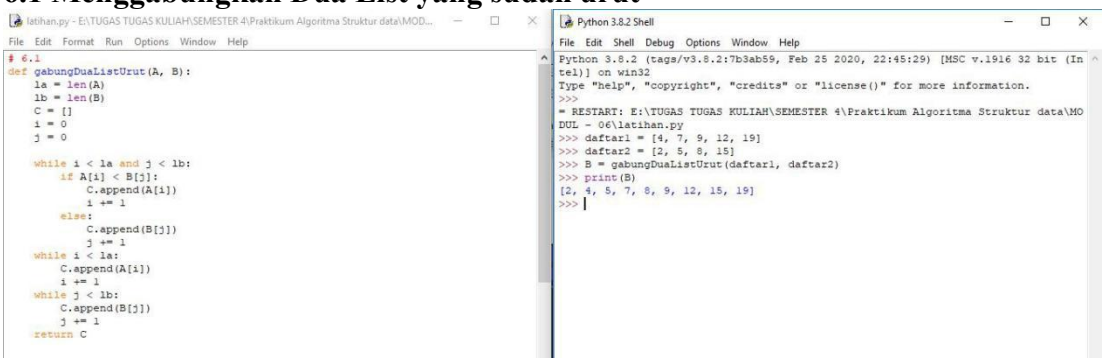
Nama : Rifqi Alwan P
NIM : L200180008
Kelas A

MODUL 6

Latihan

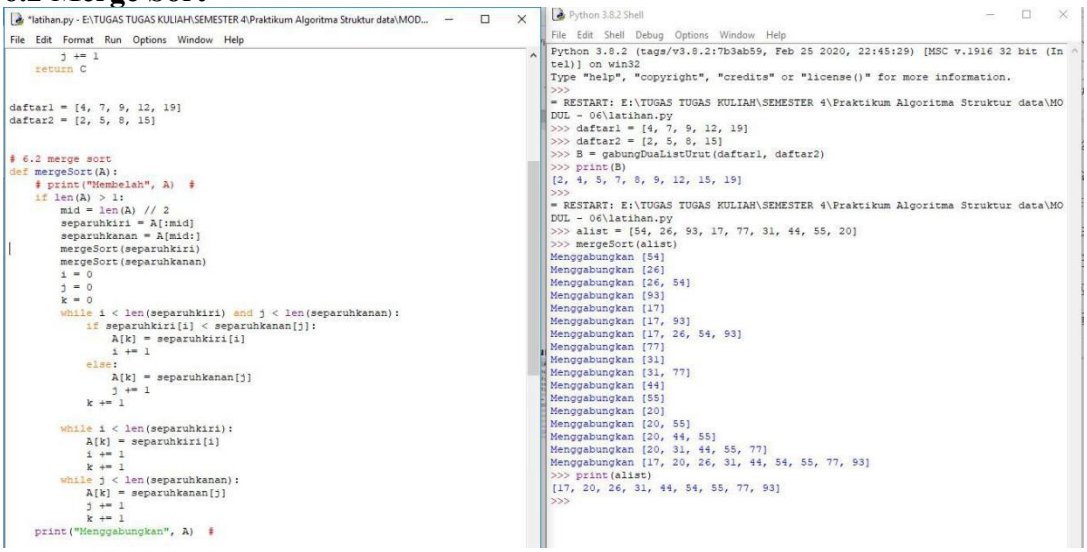
6.1 Menggabungkan Dua List yang sudah urut



```
# 6.1
def gabungDuaListUrut(A, B):
    la = len(A)
    lb = len(B)
    C = []
    i = 0
    j = 0

    while i < la and j < lb:
        if A[i] < B[j]:
            C.append(A[i])
            i += 1
        else:
            C.append(B[j])
            j += 1
    while i < la:
        C.append(A[i])
        i += 1
    while j < lb:
        C.append(B[j])
        j += 1
    return C
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL - 06\latihan.py
>>> daftar1 = [4, 7, 9, 12, 19]
>>> daftar2 = [2, 5, 8, 15]
>>> B = gabungDuaListUrut(daftar1, daftar2)
>>> print(B)
[2, 4, 5, 7, 8, 9, 12, 15, 19]
>>>
```

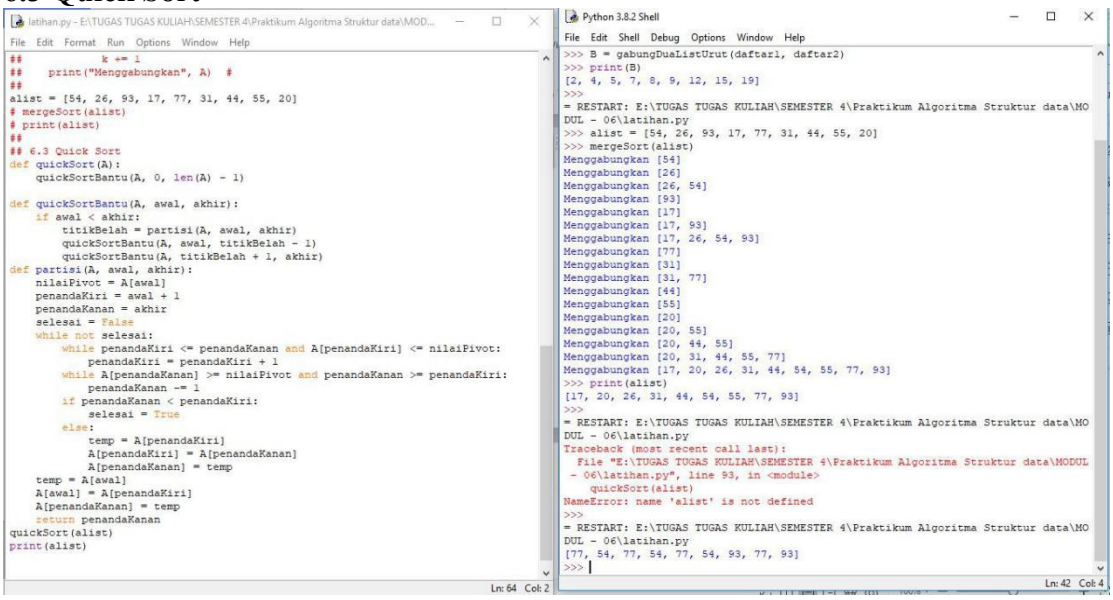
6.2 Merge Sort



```
        j += 1
    return C


daftar1 = [4, 7, 9, 12, 19]
daftar2 = [2, 5, 8, 15]


# 6.2 merge sort
def mergeSort(A):
    # print("Membelah", A)  #
    if len(A) > 1:
        mid = len(A) // 2
        separuhkiri = A[:mid]
        separuhkanan = A[mid:]
        mergeSort(separuhkiri)
        mergeSort(separuhkanan)
        i = 0
        j = 0
        k = 0
        while i < len(separuhkiri) and j < len(separuhkanan):
            if separuhkiri[i] < separuhkanan[j]:
                A[k] = separuhkiri[i]
                i += 1
            else:
                A[k] = separuhkanan[j]
                j += 1
            k += 1

        while i < len(separuhkiri):
            A[k] = separuhkiri[i]
            i += 1
            k += 1
        while j < len(separuhkanan):
            A[k] = separuhkanan[j]
            j += 1
            k += 1
    print("Menggabungkan", A)  #
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL - 06\latihan.py
>>> daftar1 = [4, 7, 9, 12, 19]
>>> daftar2 = [2, 5, 8, 15]
>>> B = gabungDuaListUrut(daftar1, daftar2)
>>> print(B)
[2, 4, 5, 7, 8, 9, 12, 15, 19]
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL - 06\latihan.py
>>> alist = [54, 26, 93, 17, 77, 31, 44, 55, 20]
>>> mergeSort(alist)
Menggabungkan [54]
Menggabungkan [26]
Menggabungkan [26, 54]
Menggabungkan [93]
Menggabungkan [17]
Menggabungkan [17, 93]
Menggabungkan [17, 26, 54, 93]
Menggabungkan [77]
Menggabungkan [31]
Menggabungkan [31, 77]
Menggabungkan [44]
Menggabungkan [55]
Menggabungkan [20]
Menggabungkan [20, 55]
Menggabungkan [20, 44, 55]
Menggabungkan [20, 31, 44, 55, 77]
Menggabungkan [17, 20, 26, 31, 44, 54, 55, 77, 93]
>>> print(alist)
[17, 20, 26, 31, 44, 54, 55, 77, 93]
>>>
```

6.3 Quick Sort

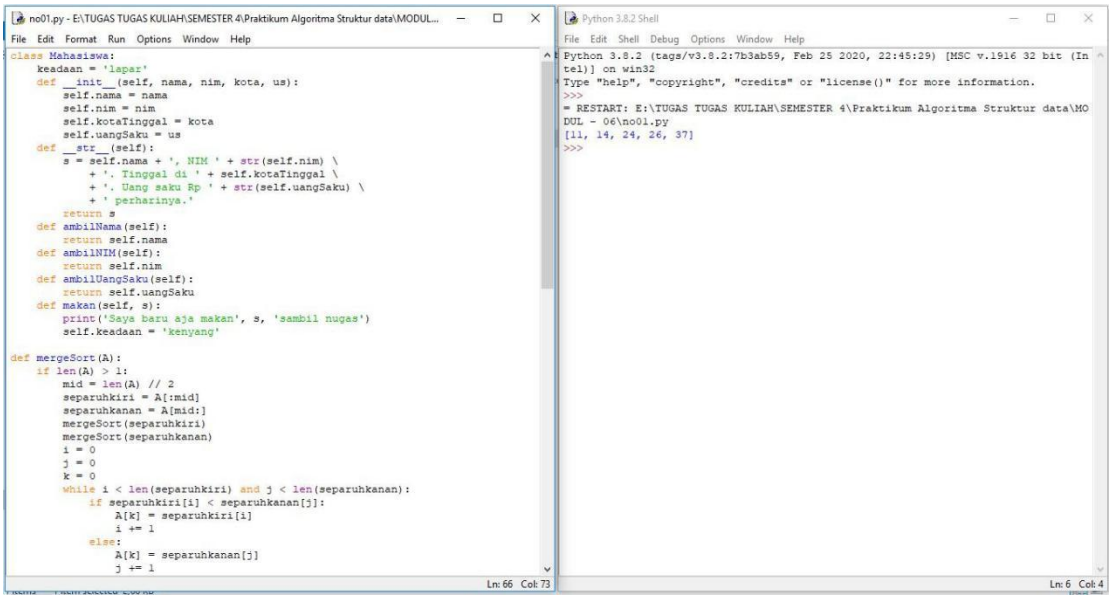


```
##            k += 1
##    print("Menggabungkan", A)  #
##
alist = [54, 26, 93, 17, 77, 31, 44, 55, 20]
# mergeSort(alist)
# print(alist)
##
## 6.3 Quick Sort
def quickSort(A):
    quickSortBantu(A, 0, len(A) - 1)

def quickSortBantu(A, awal, akhir):
    if awal < akhir:
        titikBelah = partisi(A, awal, akhir)
        quickSortBantu(A, awal, titikBelah - 1)
        quickSortBantu(A, titikBelah + 1, akhir)
def partisi(A, awal, akhir):
    nilaiPivot = A[awal]
    penandaKiri = awal + 1
    penandaKanan = akhir
    selesai = False
    while not selesai:
        while penandaKiri <= penandaKanan and A[penandaKiri] <= nilaiPivot:
            penandaKiri = penandaKiri + 1
        while A[penandaKanan] >= nilaiPivot and penandaKanan >= penandaKiri:
            penandaKanan -= 1
        if penandaKanan < penandaKiri:
            selesai = True
        else:
            temp = A[penandaKiri]
            A[penandaKiri] = A[penandaKanan]
            A[penandaKanan] = temp
    temp = A[awal]
    A[awal] = A[penandaKiri]
    A[penandaKanan] = temp
    return penandaKanan
quickSort(alist)
print(alist)
```

```
>>> B = gabungDuaListUrut(daftar1, daftar2)
>>> print(B)
[2, 4, 5, 7, 8, 9, 12, 15, 19]
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\latihan.py
>>> alist = [54, 26, 93, 17, 77, 31, 44, 55, 20]
>>> mergeSort(alist)
Menggabungkan [54]
Menggabungkan [26]
Menggabungkan [26, 54]
Menggabungkan [93]
Menggabungkan [17]
Menggabungkan [17, 93]
Menggabungkan [17, 26, 54, 93]
Menggabungkan [77]
Menggabungkan [31]
Menggabungkan [31, 77]
Menggabungkan [44]
Menggabungkan [55]
Menggabungkan [20]
Menggabungkan [20, 55]
Menggabungkan [20, 44, 55]
Menggabungkan [20, 31, 44, 55, 77]
Menggabungkan [17, 20, 26, 31, 44, 54, 55, 77, 93]
>>> print(alist)
[17, 20, 26, 31, 44, 54, 55, 77, 93]
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\latihan.py
Traceback (most recent call last):
  File "E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL
 - 06\latihan.py", line 93, in <module>
    quickSort(alist)
NameError: name 'alist' is not defined
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\latihan.py
[77, 54, 77, 54, 77, 54, 93, 77, 93]
>>>
```

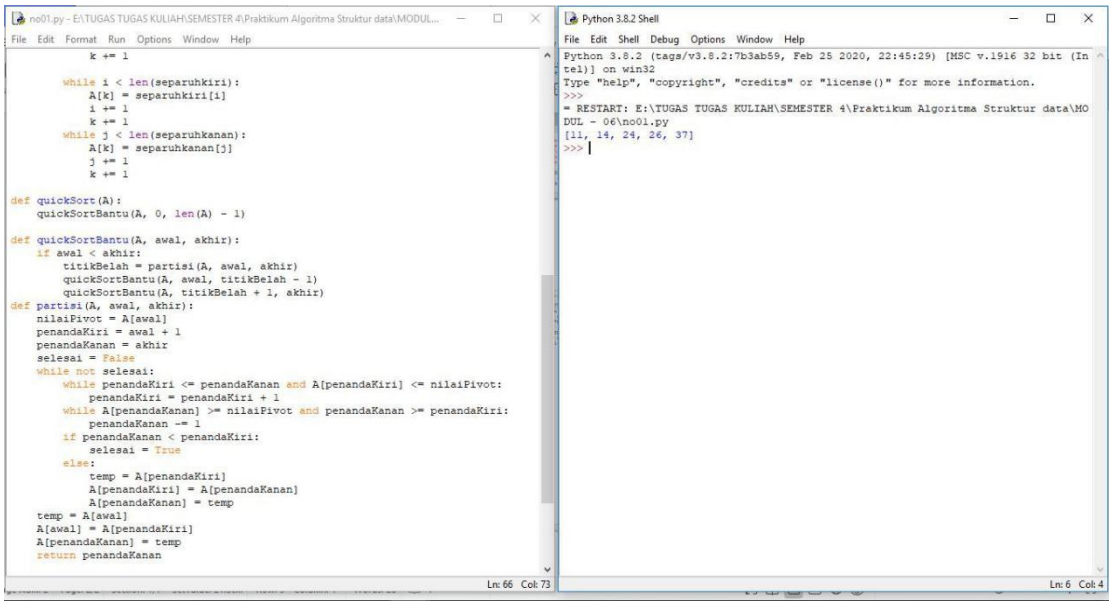
Soal-soal

1.



```
class Mahasiswa:
    keadaan = 'lapar'
    def __init__(self, nama, nim, kota, us):
        self.nama = nama
        self.nim = nim
        self.kotaTinggal = kota
        self.uangSaku = us
    def __str__(self):
        s = self.nama + ', NIM ' + str(self.nim) \
            + '. Tinggal di ' + self.kotaTinggal \
            + '. Uang saku Rp ' + str(self.uangSaku) \
            + ' perharinya.'
        return s
    def ambilNama(self):
        return self.nama
    def ambilNIM(self):
        return self.nim
    def ambilUangSaku(self):
        return self.uangSaku
    def makan(self, s):
        print('Saya baru aja makan', s, 'sambil nugas')
        self.keadaan = 'kenyang'

def mergeSort(A):
    if len(A) > 1:
        mid = len(A) // 2
        separuhkiri = A[:mid]
        separuhkanan = A[mid:]
        mergeSort(separuhkiri)
        mergeSort(separuhkanan)
        i = 0
        j = 0
        k = 0
        while i < len(separuhkiri) and j < len(separuhkanan):
            if separuhkiri[i] < separuhkanan[j]:
                A[k] = separuhkiri[i]
                i += 1
            else:
                A[k] = separuhkanan[j]
                j += 1
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no01.py
[11, 14, 24, 26, 37]
>>>
```

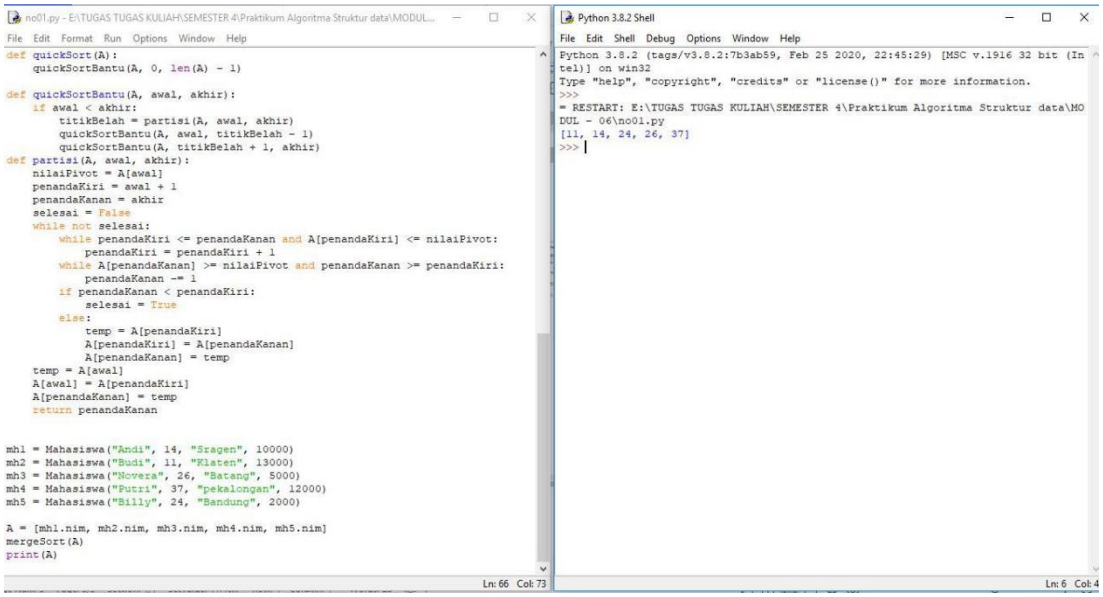


```
no01.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL...
File  Edit  Format  Run  Options  Window  Help
            k += 1

        while i < len(separuhkiri):
            A[k] = separuhkiri[i]
            i += 1
            k += 1
        while j < len(separuhkanan):
            A[k] = separuhkanan[j]
            j += 1
            k += 1

def quickSort(A):
    quickSortBantu(A, 0, len(A) - 1)

def quickSortBantu(A, awal, akhir):
    if awal < akhir:
        titikBelah = partisi(A, awal, akhir)
        quickSortBantu(A, awal, titikBelah - 1)
        quickSortBantu(A, titikBelah + 1, akhir)
def partisi(A, awal, akhir):
    nilaiPivot = A[awal]
    penandaKiri = awal + 1
    penandaKanan = akhir
    selesai = False
    while not selesai:
        while penandaKiri <= penandaKanan and A[penandaKiri] <= nilaiPivot:
            penandaKiri = penandaKiri + 1
        while A[penandaKanan] >= nilaiPivot and penandaKanan >= penandaKiri:
            penandaKanan -= 1
        if penandaKanan < penandaKiri:
            selesai = True
        else:
            temp = A[penandaKiri]
            A[penandaKiri] = A[penandaKanan]
            A[penandaKanan] = temp
    temp = A[awal]
    A[awal] = A[penandaKiri]
    A[penandaKanan] = temp
    return penandaKanan
```

Ln: 66 Col: 73

```
Python 3.8.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no01.py
[11, 14, 24, 26, 37]
>>>
```

Ln: 6 Col: 4



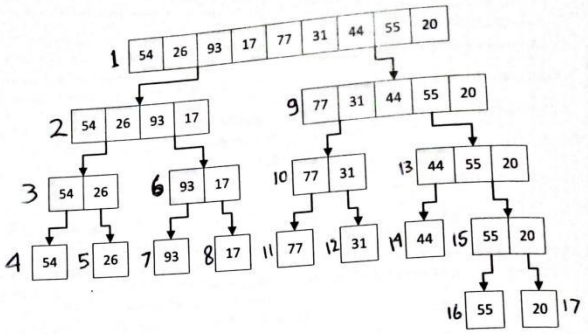
```
no01.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL...
File  Edit  Format  Run  Options  Window  Help
def quickSort(A):
    quickSortBantu(A, 0, len(A) - 1)

def quickSortBantu(A, awal, akhir):
    if awal < akhir:
        titikBelah = partisi(A, awal, akhir)
        quickSortBantu(A, awal, titikBelah - 1)
        quickSortBantu(A, titikBelah + 1, akhir)
def partisi(A, awal, akhir):
    nilaiPivot = A[awal]
    penandaKiri = awal + 1
    penandaKanan = akhir
    selesai = False
    while not selesai:
        while penandaKiri <= penandaKanan and A[penandaKiri] <= nilaiPivot:
            penandaKiri = penandaKiri + 1
        while A[penandaKanan] >= nilaiPivot and penandaKanan >= penandaKiri:
            penandaKanan -= 1
        if penandaKanan < penandaKiri:
            selesai = True
        else:
            temp = A[penandaKiri]
            A[penandaKiri] = A[penandaKanan]
            A[penandaKanan] = temp
    temp = A[awal]
    A[awal] = A[penandaKiri]
    A[penandaKanan] = temp
    return penandaKanan


mh1 = Mahasiswa("Andi", 14, "Sragen", 10000)
mh2 = Mahasiswa("Budi", 11, "Klaten", 13000)
mh3 = Mahasiswa("Novera", 26, "Batang", 5000)
mh4 = Mahasiswa("Putri", 37, "pekalongan", 12000)
mh5 = Mahasiswa("Billy", 24, "Bandung", 2000)

A = [mh1.nim, mh2.nim, mh3.nim, mh4.nim, mh5.nim]
mergeSort(A)
print(A)
```

Ln: 66 Col: 73

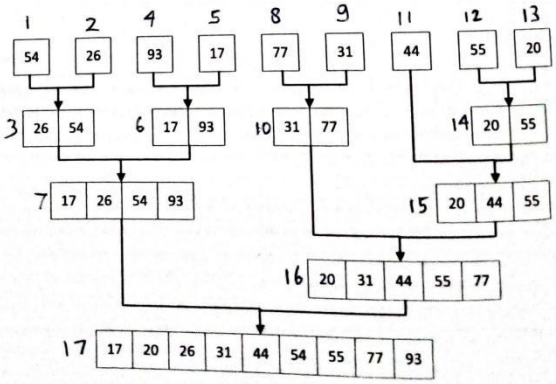
```
Python 3.8.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no01.py
[11, 14, 24, 26, 37]
>>>
```

Ln: 6 Col: 4

2.

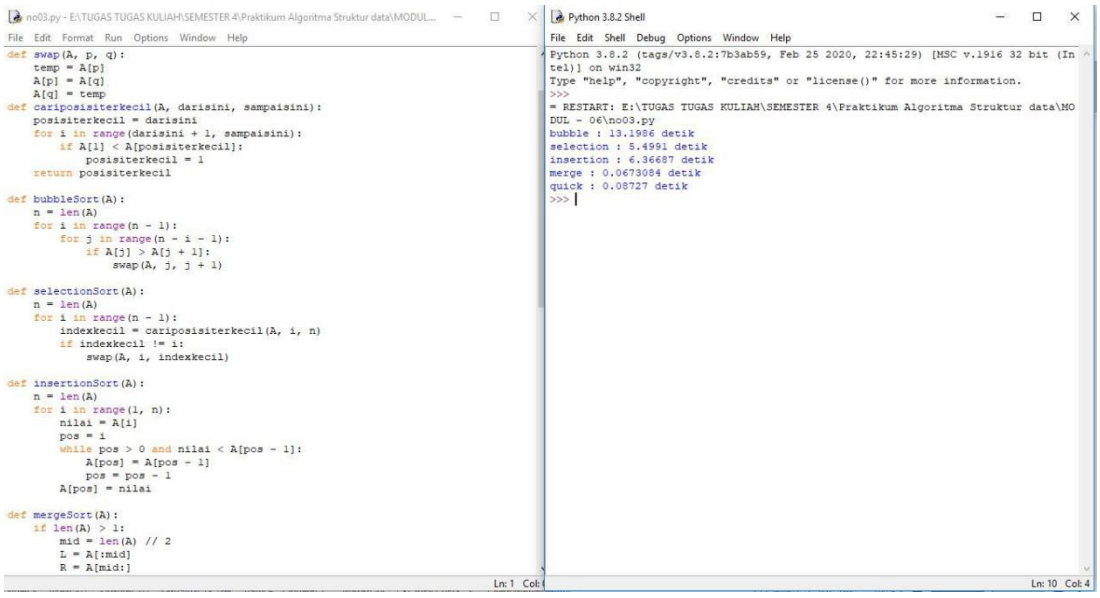


Gambar 6.1: Membelah list sampai tiap sub-list berisi satu elemen atau kosong. Sesudah itu digabung seperti ditunjukkan di Gambar 6.2.



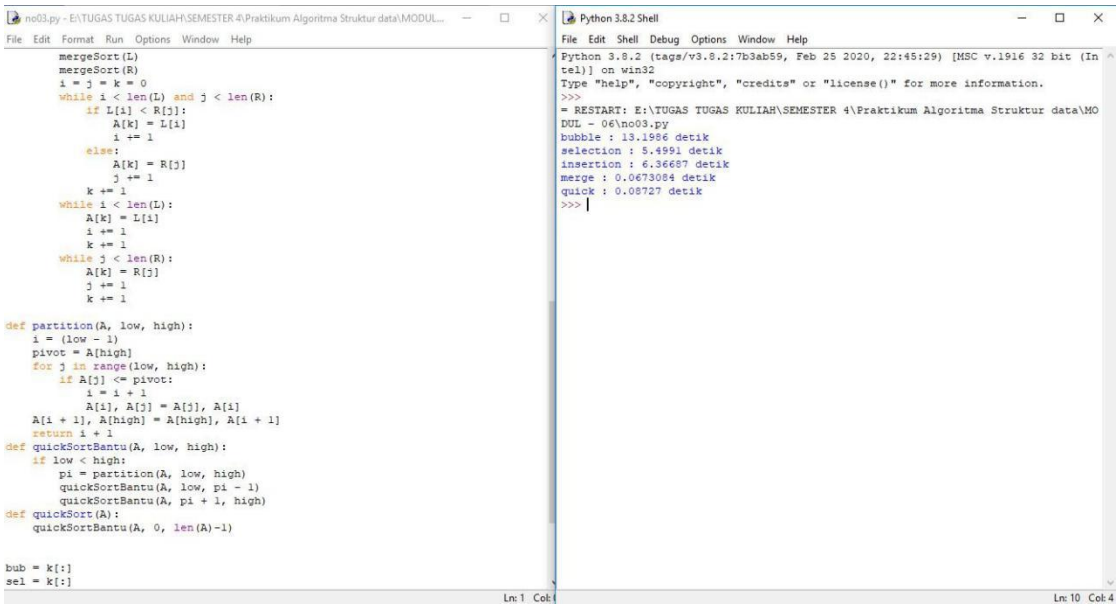
Gambar 6.2: Menggabungkan list satu demi satu.

3.



```
def swap(A, p, q):
    temp = A[p]
    A[p] = A[q]
    A[q] = temp
def caripositerkecil(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 = caripositerkecil(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:
        mid = len(A) // 2
        L = A[:mid]
        R = A[mid:]
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL - 06\no03.py
bubble : 13.1986 detik
selection : 5.4991 detik
insertion : 6.36687 detik
merge : 0.0673084 detik
quick : 0.08727 detik
>>>
```

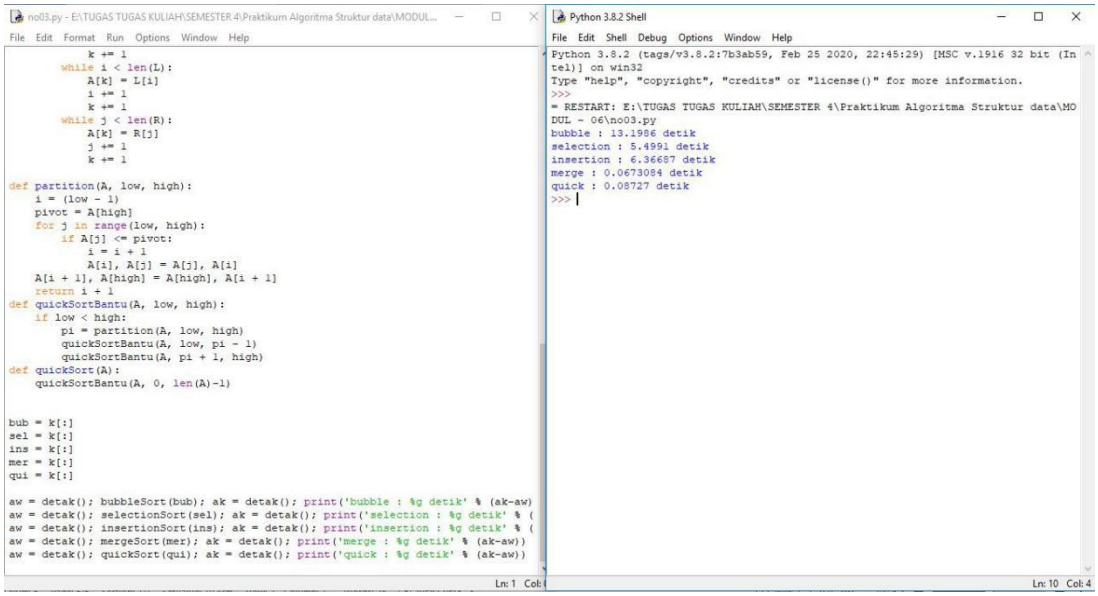


```
no03.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL...
File  Edit  Format  Run  Options  Window  Help
        mergeSort(L)
        mergeSort(R)
        i = j = k = 0
        while i < len(L) and j < len(R):
            if L[i] < R[j]:
                A[k] = L[i]
                i += 1
            else:
                A[k] = R[j]
                j += 1
            k += 1
        while i < len(L):
            A[k] = L[i]
            i += 1
            k += 1
        while j < len(R):
            A[k] = R[j]
            j += 1
            k += 1

def partition(A, low, high):
    i = (low - 1)
    pivot = A[high]
    for j in range(low, high):
        if A[j] <= pivot:
            i = i + 1
            A[i], A[j] = A[j], A[i]
    A[i + 1], A[high] = A[high], A[i + 1]
    return i + 1
def quickSortBantu(A, low, high):
    if low < high:
        pi = partition(A, low, high)
        quickSortBantu(A, low, pi - 1)
        quickSortBantu(A, pi + 1, high)
def quickSort(A):
    quickSortBantu(A, 0, len(A)-1)


bub = k[:]
sel = k[:]
```

Ln: 1 Col:

```
Python 3.8.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no03.py
bubble : 13.1986 detik
selection : 5.4991 detik
insertion : 6.36687 detik
merge : 0.0673084 detik
quick : 0.08727 detik
>>>
```

Ln: 10 Col: 4



```
no03.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL...
File  Edit  Format  Run  Options  Window  Help
            k += 1
        while i < len(L):
            A[k] = L[i]
            i += 1
            k += 1
        while j < len(R):
            A[k] = R[j]
            j += 1
            k += 1

def partition(A, low, high):
    i = (low - 1)
    pivot = A[high]
    for j in range(low, high):
        if A[j] <= pivot:
            i = i + 1
            A[i], A[j] = A[j], A[i]
    A[i + 1], A[high] = A[high], A[i + 1]
    return i + 1
def quickSortBantu(A, low, high):
    if low < high:
        pi = partition(A, low, high)
        quickSortBantu(A, low, pi - 1)
        quickSortBantu(A, pi + 1, high)
def quickSort(A):
    quickSortBantu(A, 0, len(A)-1)


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))
```

Ln: 1 Col:

```
Python 3.8.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no03.py
bubble : 13.1986 detik
selection : 5.4991 detik
insertion : 6.36687 detik
merge : 0.0673084 detik
quick : 0.08727 detik
>>>
```

Ln: 10 Col: 4

4A. Merge sort

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. Quick Sort

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

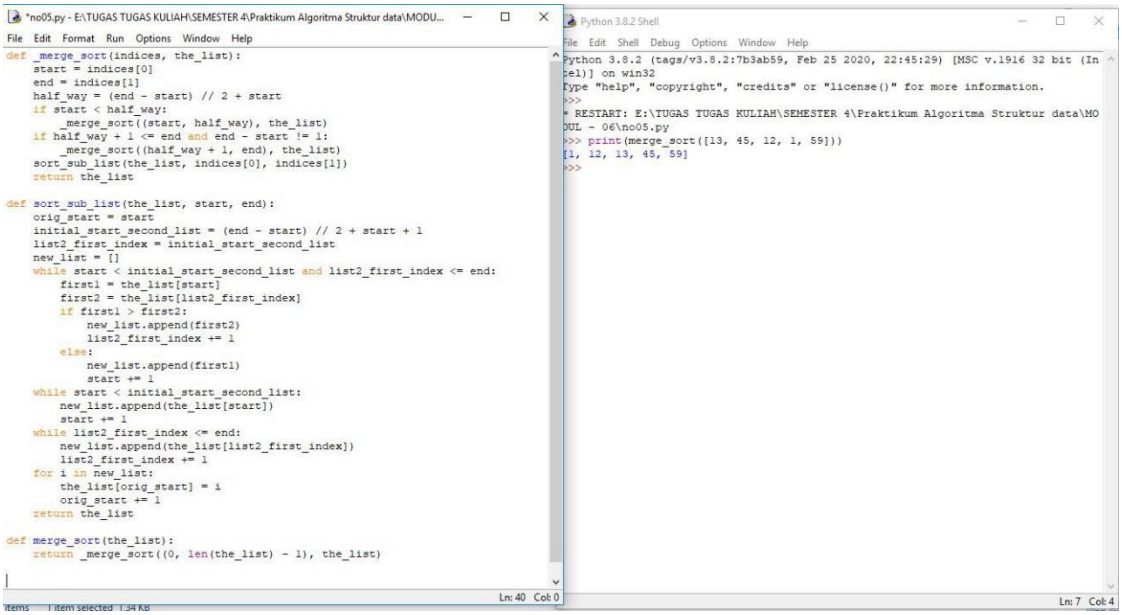
Low High

pivot

72	7	24	16	43	19	35	2	80	91

Low High

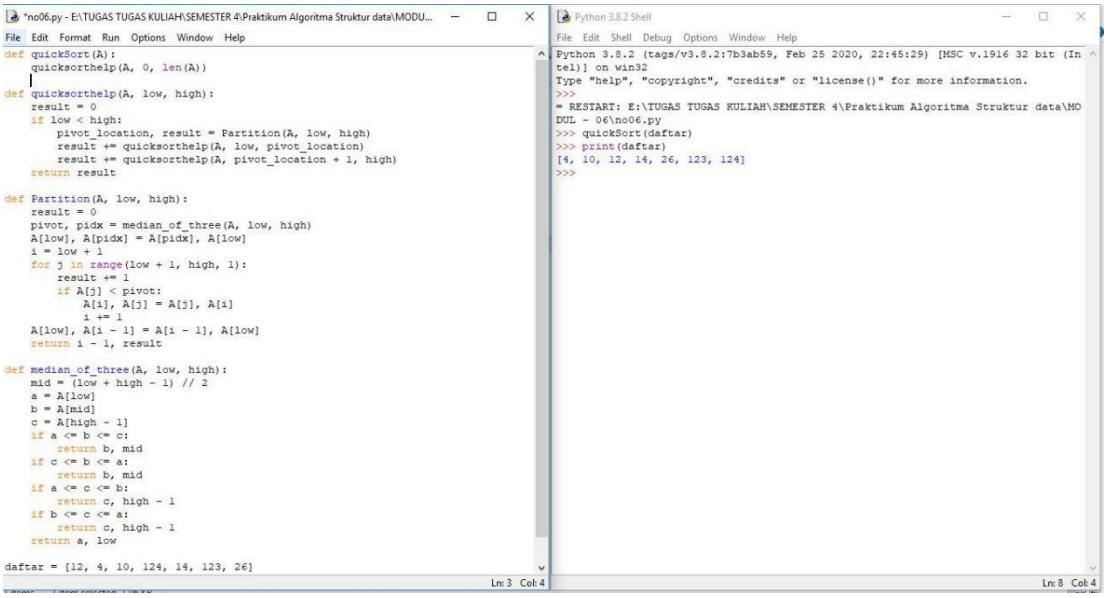
5.



```
*no05.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODU...
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

def merge_sort(the_list):
    return _merge_sort((0, len(the_list) - 1), the_list)

Ln: 40  Col: 0
```

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no05.py
>>> print(merge_sort([13, 45, 12, 1, 59]))
[1, 12, 13, 45, 59]
>>>

Ln: 7  Col: 4
```

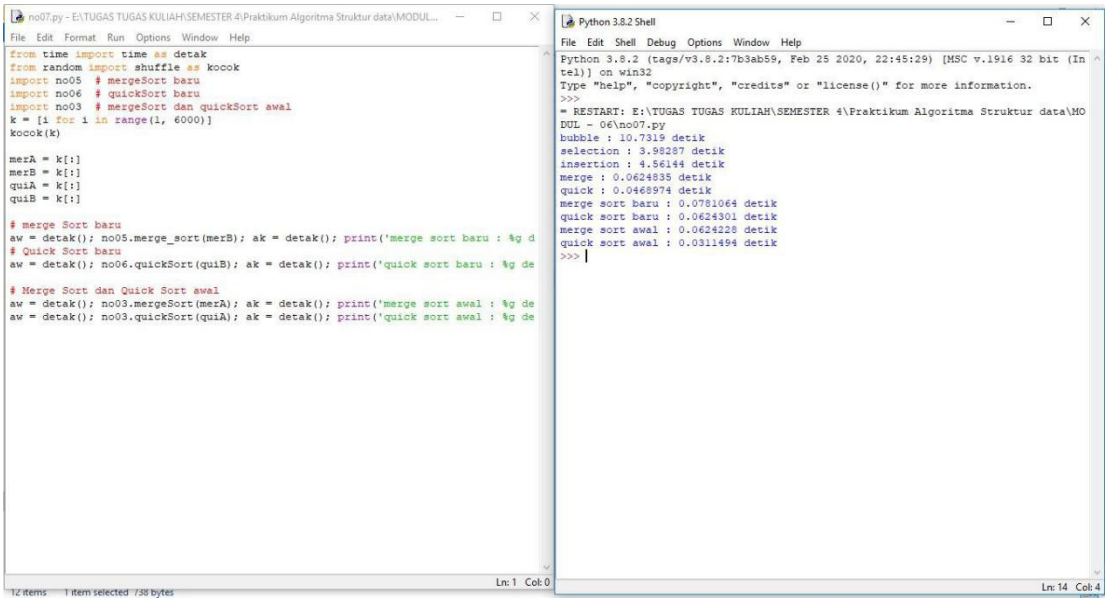
6.



```
*no06.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODU...
File Edit Format Run Options Window Help
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]

Ln: 3  Col: 4
```

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no06.py
>>> quickSort(daftar)
>>> print(daftar)
[4, 10, 12, 14, 26, 123, 124]
>>>

Ln: 8  Col: 4
```

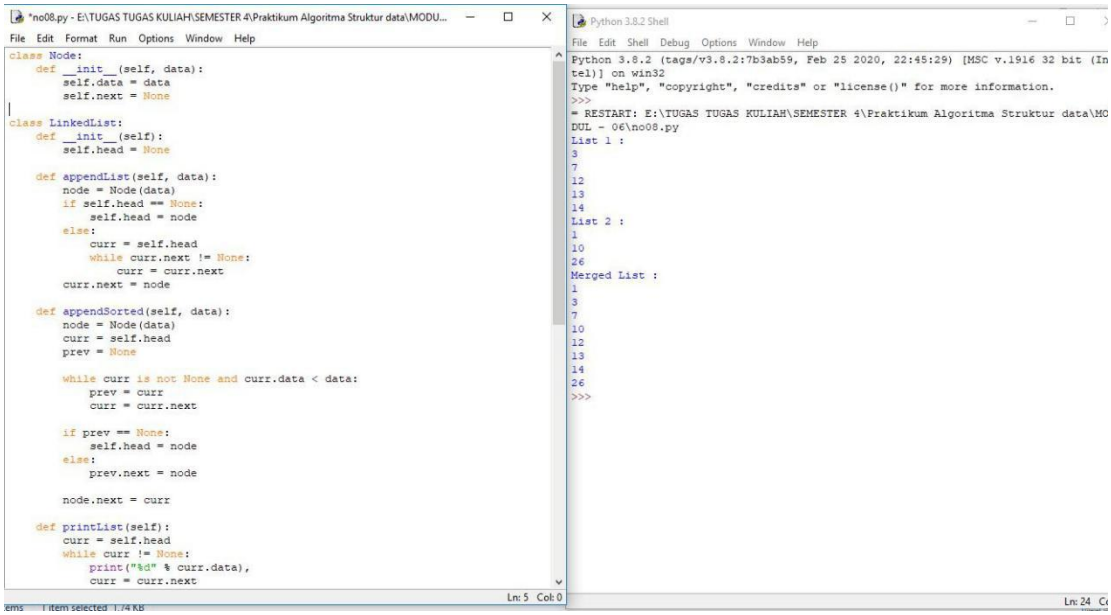
7.



```
from time import time as detak
from random import shuffle as kocok
import no05  # mergeSort baru
import no06  # quickSort baru
import no03  # 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(); no05.merge_sort(merB); ak = detak(); print('merge sort baru : %g d
# Quick Sort baru
aw = detak(); no06.quickSort(quiB); ak = detak(); print('quick sort baru : %g de

# Merge Sort dan Quick Sort awal
aw = detak(); no03.mergeSort(merA); ak = detak(); print('merge sort awal : %g de
aw = detak(); no03.quickSort(quiA); ak = detak(); print('quick sort awal : %g de
```

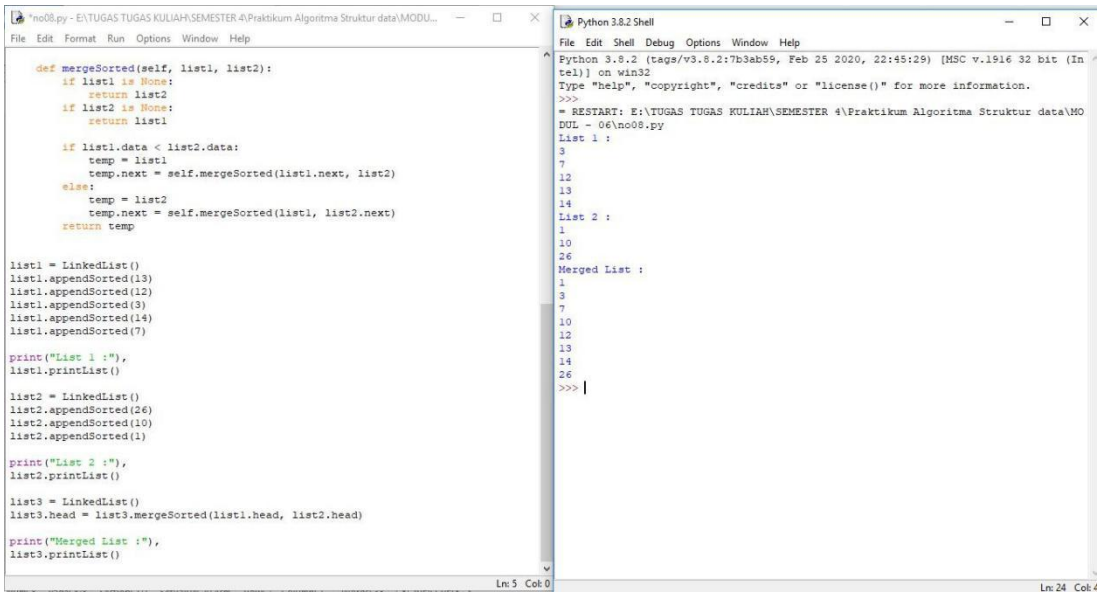
```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no07.py
bubble : 10.7319 detik
selection : 3.98287 detik
insertion : 4.56144 detik
merge : 0.0624835 detik
quick : 0.0468974 detik
merge sort baru : 0.0781064 detik
quick sort baru : 0.0624301 detik
merge sort awal : 0.0624228 detik
quick sort awal : 0.0311494 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),
            curr = curr.next
```

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MO
DUL - 06\no08.py
List 1 :
3
7
12
13
14
List 2 :
1
10
26
Merged List :
1
3
7
10
12
13
14
26
>>>
```


*no08.py - E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODU...
File Edit Format Run Options Window Help
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()
Ln: 5 Col: 0
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\TUGAS TUGAS KULIAH\SEMESTER 4\Praktikum Algoritma Struktur data\MODUL - 06\no08.py
List 1 :
3
7
12
13
14
List 2 :
1
10
26
Merged List :
1
3
7
10
12
13
14
26
>>>
Ln: 24 Col: 4