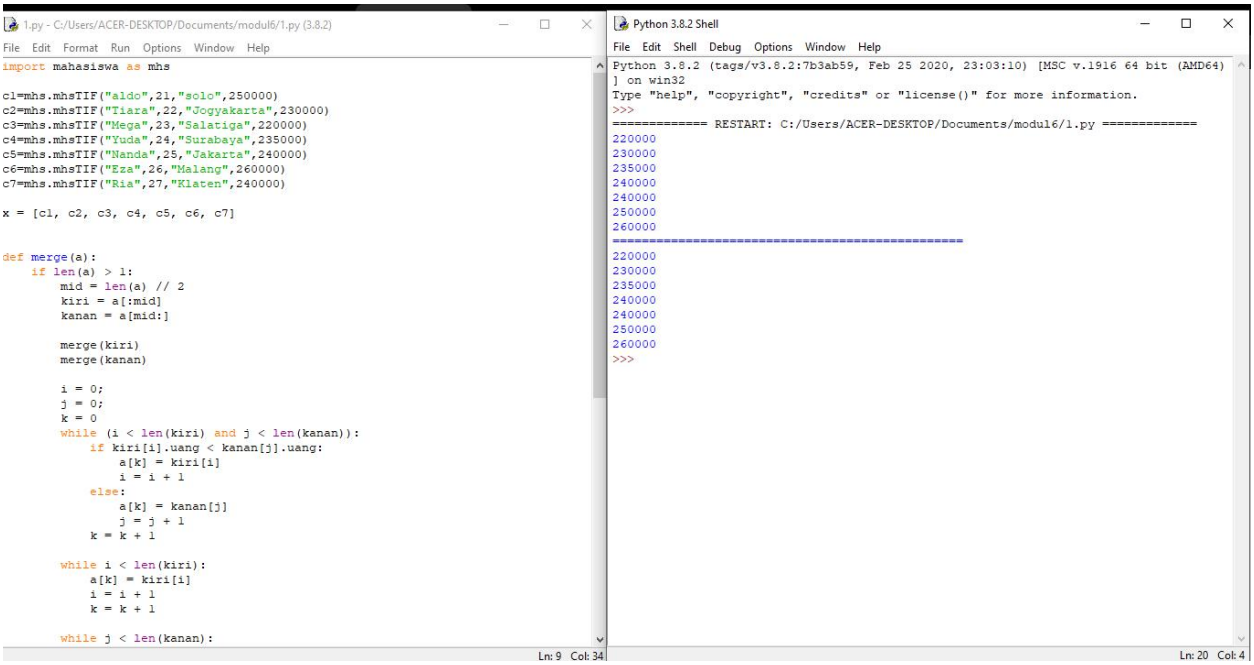
Nama : Fannisa Rif'ani M

Kelas : B

NIM : L200180048

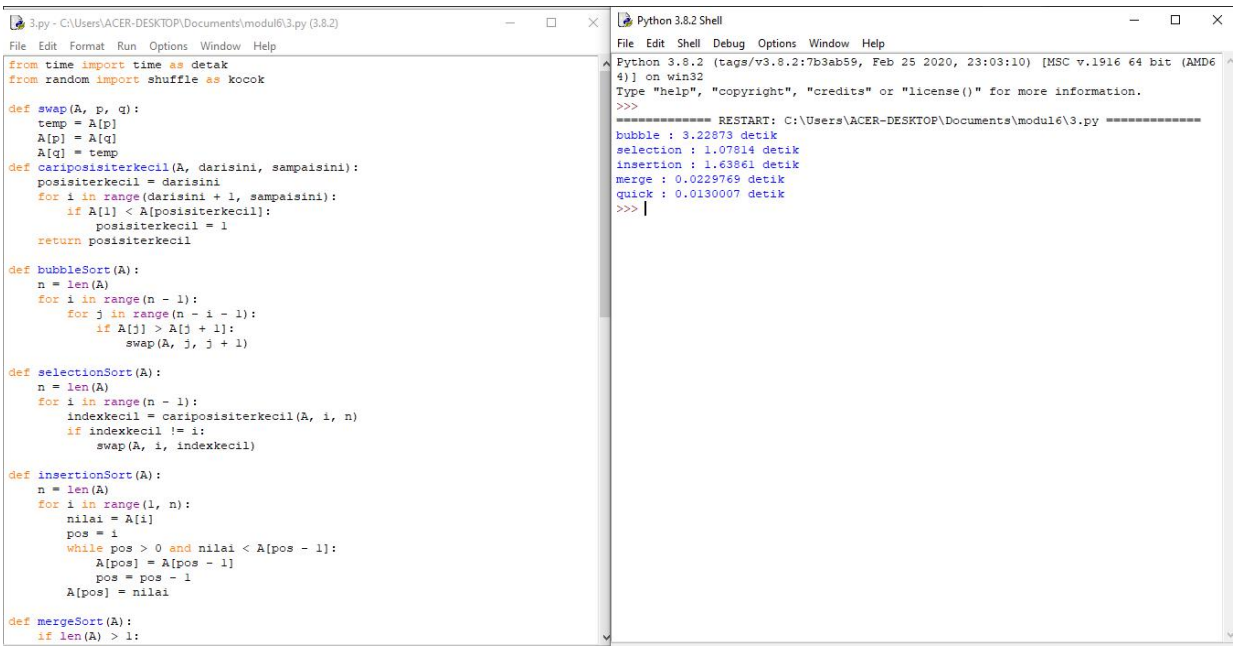
1.



```
import mahasiswa as mhs

c1=mhs.mhsTIF("aldo",21,"solo",250000)
c2=mhs.mhsTIF("Tiara",22,"Jogyakarta",230000)
c3=mhs.mhsTIF("Mega",23,"Salatiga",220000)
c4=mhs.mhsTIF("Yuda",24,"Surabaya",235000)
c5=mhs.mhsTIF("Nanda",25,"Jakarta",240000)
c6=mhs.mhsTIF("Eza",26,"Malang",260000)
c7=mhs.mhsTIF("Ria",27,"Klaten",240000)

x = [c1, c2, c3, c4, c5, c6, c7]


def merge(a):
    if len(a) > 1:
        mid = len(a) // 2
        kiri = a[:mid]
        kanan = a[mid:]

        merge(kiri)
        merge(kanan)

        i = 0;
        j = 0;
        k = 0
        while (i < len(kiri) and j < len(kanan)):
            if kiri[i].uang < kanan[j].uang:
                a[k] = kiri[i]
                i = i + 1
            else:
                a[k] = kanan[j]
                j = j + 1
            k = k + 1

        while i < len(kiri):
            a[k] = kiri[i]
            i = i + 1
            k = k + 1

        while j < len(kanan):
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)
] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/ACER-DESKTOP/Documents/modul6/1.py ==========
220000
230000
235000
240000
240000
250000
260000
================================================
220000
230000
235000
240000
240000
250000
260000
>>>
```

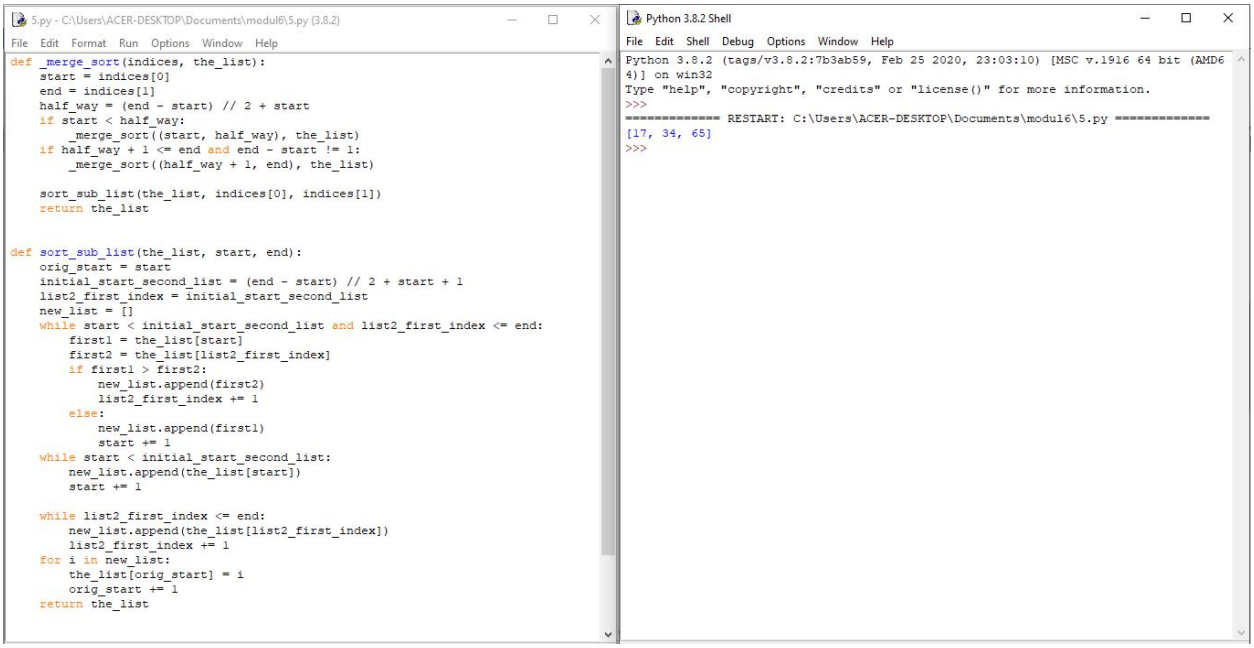
3.



```
from time import time as detak
from random import shuffle as kocok

def swap(A, p, q):
    temp = A[p]
    A[p] = A[q]
    A[q] = temp
def cariposisiterkecil(A, darisini, sampaisini):
    posisiterkecil = darisini
    for i in range(darisini + 1, sampaisini):
        if A[i] < A[posisiterkecil]:
            posisiterkecil = i
    return posisiterkecil

def bubbleSort(A):
    n = len(A)
    for i in range(n - 1):
        for j in range(n - i - 1):
            if A[j] > A[j + 1]:
                swap(A, j, j + 1)

def selectionSort(A):
    n = len(A)
    for i in range(n - 1):
        indexkecil = cariposisiterkecil(A, i, n)
        if indexkecil != i:
            swap(A, i, indexkecil)

def insertionSort(A):
    n = len(A)
    for i in range(1, n):
        nilai = A[i]
        pos = i
        while pos > 0 and nilai < A[pos - 1]:
            A[pos] = A[pos - 1]
            pos = pos - 1
        A[pos] = nilai

def mergeSort(A):
    if len(A) > 1:
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ACER-DESKTOP\Documents\modul6\3.py ==========
bubble : 3.22873 detik
selection : 1.07814 detik
insertion : 1.63861 detik
merge : 0.0229769 detik
quick : 0.0130007 detik
>>>
```

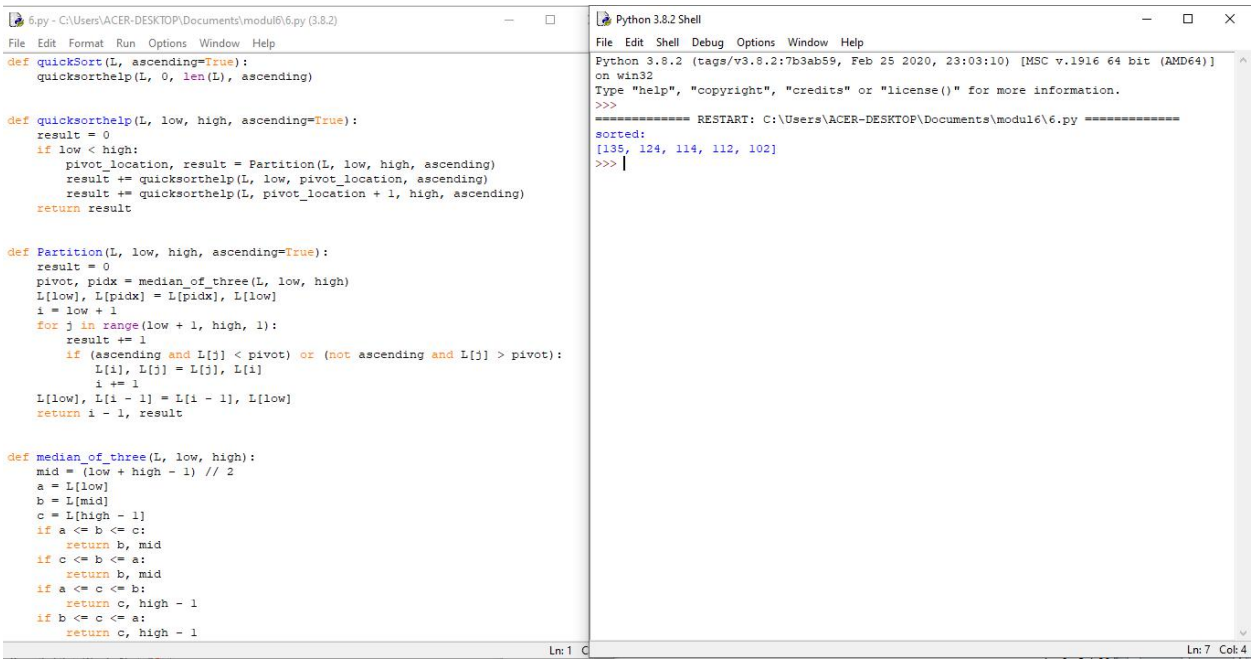
5.



```
5.py - C:\Users\ACER-DESKTOP\Documents\modul6\5.py (3.8.2)
File  Edit  Format  Run  Options  Window  Help
def _merge_sort(indices, the_list):
    start = indices[0]
    end = indices[1]
    half_way = (end - start) // 2 + start
    if start < half_way:
        _merge_sort((start, half_way), the_list)
    if half_way + 1 <= end and end - start != 1:
        _merge_sort((half_way + 1, end), the_list)

    sort_sub_list(the_list, indices[0], indices[1])
    return the_list


def sort_sub_list(the_list, start, end):
    orig_start = start
    initial_start_second_list = (end - start) // 2 + start + 1
    list2_first_index = initial_start_second_list
    new_list = []
    while start < initial_start_second_list and list2_first_index <= end:
        first1 = the_list[start]
        first2 = the_list[list2_first_index]
        if first1 > first2:
            new_list.append(first2)
            list2_first_index += 1
        else:
            new_list.append(first1)
            start += 1
    while start < initial_start_second_list:
        new_list.append(the_list[start])
        start += 1

    while list2_first_index <= end:
        new_list.append(the_list[list2_first_index])
        list2_first_index += 1
    for i in new_list:
        the_list[orig_start] = i
        orig_start += 1
    return the_list
```

```
Python 3.8.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ACER-DESKTOP\Documents\modul6\5.py ==========
[17, 34, 65]
>>>
```

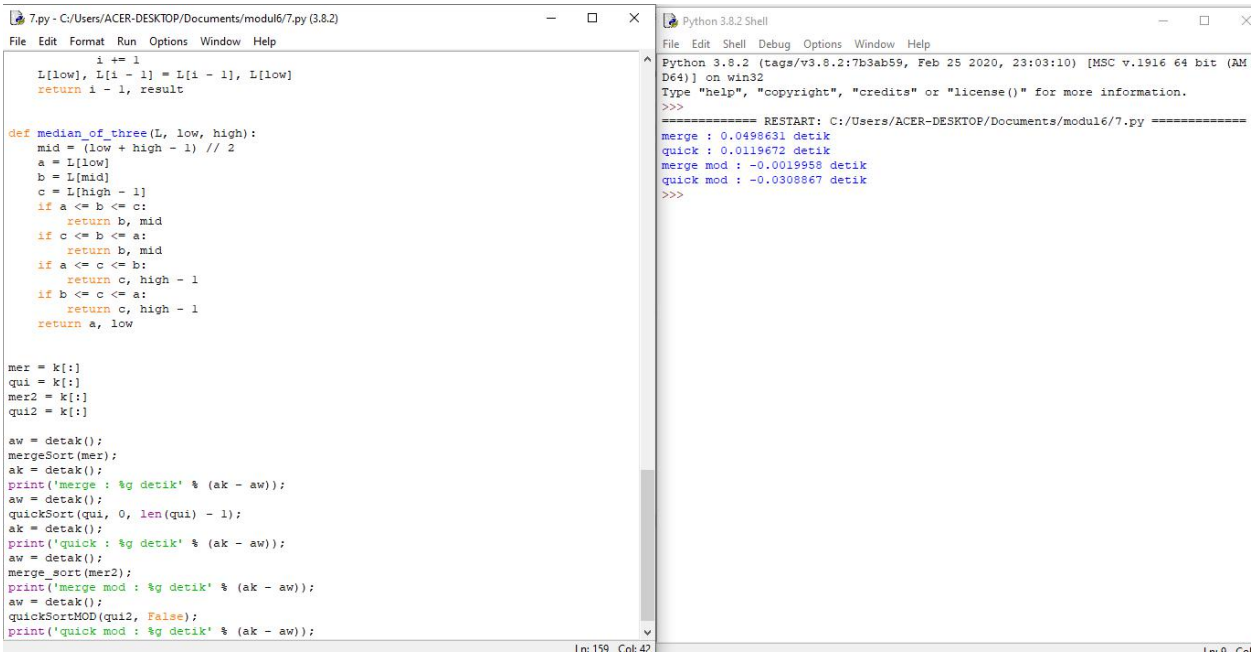
6.



```
6.py - C:\Users\ACER-DESKTOP\Documents\modul6\6.py (3.8.2)
File  Edit  Format  Run  Options  Window  Help
def quickSort(L, ascending=True):
    quicksorthelp(L, 0, len(L), ascending)


def quicksorthelp(L, low, high, ascending=True):
    result = 0
    if low < high:
        pivot_location, result = Partition(L, low, high, ascending)
        result += quicksorthelp(L, low, pivot_location, ascending)
        result += quicksorthelp(L, pivot_location + 1, high, ascending)
    return result


def Partition(L, low, high, ascending=True):
    result = 0
    pivot, pidx = median_of_three(L, low, high)
    L[low], L[pidx] = L[pidx], L[low]
    i = low + 1
    for j in range(low + 1, high, 1):
        result += 1
        if (ascending and L[j] < pivot) or (not ascending and L[j] > pivot):
            L[i], L[j] = L[j], L[i]
            i += 1
    L[low], L[i - 1] = L[i - 1], L[low]
    return i - 1, result


def median_of_three(L, low, high):
    mid = (low + high - 1) // 2
    a = L[low]
    b = L[mid]
    c = L[high - 1]
    if a <= b <= c:
        return b, mid
    if c <= b <= a:
        return b, mid
    if a <= c <= b:
        return c, high - 1
    if b <= c <= a:
        return c, high - 1
```

```
Python 3.8.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ACER-DESKTOP\Documents\modul6\6.py ==========
sorted:
[135, 124, 114, 112, 102]
>>>
```

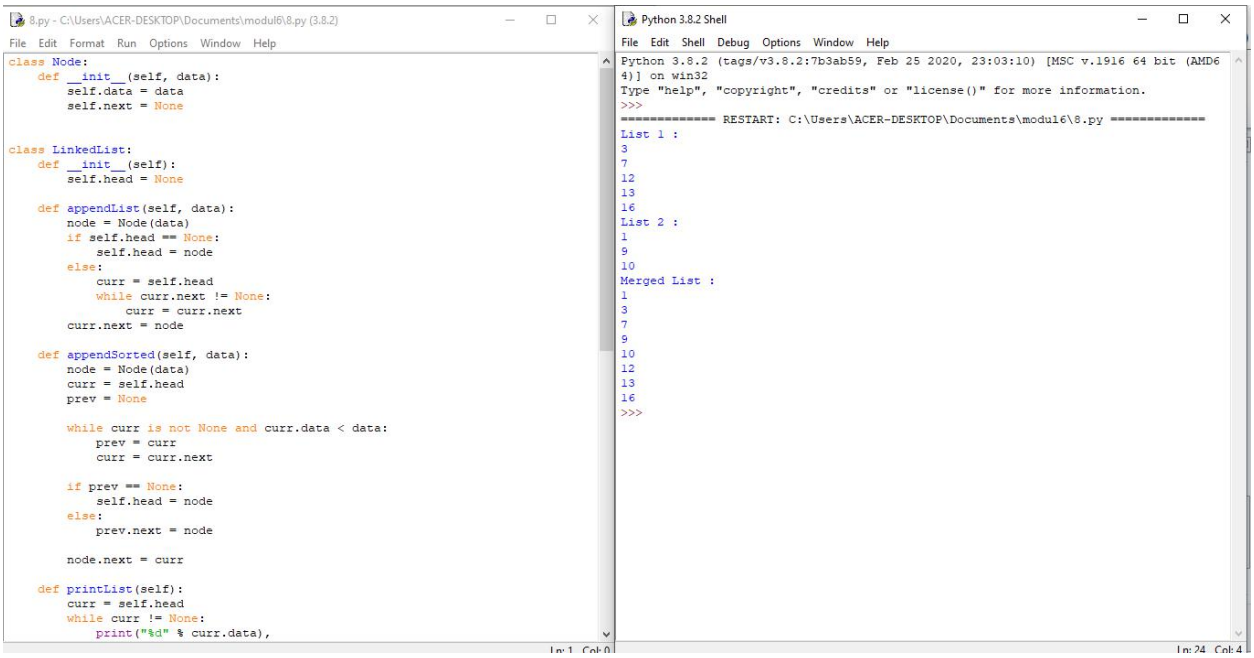
7.



```
            i += 1
    L[low], L[i - 1] = L[i - 1], L[low]
    return i - 1, result


def median_of_three(L, low, high):
    mid = (low + high - 1) // 2
    a = L[low]
    b = L[mid]
    c = L[high - 1]
    if a <= b <= c:
        return b, mid
    if c <= b <= a:
        return b, mid
    if a <= c <= b:
        return c, high - 1
    if b <= c <= a:
        return c, high - 1
    return a, low


mer = k[:]
qui = k[:]
mer2 = k[:]
qui2 = k[:]

aw = detak();
mergeSort(mer);
ak = detak();
print('merge : %g detik' % (ak - aw));
aw = detak();
quickSort(qui, 0, len(qui) - 1);
ak = detak();
print('quick : %g detik' % (ak - aw));
aw = detak();
merge_sort(mer2);
print('merge mod : %g detik' % (ak - aw));
aw = detak();
quickSortMOD(qui2, False);
print('quick mod : %g detik' % (ak - aw));
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/ACER-DESKTOP/Documents/modul6/7.py ==========
merge : 0.0498631 detik
quick : 0.0119672 detik
merge mod : -0.0019958 detik
quick mod : -0.0308867 detik
>>>
```

8.



```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None


class LinkedList:
    def __init__(self):
        self.head = None

    def appendList(self, data):
        node = Node(data)
        if self.head == None:
            self.head = node
        else:
            curr = self.head
            while curr.next != None:
                curr = curr.next
        curr.next = node

    def appendSorted(self, data):
        node = Node(data)
        curr = self.head
        prev = None

        while curr is not None and curr.data < data:
            prev = curr
            curr = curr.next

        if prev == None:
            self.head = node
        else:
            prev.next = node

        node.next = curr

    def printList(self):
        curr = self.head
        while curr != None:
            print("%d" % curr.data),
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ACER-DESKTOP\Documents\modul6\8.py ==========
List 1 :
3
7
12
13
16
List 2 :
1
9
10
Merged List :
1
3
7
9
10
12
13
16
>>>
```