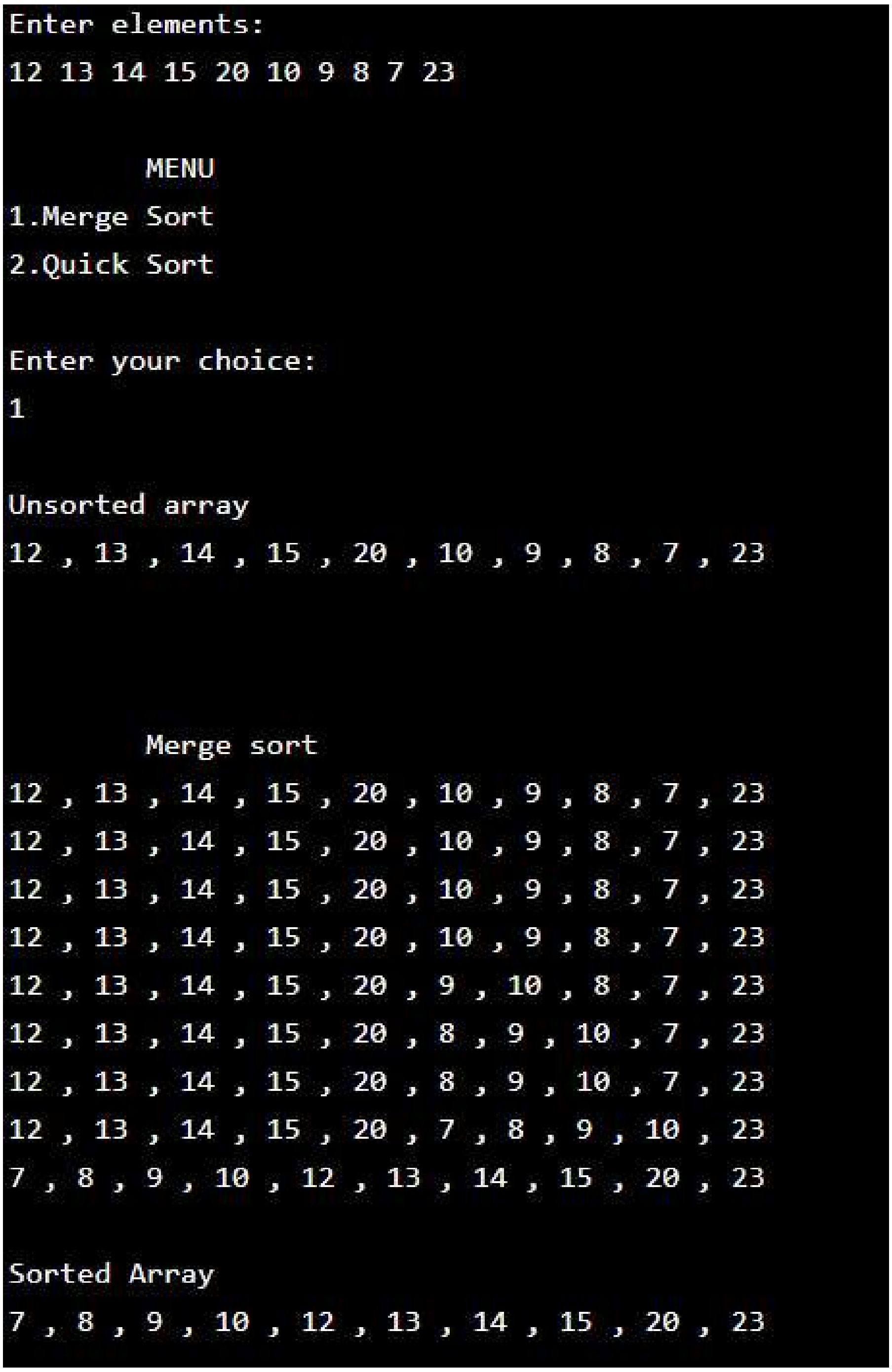
```
def disp(array):
   print(*array, sep=' , ')

def merge(array, l , m, h):
   n1,n2 = m-l+1 , h-m

   #declaring temp arrays and copying elements into temp arrays
   L,R= array[l : m+1] , array[m+1 : h+1]

   #merge temp arrays back to one array
   i,j,k = 0,0,l
   while i<n1 and j<n2:
      if L[i] <= R[j]:
         array[k] = L[i]
         i+=1
      else:
         array[k] = R[j]
         j+=1
      k+=1

   #copy remaining element of arrays L and R
   while i<n1:
      array[k]=L[i]
      i+=1
      k+=1
   while j<n2:
      array[k]=R[j]
      j+=1
      k+=1

def mergeSort(array , l, h):
    if l<h:
        m =(l+h)//2
        mergeSort(array , l , m)
        mergeSort(array , m+1 , h)
        merge(array , l , m, h)
        disp(array)

def partition(array , l , h):
    pivot , i = array[h] , l-1
    for j in range(l,h):
        if array[j]<=pivot:
            i+=1
            array[i] , array[j] = array[j] , array[i]
    array[i+1] , array[h] = array[h] , array[i+1]
    return (i+1)


def quickSort(array , l, h):
    if l<h:
        p = partition(array , l , h)
        quickSort(array , l , p-1)
        quickSort(array , p+1, h)
        disp(array)



inpt = input("\nEnter elements: \n")
array = list(map(int , inpt.split()))
n = len(array)

    #Menu and input choice
ch  = int(input("\n\tMENU\n1.Merge Sort\n2.Quick Sort\n\nEnter your choice: "))

print("\nUnsorted array")
disp(array)
print("\n")

if ch == 1:
    print("\n\tMerge sort")
    mergeSort(array,0,n-1)
    print("\nSorted Array")
    disp(array)

else:
    print("\n\tQuick sort")
    quickSort(array,0,n-1)
    print("\nSorted Array")
    disp(array)
```

AOA Assignment 2

Keegan Vaz
192120
SECMPNB
42

Output:

1. Merge Sort

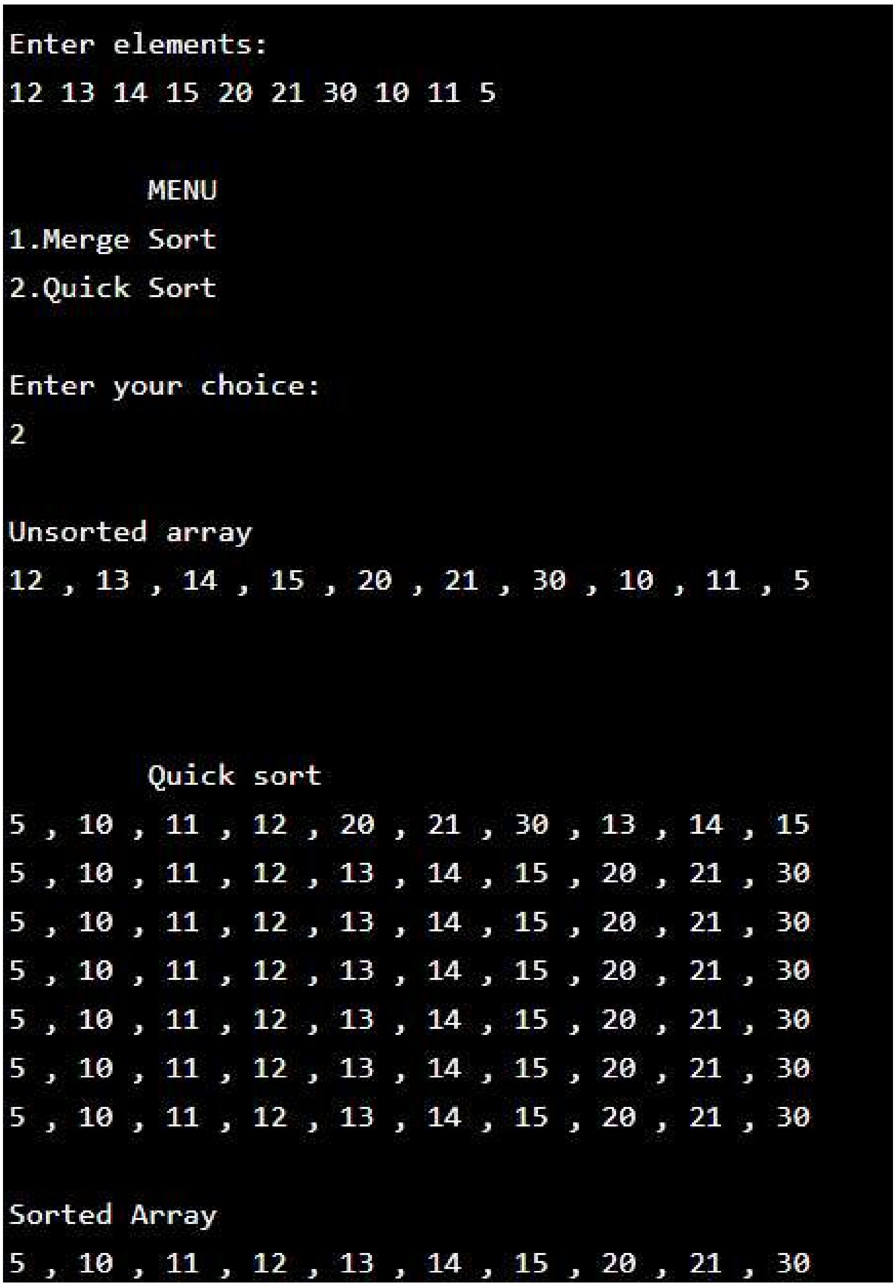


```
Enter elements:
12 13 14 15 20 10 9 8 7 23

        MENU
1.Merge Sort
2.Quick Sort

Enter your choice:
1

Unsorted array
12 , 13 , 14 , 15 , 20 , 10 , 9 , 8 , 7 , 23



        Merge sort
12 , 13 , 14 , 15 , 20 , 10 , 9 , 8 , 7 , 23
12 , 13 , 14 , 15 , 20 , 10 , 9 , 8 , 7 , 23
12 , 13 , 14 , 15 , 20 , 10 , 9 , 8 , 7 , 23
12 , 13 , 14 , 15 , 20 , 10 , 9 , 8 , 7 , 23
12 , 13 , 14 , 15 , 20 , 9 , 10 , 8 , 7 , 23
12 , 13 , 14 , 15 , 20 , 8 , 9 , 10 , 7 , 23
12 , 13 , 14 , 15 , 20 , 8 , 9 , 10 , 7 , 23
12 , 13 , 14 , 15 , 20 , 7 , 8 , 9 , 10 , 23
7 , 8 , 9 , 10 , 12 , 13 , 14 , 15 , 20 , 23

Sorted Array
7 , 8 , 9 , 10 , 12 , 13 , 14 , 15 , 20 , 23
```

AOA Assignment 2

Keegan Vaz
192120
SECMPNB
42

2. Quick Sort



```
Enter elements:
12 13 14 15 20 21 30 10 11 5

        MENU
1.Merge Sort
2.Quick Sort

Enter your choice:
2

Unsorted array
12 , 13 , 14 , 15 , 20 , 21 , 30 , 10 , 11 , 5



        Quick sort
5 , 10 , 11 , 12 , 20 , 21 , 30 , 13 , 14 , 15
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30

Sorted Array
5 , 10 , 11 , 12 , 13 , 14 , 15 , 20 , 21 , 30
```