Nama : Ismi Dzikrina

NIM : L200180010

Kelas : A

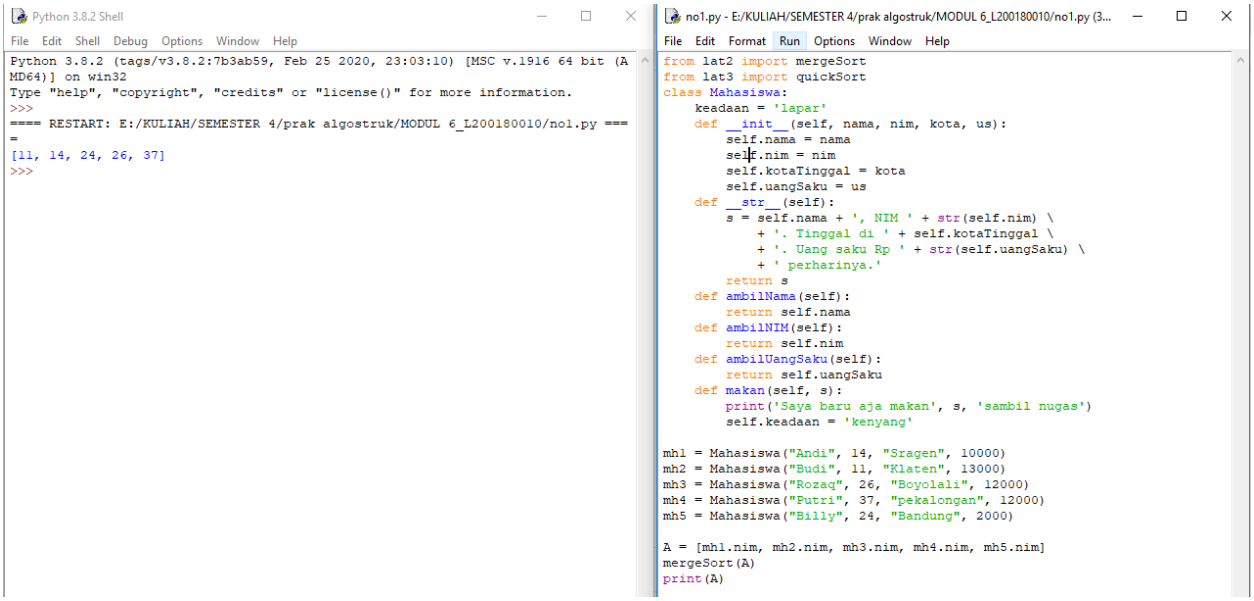
Matkul : Praktikum Algoritma dan Struktur Data

MODUL 6

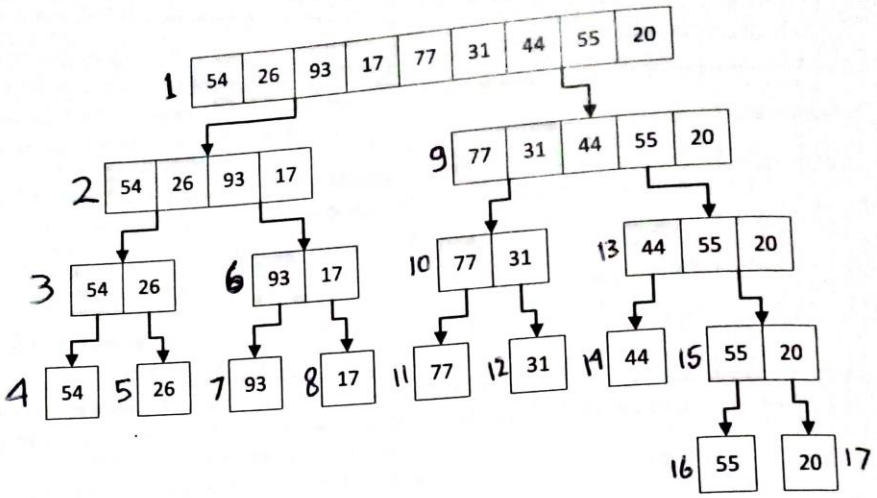
PENGURUTAN LANJUTAN

SOAL-SOAL UNTUK MAHASISWA

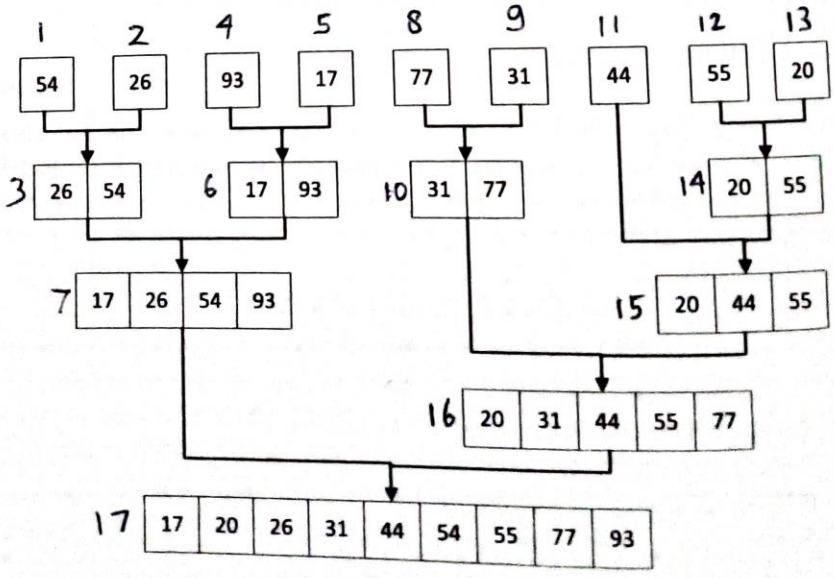
no1



No 2

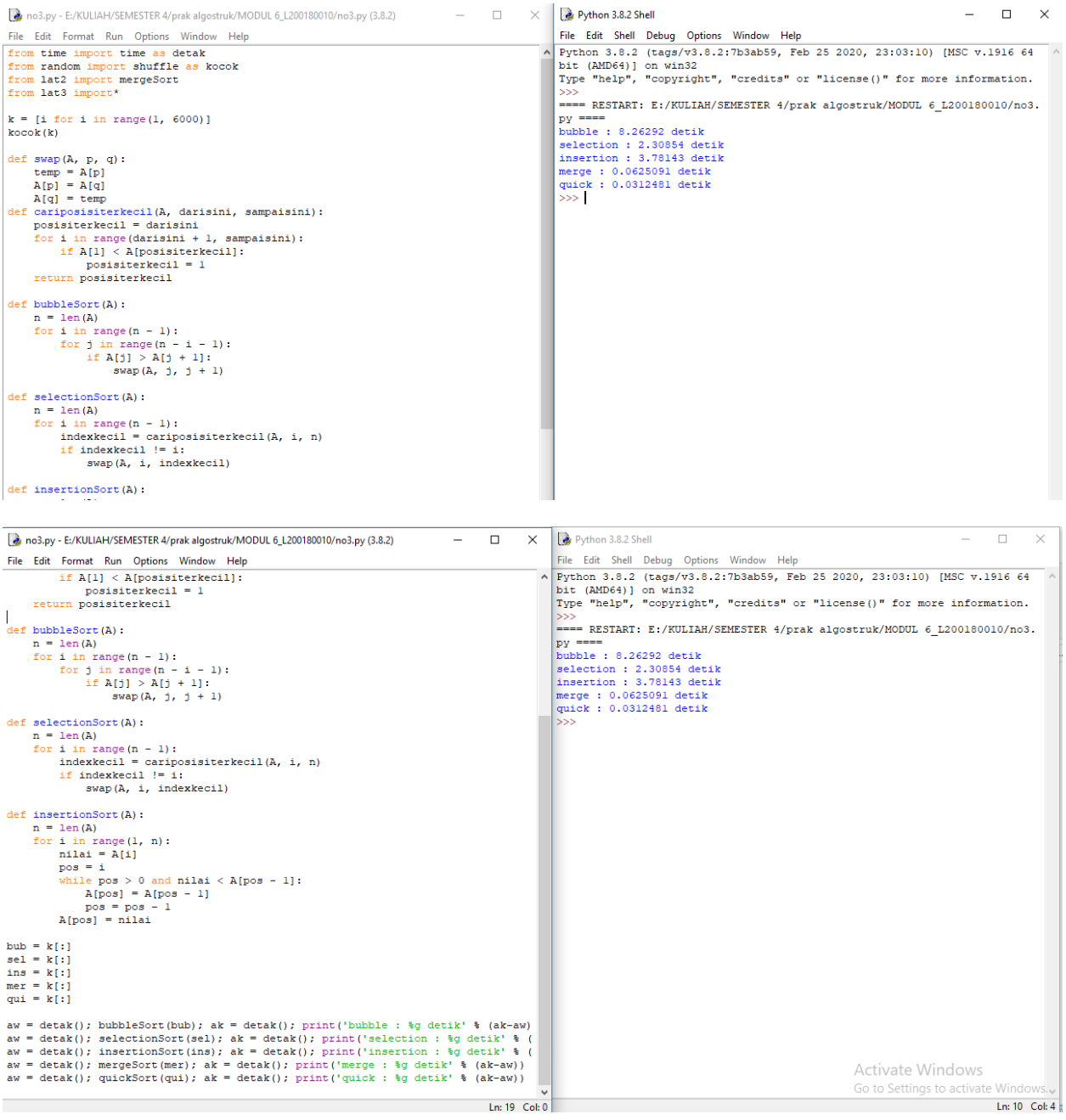


Gambar 6.1: Membelah list sampai tiap sub-list berisi satu elemen atau kosong. Sesudah itu digabung seperti ditunjukkan di Gambar 6.2.



Gambar 6.2: Menggabungkan list satu demi satu.

No3



```
from time import time as detak
from random import shuffle as kocok
from lat2 import mergeSort
from lat3 import*

k = [i for i in range(1, 6000)]
kocok(k)

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):
```

```
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: E:/KULIAH/SEMESTER 4/prak algostruk/MODUL 6_L200180010/no3.
py ====
bubble : 8.26292 detik
selection : 2.30854 detik
insertion : 3.78143 detik
merge : 0.0625091 detik
quick : 0.0312481 detik
>>>
```

```
        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

bub = k[:]
sel = k[:]
ins = k[:]
mer = k[:]
qui = k[:]

aw = detak(); bubbleSort(bub); ak = detak(); print('bubble : %g detik' % (ak-aw)
aw = detak(); selectionSort(sel); ak = detak(); print('selection : %g detik' % (
aw = detak(); insertionSort(ins); ak = detak(); print('insertion : %g detik' % (
aw = detak(); mergeSort(mer); ak = detak(); print('merge : %g detik' % (ak-aw))
aw = detak(); quickSort(qui); ak = detak(); print('quick : %g detik' % (ak-aw))
```

```
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: E:/KULIAH/SEMESTER 4/prak algostruk/MODUL 6_L200180010/no3.
py ====
bubble : 8.26292 detik
selection : 2.30854 detik
insertion : 3.78143 detik
merge : 0.0625091 detik
quick : 0.0312481 detik
>>>
```

No4

4a

L=[80,7,24,16,43,91,35,2,19,72]

80	7	24	16	43	91	35	2	19	72

Proses 1

7	80	26	24	43	91	2	35	19	72

Proses 2

7	16	24	80	2	35	43	91	19	72

Proses 3

2	7	16	24	35	43	80	91	19	72

Proses 4

2	7	16	19	24	35	43	72	80	91

4b

L=[80,7,24,16,43,91,35,2,19,72]

80	7	24	16	43	91	35	2	19	72

pivot

80	7	24	16	43	91	35	2	19	72
Low									High

pivot

72	7	24	16	43	91	35	2	19	80
Low									High

pivot

72	7	24	16	43	91	35	2	19	80
					Low				High

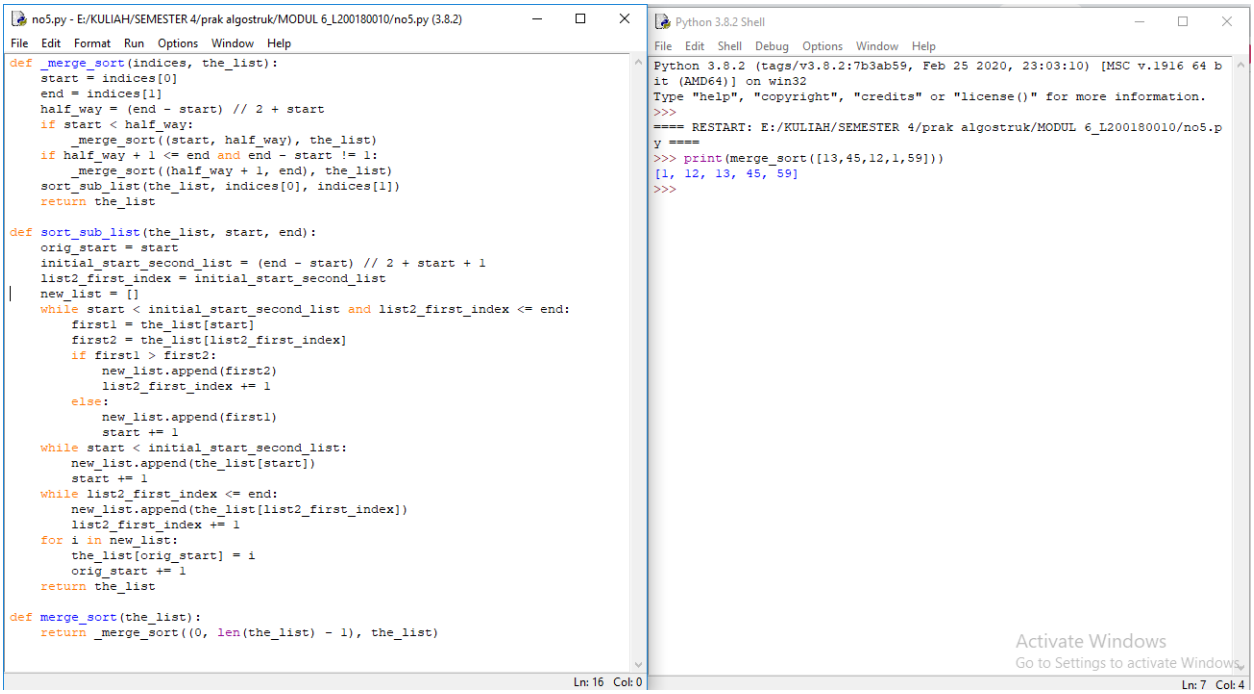
pivot

72	7	24	16	43	80	35	2	19	91
					Low				High

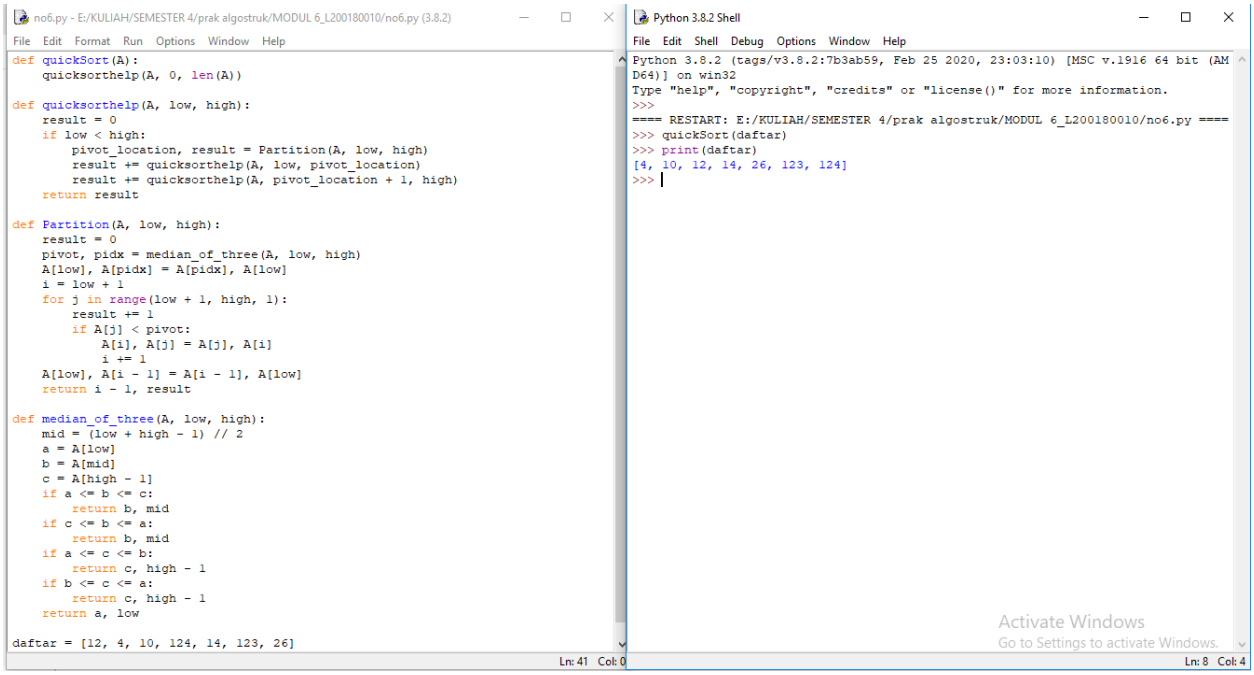
pivot

72	7	24	16	43	19	35	2	80	91
					Low			High	

No5



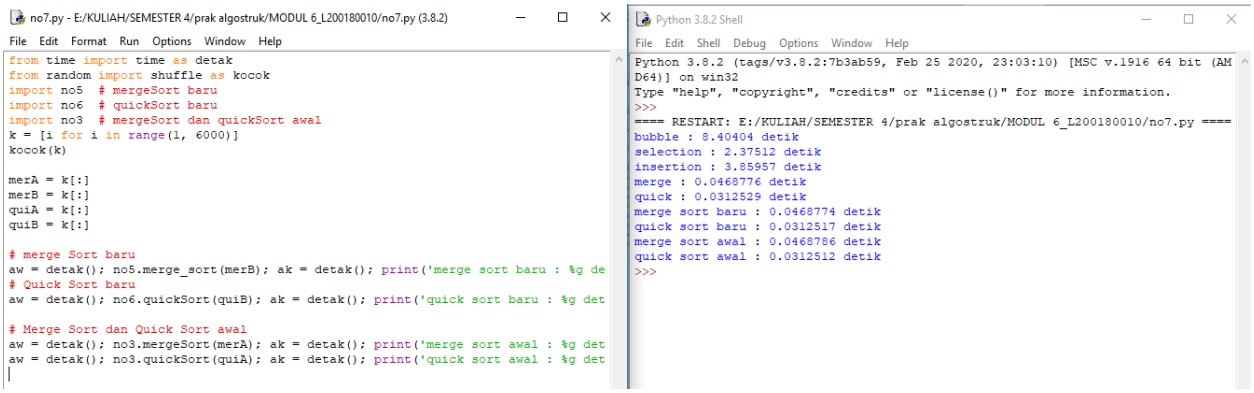
No6



```
def quickSort(A):
    quicksorthelp(A, 0, len(A))

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

def Partition(A, low, high):
    result = 0
    pivot, pidx = median_of_three(A, low, high)
    A[low], A[pidx] = A[pidx], A[low]
    i = low + 1
    for j in range(low + 1, high, 1):
        result += 1
        if A[j] < pivot:
            A[i], A[j] = A[j], A[i]
            i += 1
    A[low], A[i - 1] = A[i - 1], A[low]
    return i - 1, result

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

daftar = [12, 4, 10, 124, 14, 123, 26]
```

```
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: E:/KULIAH/SEMESTER 4/prak algostruk/MODUL 6_L200180010/no6.py ====
>>> quickSort(daftar)
>>> print(daftar)
[4, 10, 12, 14, 26, 123, 124]
>>>
```

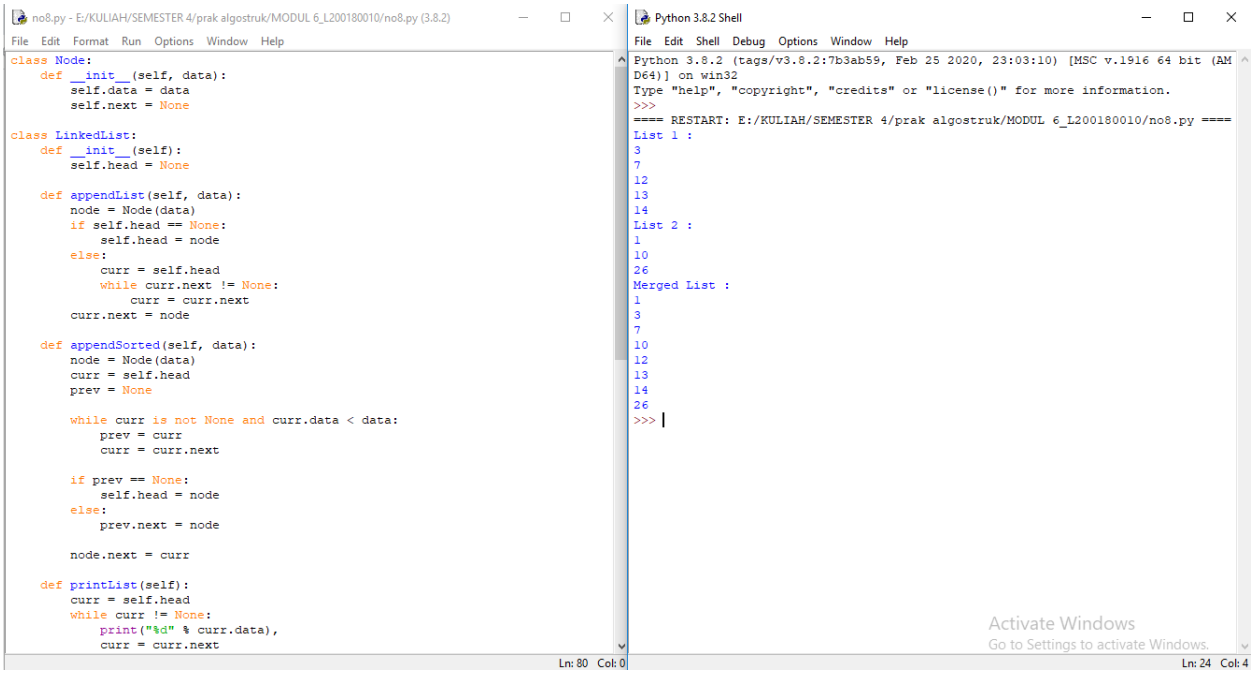
No7



```
from time import time as detak
from random import shuffle as kocok
import no5  # mergeSort baru
import no6  # quickSort baru
import no3  # mergeSort dan quickSort awal
k = [i for i in range(1, 6000)]
kocok(k)

merA = k[:]
merB = k[:]
quiA = k[:]
quiB = k[:]

# merge Sort baru
aw = detak(); no5.merge_sort(merB); ak = detak(); print('merge sort baru : %g de
# Quick Sort baru
aw = detak(); no6.quickSort(quiB); ak = detak(); print('quick sort baru : %g det

# Merge Sort dan Quick Sort awal
aw = detak(); no3.mergeSort(merA); ak = detak(); print('merge sort awal : %g det
aw = detak(); no3.quickSort(quiA); ak = detak(); print('quick sort awal : %g det
```

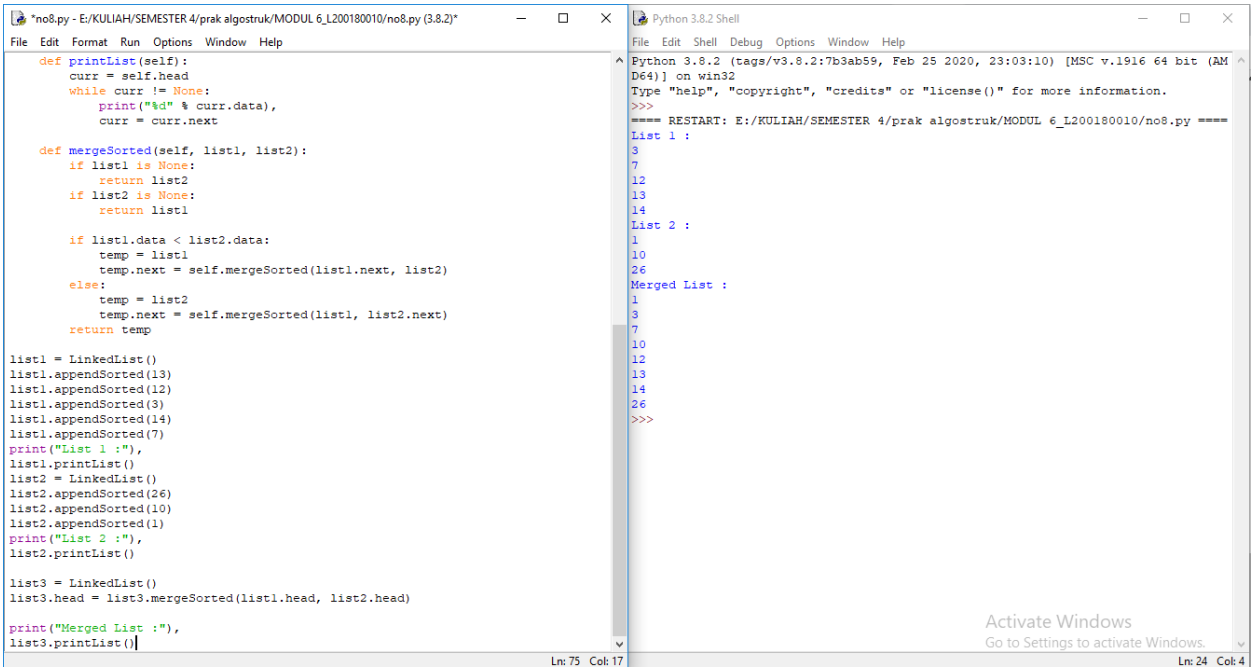
```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:/KULIAH/SEMESTER 4/prak algostruk/MODUL 6_L200180010/no7.py ====
bubble : 8.40404 detik
selection : 2.37512 detik
insertion : 3.85957 detik
merge : 0.0468776 detik
quick : 0.0312529 detik
merge sort baru : 0.0468774 detik
quick sort baru : 0.0312517 detik
merge sort awal : 0.0468786 detik
quick sort awal : 0.0312512 detik
>>>
```

No8



```
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),
            curr = curr.next
```

```
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: E:/KULIAH/SEMESTER 4/prak algostruk/MODUL 6_L200180010/no8.py ====
List 1 :
3
7
12
13
14
List 2 :
1
10
26
Merged List :
1
3
7
10
12
13
14
26
>>>
```



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

    def mergeSorted(self, list1, list2):
        if list1 is None:
            return list2
        if list2 is None:
            return list1

        if list1.data < list2.data:
            temp = list1
            temp.next = self.mergeSorted(list1.next, list2)
        else:
            temp = list2
            temp.next = self.mergeSorted(list1, list2.next)
        return temp

list1 = LinkedList()
list1.appendSorted(13)
list1.appendSorted(12)
list1.appendSorted(3)
list1.appendSorted(14)
list1.appendSorted(7)
print("List 1 :"),
list1.printList()
list2 = LinkedList()
list2.appendSorted(26)
list2.appendSorted(10)
list2.appendSorted(1)
print("List 2 :"),
list2.printList()

list3 = LinkedList()
list3.head = list3.mergeSorted(list1.head, list2.head)

print("Merged List :"),
list3.printList()
```

```
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: E:/KULIAH/SEMESTER 4/prak algostruk/MODUL 6_L200180010/no8.py ====
List 1 :
3
7
12
13
14
List 2 :
1
10
26
Merged List :
1
3
7
10
12
13
14
26
>>>
```