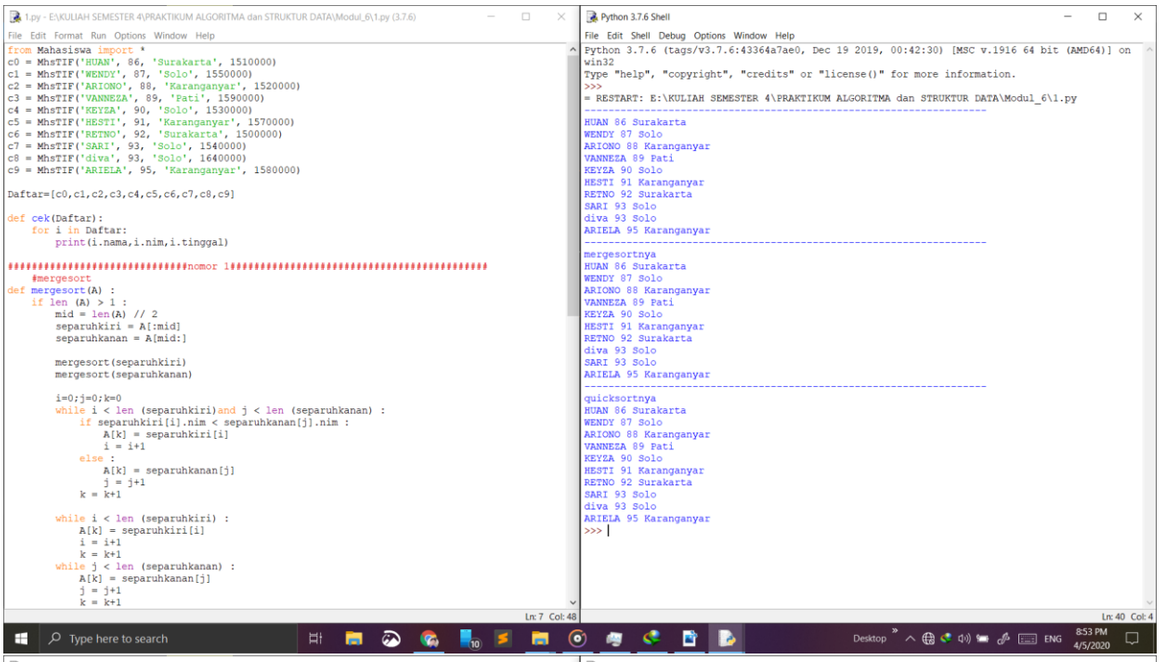
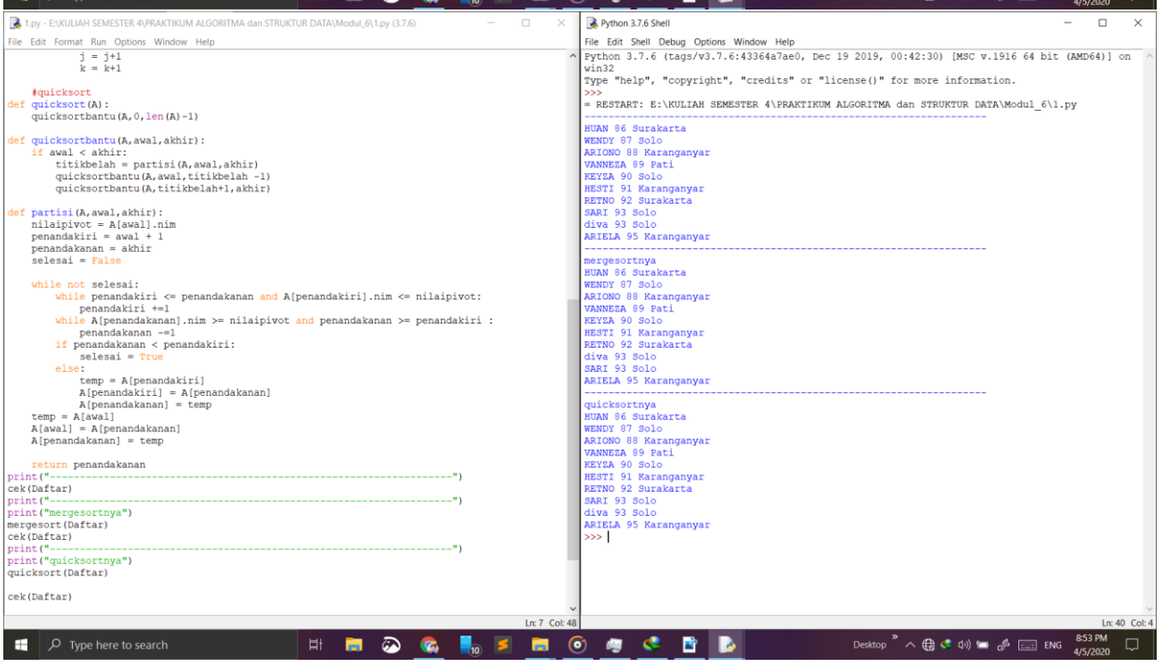
Nama : Huan Wendy Ariono
Nim : L200180086
Kelas : D

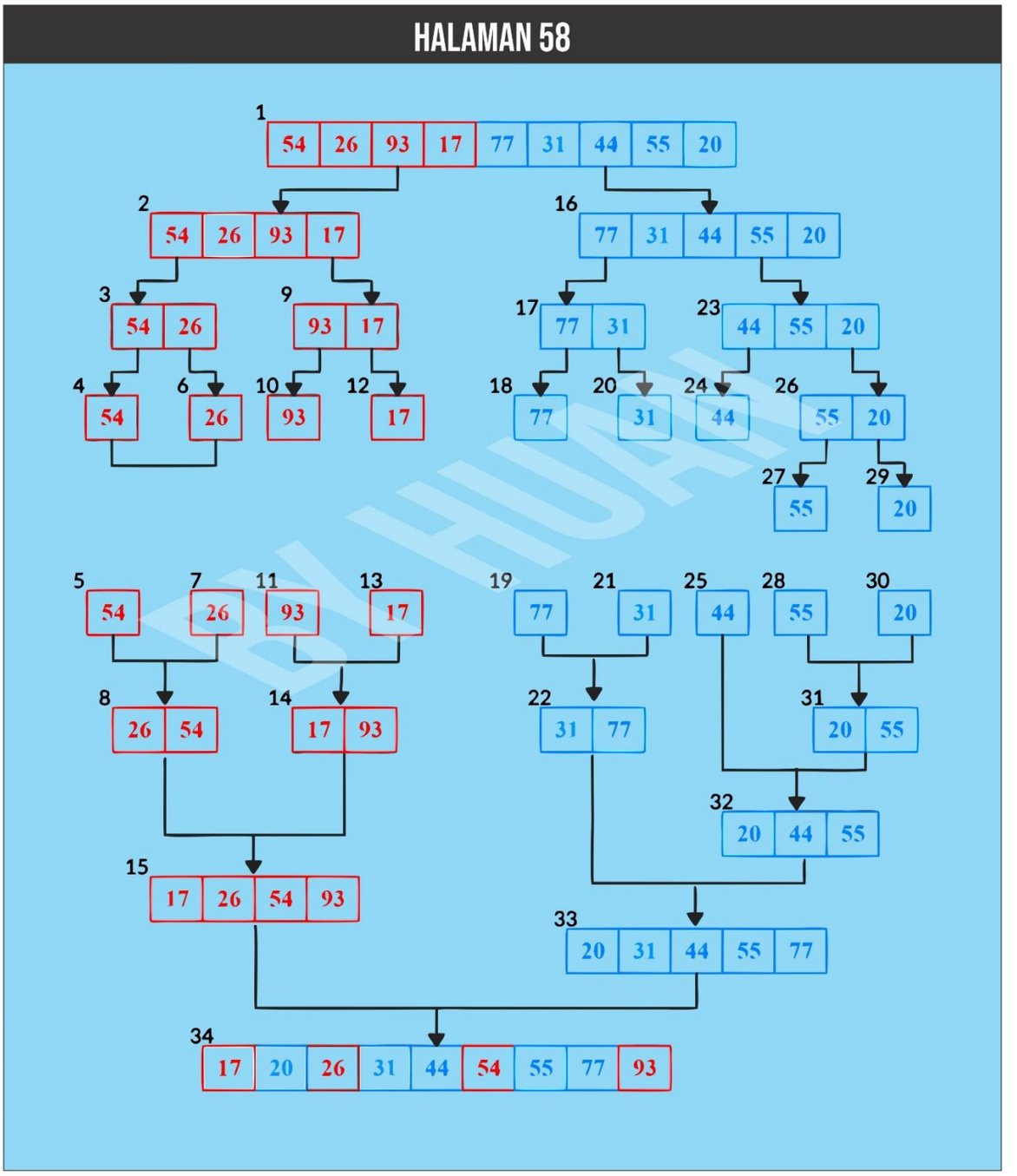
LAPORAN MODUL 6

Nomer 1

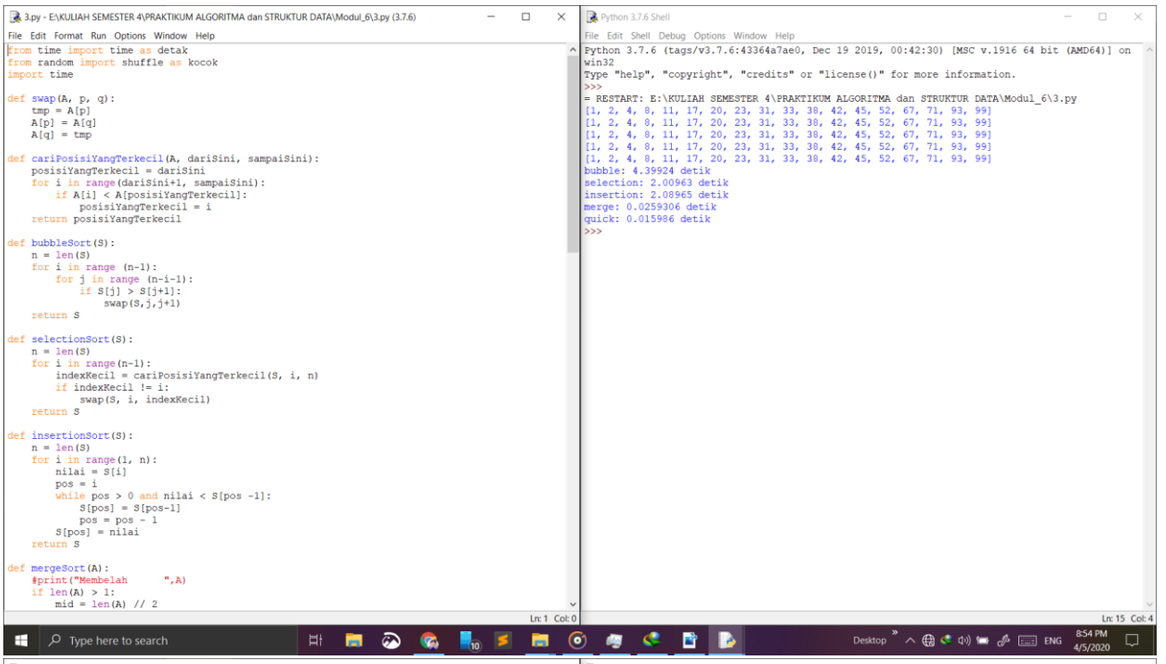


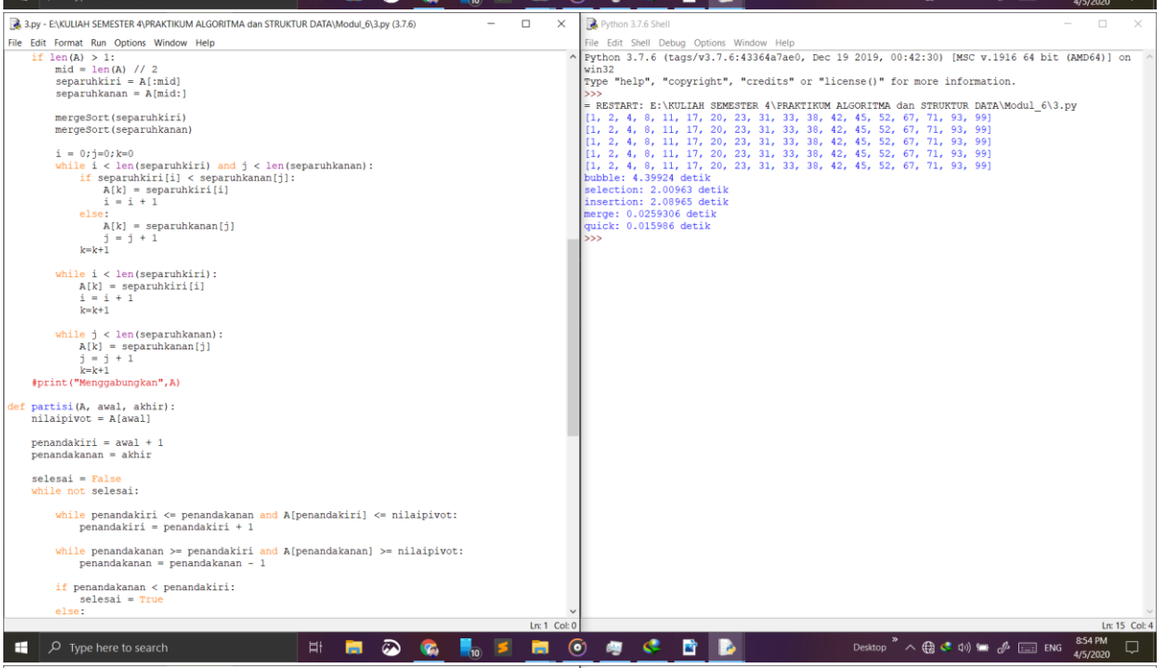


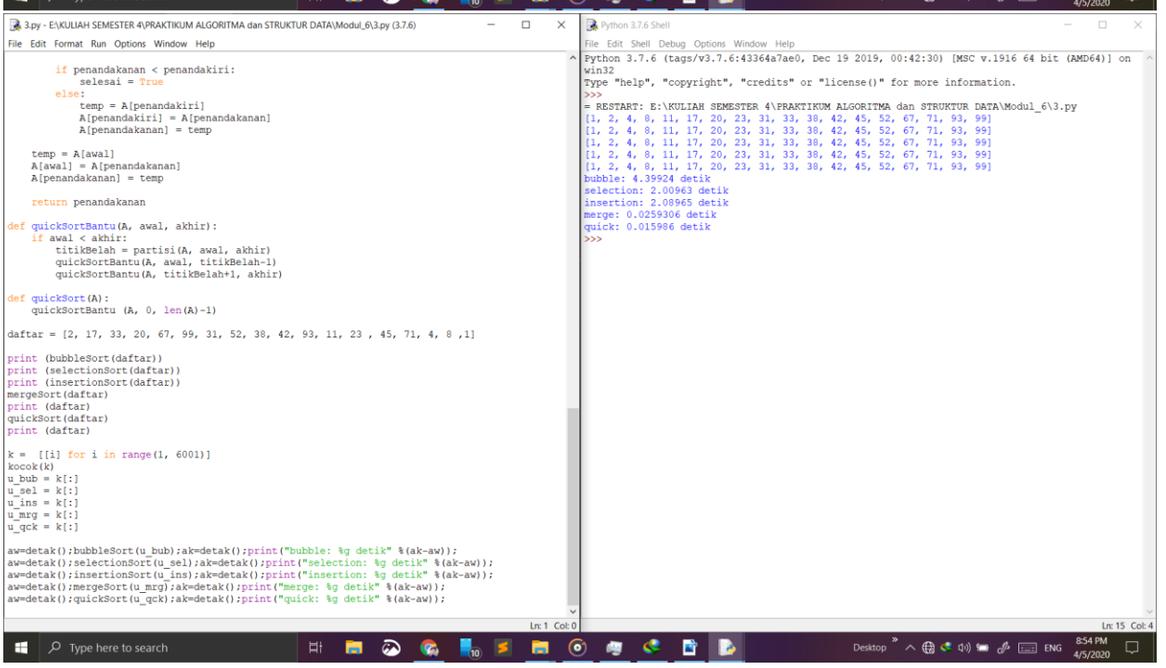
Nomer 2


HALAMAN 58
1
54 26 93 17 77 31 44 55 20
2
54 26 93 17
16
77 31 44 55 20
3
54 26
9
93 17
17
77 31
23
44 55 20
4
54
6
26
10
93
12
17
18
77
20
31
24
44
26
55 20
27
55
29
20
5
54
7
26
11
93
13
17
19
77
21
31
25
44
28
55
30
20
8
26 54
14
17 93
22
31 77
31
20 55
32
20 44 55
15
17 26 54 93
33
20 31 44 55 77
34
17 20 26 31 44 54 55 77 93

Nomer 3

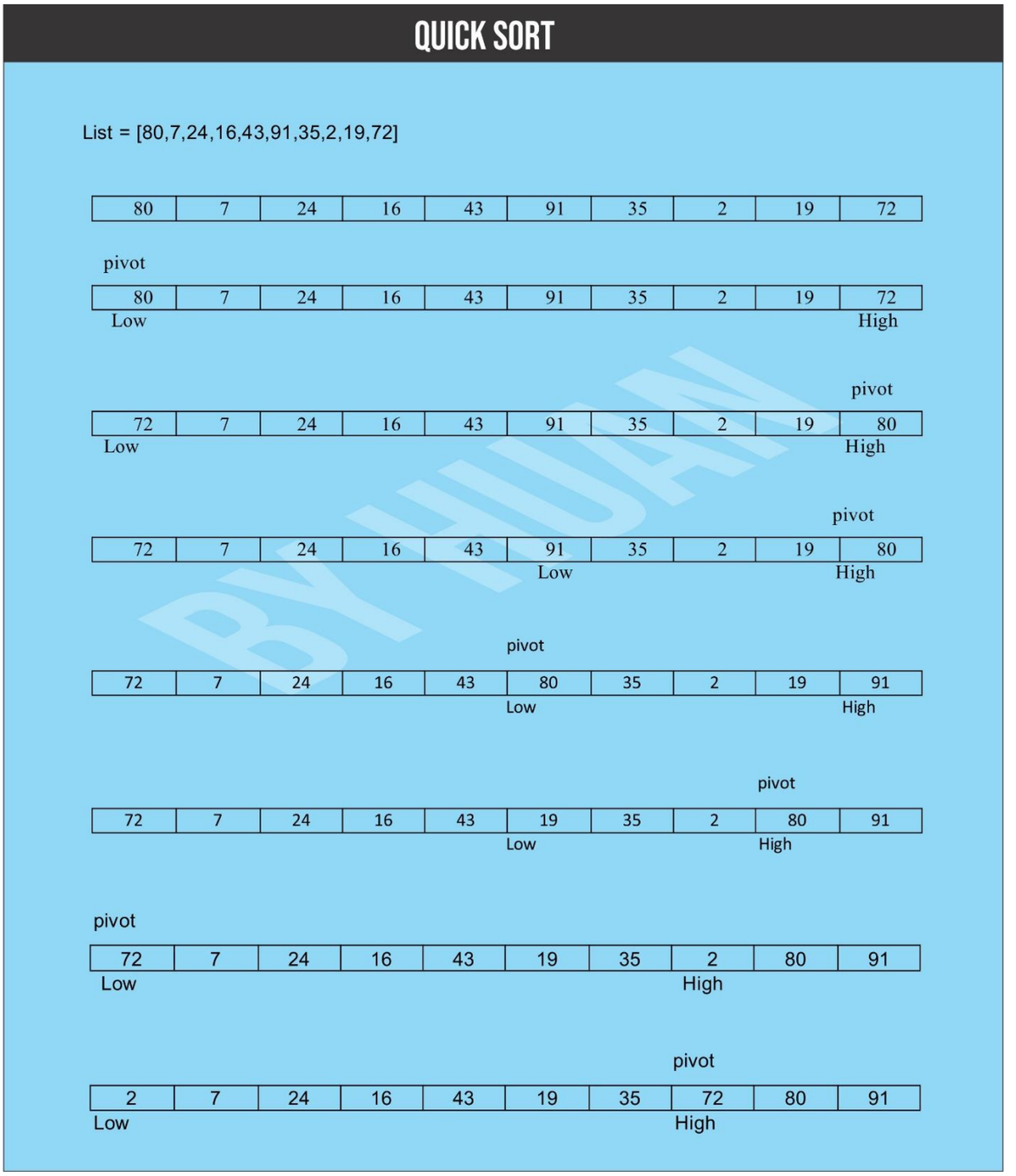

3.py - E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\3.py (3.7.6)
File Edit Format Run Options Window Help
from time import time as detak
from random import shuffle as kocok
import time
def swap(A, p, q):
tmp = A[p]
A[p] = A[q]
A[q] = tmp
def cariPosisiYangTerkecil(A, dariSini, sampaiSini):
posisiYangTerkecil = dariSini
for i in range(dariSini+1, sampaiSini):
if A[i] < A[posisiYangTerkecil]:
posisiYangTerkecil = i
return posisiYangTerkecil
def bubbleSort(S):
n = len(S)
for i in range (n-1):
for j in range (n-i-1):
if S[j] > S[j+1]:
swap(S,j,j+1)
return S
def selectionSort(S):
n = len(S)
for i in range(n-1):
indexKecil = cariPosisiYangTerkecil(S, i, n)
if indexKecil != i:
swap(S, i, indexKecil)
return S
def insertionSort(S):
n = len(S)
for i in range(1, n):
nilai = S[i]
pos = i
while pos > 0 and nilai < S[pos -1]:
S[pos] = S[pos-1]
pos = pos - 1
S[pos] = nilai
return S
def mergeSort(A):
#print("Membelah ",A)
if len(A) > 1:
mid = len(A) // 2
Ln: 1 Col: 0
Python 3.7.6 Shell
File Edit Shell Debug Options Window Help
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\3.py
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
bubble: 4.39924 detik
selection: 2.00963 detik
insertion: 2.08965 detik
merge: 0.0259306 detik
quick: 0.015986 detik
>>>
Ln: 15 Col: 4
Type here to search
Desktop
ENG
8:54 PM
4/5/2020


3.py - E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\3.py (3.7.6)
File Edit Format Run Options Window Help
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 = i + 1
else:
A[k] = separuhkanan[j]
j = j + 1
k=k+1
while i < len(separuhkiri):
A[k] = separuhkiri[i]
i = i + 1
k=k+1
while j < len(separuhkanan):
A[k] = separuhkanan[j]
j = j + 1
k=k+1
#print("Menggabungkan",A)
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 penandakanan >= penandakiri and A[penandakanan] >= nilaipivot:
penandakanan = penandakanan - 1
if penandakanan < penandakiri:
selesai = True
else:
Ln: 1 Col: 0
Python 3.7.6 Shell
File Edit Shell Debug Options Window Help
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\3.py
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
bubble: 4.39924 detik
selection: 2.00963 detik
insertion: 2.08965 detik
merge: 0.0259306 detik
quick: 0.015986 detik
>>>
Ln: 15 Col: 4
Type here to search
Desktop
ENG
8:54 PM
4/5/2020


3.py - E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\3.py (3.7.6)
File Edit Format Run Options Window Help
if penandakanan < penandakiri:
selesai = True
else:
temp = A[penandakiri]
A[penandakiri] = A[penandakanan]
A[penandakanan] = temp
temp = A[awal]
A[awal] = A[penandakanan]
A[penandakanan] = temp
return penandakanan
def quickSortBantu(A, awal, akhir):
if awal < akhir:
titikBelah = partisi(A, awal, akhir)
quickSortBantu(A, awal, titikBelah-1)
quickSortBantu(A, titikBelah+1, akhir)
def quickSort(A):
quickSortBantu (A, 0, len(A)-1)
daftar = [2, 17, 33, 20, 67, 99, 31, 52, 38, 42, 93, 11, 23 , 45, 71, 4, 8 ,1]
print (bubbleSort(daftar))
print (selectionSort(daftar))
print (insertionSort(daftar))
mergeSort(daftar)
print (daftar)
quickSort(daftar)
print (daftar)
k = [[i] for i in range(1, 6001)]
kocok(k)
u_bub = k[:]
u_sel = k[:]
u_ins = k[:]
u_mrg = k[:]
u_qck = k[:]
aw=detak();bubbleSort(u_bub);ak=detak();print("bubble: %g detik" %(ak-aw));
aw=detak();selectionSort(u_sel);ak=detak();print("selection: %g detik" %(ak-aw));
aw=detak();insertionSort(u_ins);ak=detak();print("insertion: %g detik" %(ak-aw));
aw=detak();mergeSort(u_mrg);ak=detak();print("merge: %g detik" %(ak-aw));
aw=detak();quickSort(u_qck);ak=detak();print("quick: %g detik" %(ak-aw));
Ln: 1 Col: 0
Python 3.7.6 Shell
File Edit Shell Debug Options Window Help
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\3.py
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
[1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
bubble: 4.39924 detik
selection: 2.00963 detik
insertion: 2.08965 detik
merge: 0.0259306 detik
quick: 0.015986 detik
>>>
Ln: 15 Col: 4
Type here to search
Desktop
ENG
8:54 PM
4/5/2020

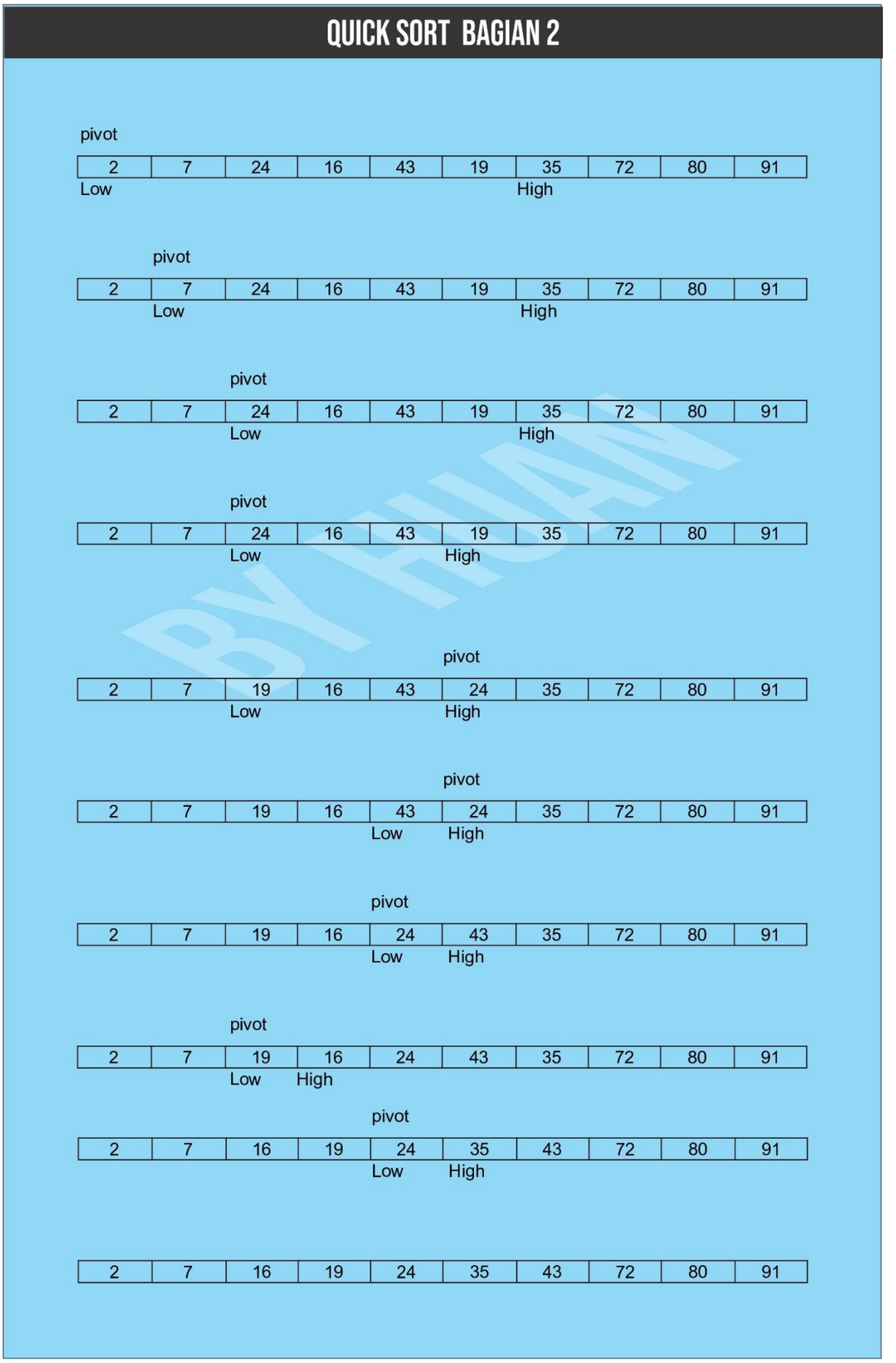
Nomer 4

A

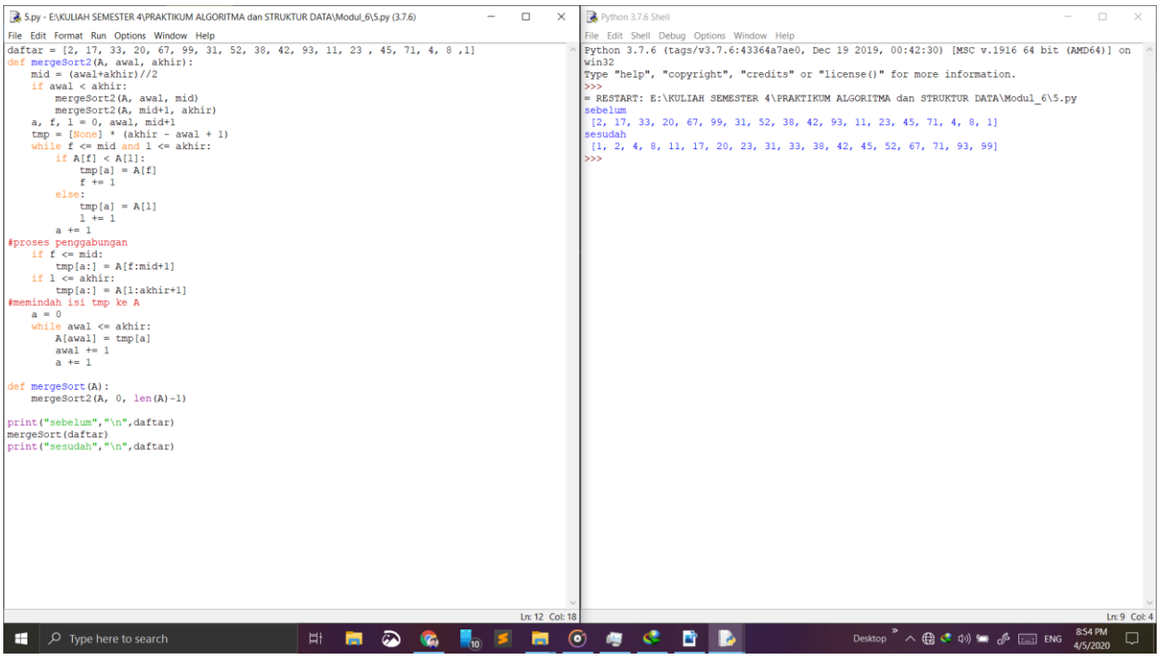

QUICK SORT
List = [80,7,24,16,43,91,35,2,19,72]
80 7 24 16 43 91 35 2 19 72
pivot
80 7 24 16 43 91 35 2 19 72
Low
High
pivot
72 7 24 16 43 91 35 2 19 80
Low
High
pivot
72 7 24 16 43 91 35 2 19 80
Low
High
pivot
72 7 24 16 43 80 35 2 19 91
Low
High
pivot
72 7 24 16 43 19 35 2 80 91
Low
High
pivot
72 7 24 16 43 19 35 2 80 91
Low
High
pivot
2 7 24 16 43 19 35 72 80 91
Low
High

B

QUICK SORT BAGIAN 2



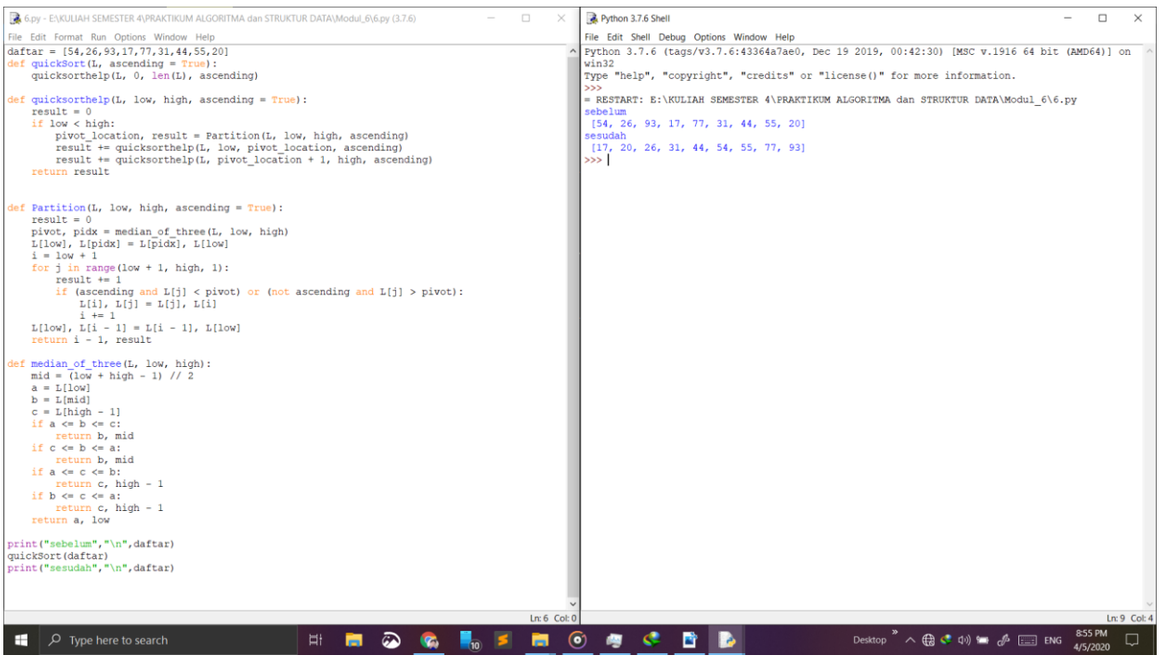
Nomer 5



```
daftar = [2, 17, 33, 20, 67, 99, 31, 52, 38, 42, 93, 11, 23 , 45, 71, 4, 8 ,1]
def mergeSort2(A, awal, akhir):
    mid = (awal+akhir)//2
    if awal < akhir:
        mergeSort2(A, awal, mid)
        mergeSort2(A, mid+1, akhir)
    a, f, l = 0, awal, mid+1
    tmp = [None] * (akhir - awal + 1)
    while f <= mid and l <= akhir:
        if A[f] < A[l]:
            tmp[a] = A[f]
            f += 1
        else:
            tmp[a] = A[l]
            l += 1
        a += 1
#proses penggabungan
    if f <= mid:
        tmp[a:] = A[f:mid+1]
    if l <= akhir:
        tmp[a:] = A[l:akhir+1]
#memindah isi tmp ke A
    a = 0
    while awal <= akhir:
        A[awal] = tmp[a]
        awal += 1
        a += 1

def mergeSort(A):
    mergeSort2(A, 0, len(A)-1)

print("sebelum","\n",daftar)
mergeSort(daftar)
print("sesudah","\n",daftar)
```

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\5.py
sebelum
 [2, 17, 33, 20, 67, 99, 31, 52, 38, 42, 93, 11, 23, 45, 71, 4, 8, 1]
sesudah
 [1, 2, 4, 8, 11, 17, 20, 23, 31, 33, 38, 42, 45, 52, 67, 71, 93, 99]
>>>
```

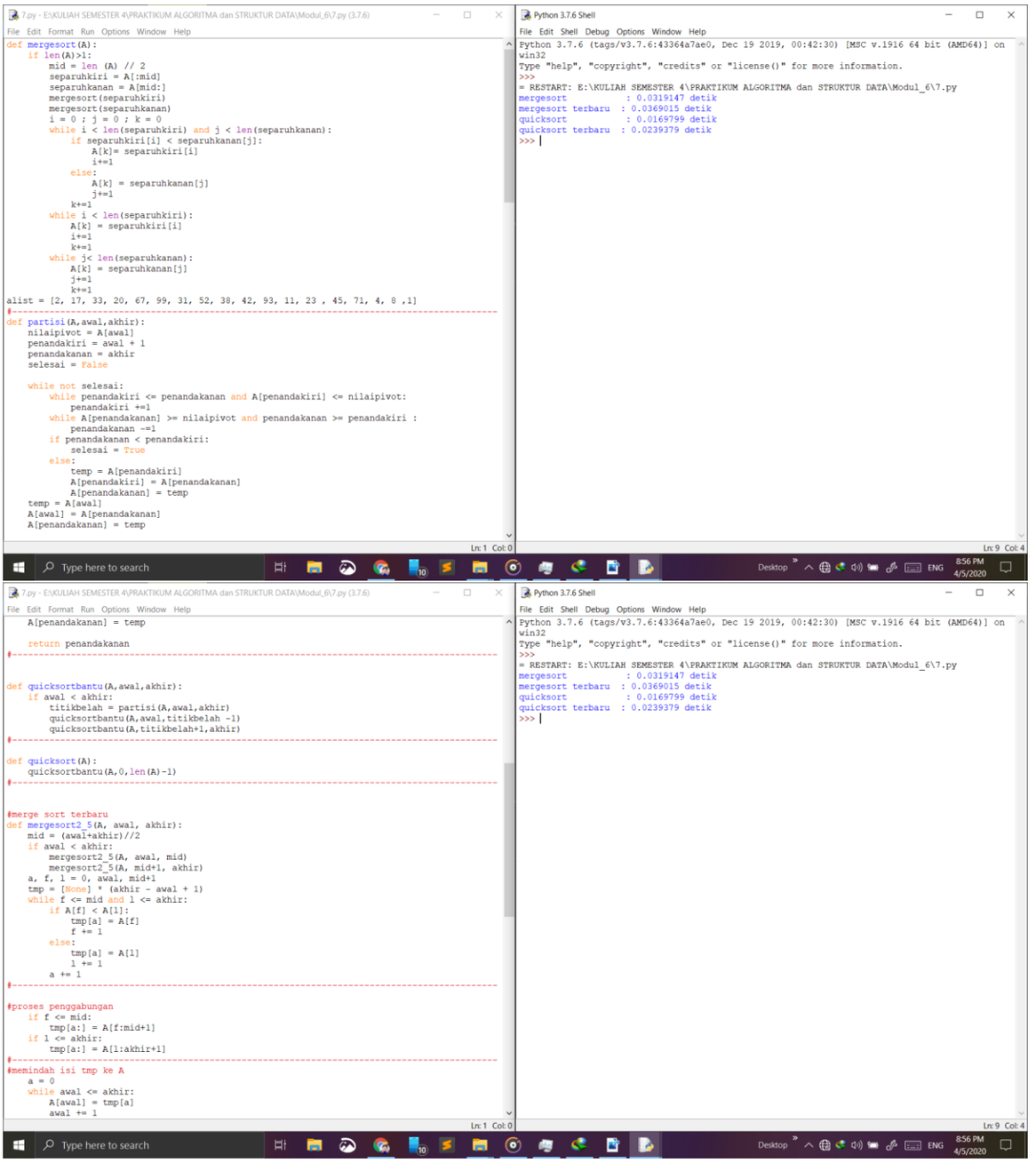
Nomer 6



```
daftar = [54,26,93,17,77,31,44,55,20]
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
    return a, low

print("sebelum","\n",daftar)
quickSort(daftar)
print("sesudah","\n",daftar)
```

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\6.py
sebelum
 [54, 26, 93, 17, 77, 31, 44, 55, 20]
sesudah
 [17, 20, 26, 31, 44, 54, 55, 77, 93]
>>>
```

Nomer 7



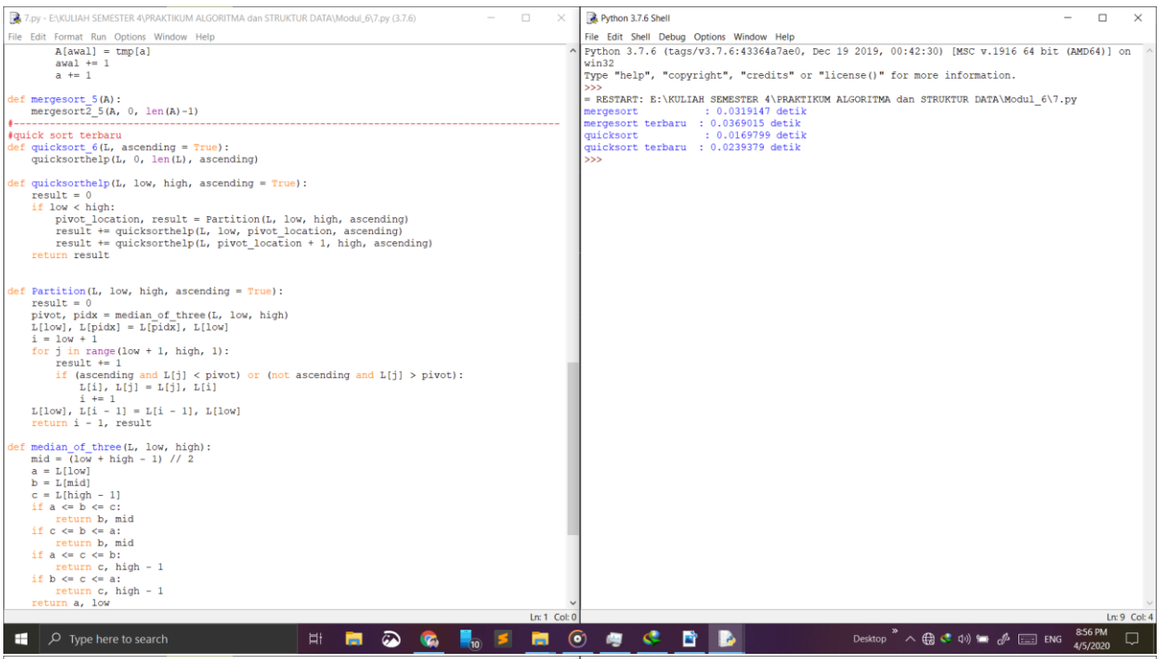
```
def mergesort(A):
    if len(A)>1:
        mid = len (A) // 2
        separuhkiri = A[:mid]
        separuhkanan = A[mid:]
        mergesort(separuhkiri)
        mergesort(separuhkanan)
        i = 0 ; j = 0 ; k = 0
        while i < len(separuhkiri) and j < len(separuhkanan):
            if separuhkiri[i] < separuhkanan[j]:
                A[k]= separuhkiri[i]
                i+=1
            else:
                A[k] = separuhkanan[j]
                j+=1
            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
alist = [2, 17, 33, 20, 67, 99, 31, 52, 38, 42, 93, 11, 23 , 45, 71, 4, 8 ,1]
#-------------------------------------------------------------------------------
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 +=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[penandakanan]
    A[penandakanan] = temp
```

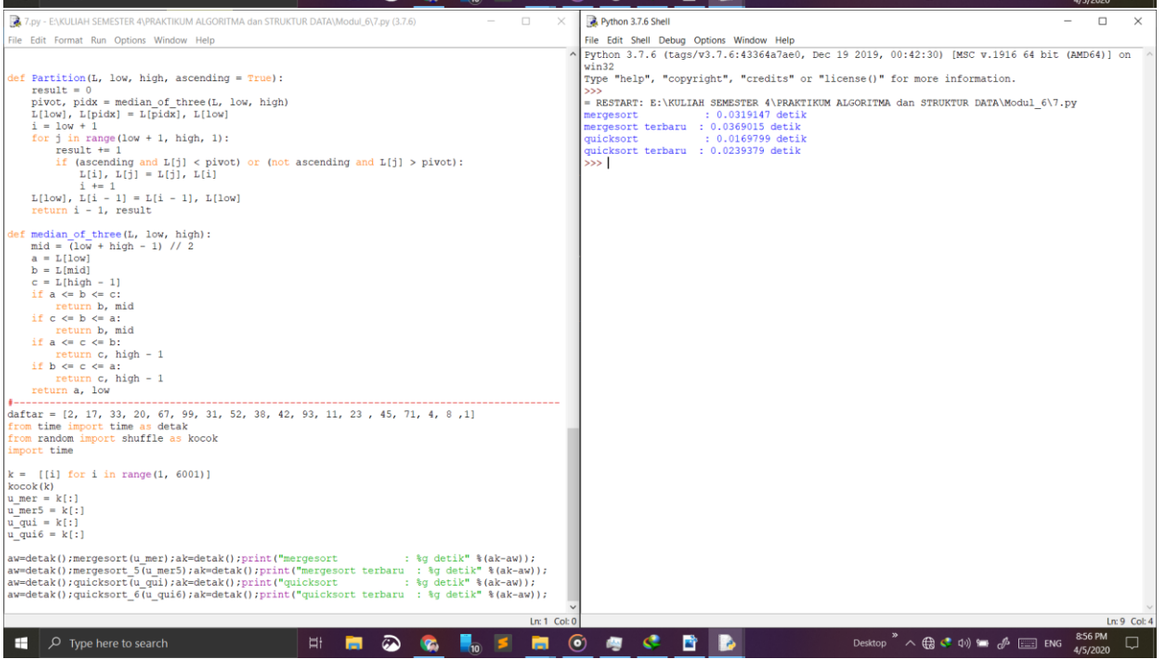
```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\7.py
mergesort           : 0.0319147 detik
mergesort terbaru   : 0.0369015 detik
quicksort           : 0.0169799 detik
quicksort terbaru   : 0.0239379 detik
>>>
```



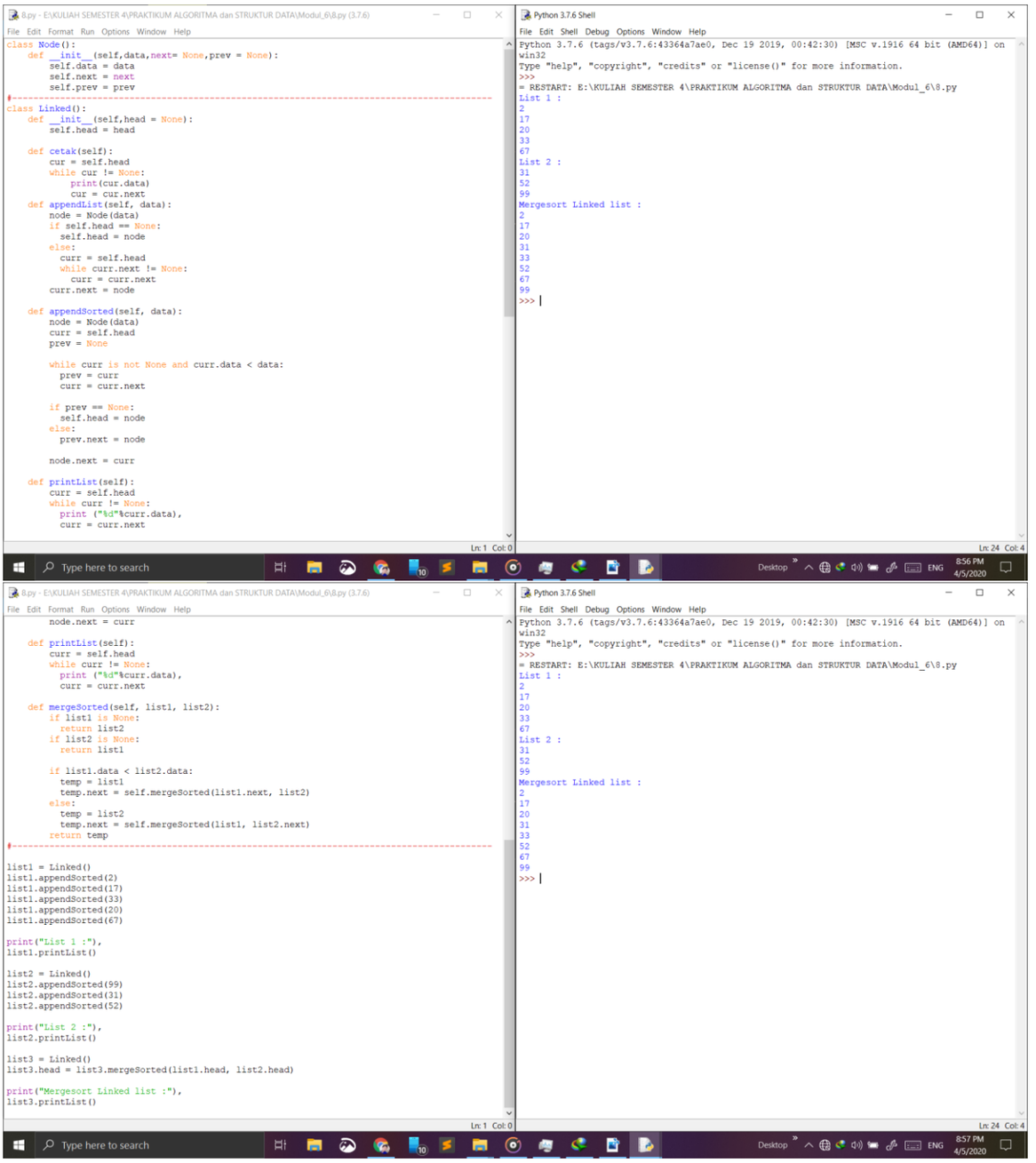
```
    A[penandakanan] = temp

    return penandakanan
#-------------------------------------------------------------------------------


def quicksortbantu(A,awal,akhir):
    if awal < akhir:
        titikbelah = partisi(A,awal,akhir)
        quicksortbantu(A,awal,titikbelah -1)
        quicksortbantu(A,titikbelah+1,akhir)
#-------------------------------------------------------------------------------

def quicksort(A):
    quicksortbantu(A,0,len(A)-1)
#-------------------------------------------------------------------------------


#merge sort terbaru
def mergesort2_5(A, awal, akhir):
    mid = (awal+akhir)//2
    if awal < akhir:
        mergesort2_5(A, awal, mid)
        mergesort2_5(A, mid+1, akhir)
    a, f, l = 0, awal, mid+1
    tmp = [None] * (akhir - awal + 1)
    while f <= mid and l <= akhir:
        if A[f] < A[l]:
            tmp[a] = A[f]
            f += 1
        else:
            tmp[a] = A[l]
            l += 1
        a += 1
#-------------------------------------------------------------------------------

#proses penggabungan
    if f <= mid:
        tmp[a:] = A[f:mid+1]
    if l <= akhir:
        tmp[a:] = A[l:akhir+1]
#-------------------------------------------------------------------------------
#memindah isi tmp ke A
    a = 0
    while awal <= akhir:
        A[awal] = tmp[a]
        awal += 1
```

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\7.py
mergesort           : 0.0319147 detik
mergesort terbaru   : 0.0369015 detik
quicksort           : 0.0169799 detik
quicksort terbaru   : 0.0239379 detik
>>>
```


7.py - E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\7.py (3.7.6)
File Edit Format Run Options Window Help
A[awal] = tmp[a]
awal += 1
a += 1
def mergesort_5(A):
mergesort2_5(A, 0, len(A)-1)
#----------
#quick sort terbaru
def quicksort_6(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
return a, low
Ln: 1 Col: 0
Python 3.7.6 Shell
File Edit Shell Debug Options Window Help
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\7.py
mergesort : 0.0319147 detik
mergesort terbaru : 0.0369015 detik
quicksort : 0.0169799 detik
quicksort terbaru : 0.0239379 detik
>>>
Ln: 9 Col: 4
Type here to search
Desktop
ENG
8:56 PM
4/5/2020


7.py - E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\7.py (3.7.6)
File Edit Format Run Options Window Help
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
return a, low
#----------
daftar = [2, 17, 33, 20, 67, 99, 31, 52, 38, 42, 93, 11, 23 , 45, 71, 4, 8 ,1]
from time import time as detak
from random import shuffle as kocok
import time
k = [[i] for i in range(1, 6001)]
kocok(k)
u_mer = k[:]
u_mer5 = k[:]
u_qui = k[:]
u_qui6 = k[:]
aw=detak();mergesort(u_mer);ak=detak();print("mergesort : %g detik" %(ak-aw));
aw=detak();mergesort_5(u_mer5);ak=detak();print("mergesort terbaru : %g detik" %(ak-aw));
aw=detak();quicksort(u_qui);ak=detak();print("quicksort : %g detik" %(ak-aw));
aw=detak();quicksort_6(u_qui6);ak=detak();print("quicksort terbaru : %g detik" %(ak-aw));
Ln: 1 Col: 0
Python 3.7.6 Shell
File Edit Shell Debug Options Window Help
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\7.py
mergesort : 0.0319147 detik
mergesort terbaru : 0.0369015 detik
quicksort : 0.0169799 detik
quicksort terbaru : 0.0239379 detik
>>>
Ln: 9 Col: 4
Type here to search
Desktop
ENG
8:56 PM
4/5/2020

Nomer 8



```
class Node():
    def __init__(self,data,next= None,prev = None):
        self.data = data
        self.next = next
        self.prev = prev
#--------------------------------------------------------------------------------
class Linked():
    def __init__(self,head = None):
        self.head = head

    def cetak(self):
        cur = self.head
        while cur != None:
            print(cur.data)
            cur = cur.next
    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
        node.next = curr

    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 = Linked()
list1.appendSorted(2)
list1.appendSorted(17)
list1.appendSorted(33)
list1.appendSorted(20)
list1.appendSorted(67)

print("List 1 :"),
list1.printList()

list2 = Linked()
list2.appendSorted(99)
list2.appendSorted(31)
list2.appendSorted(52)

print("List 2 :"),
list2.printList()

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

print("Mergesort Linked list :"),
list3.printList()
```

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\KULIAH SEMESTER 4\PRAKTIKUM ALGORITMA dan STRUKTUR DATA\Modul_6\8.py
List 1 :
2
17
20
33
67
List 2 :
31
52
99
Mergesort Linked list :
2
17
20
31
33
52
67
99
>>>
```