Nama : Diah Fitri Ramadhani

NIM : L200180106

Kelas : D

Tugas Modul 6 Praktikum Algoritma dan Struktur Data

No. 1


```
from MhsTIF import *

c0 = MhsTIF("diah", 12, "Sragen", 240000)
c1 = MhsTIF("siwi", 41, "Tegal", 230000)
c2 = MhsTIF("amanda", 8, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 17, "Magelang", 235000)
c4 = MhsTIF("brian", 62, "Boyolali", 240000)
c5 = MhsTIF("nisa", 51, "Salatiga", 250000)
c6 = MhsTIF("fandi", 15, "Klaten", 245000)
c7 = MhsTIF("fauzi", 64, "Wonogiri", 245000)
c8 = MhsTIF("putri", 43, "Klaten", 245000)
c9 = MhsTIF("nimas", 74, "Karanganyar", 270000)
c10 = MhsTIF("viola", 24, "Purwodadi", 265000)

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]


def cek(Daftar):
    for i in Daftar:
        print(i.nama, i.nim, i.kotaTinggal)

## Merge Sort
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].nim < separuhKanan[j].nim :
                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

## 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].nim
    penandaKiri = awal + 1
    penandaKanan = akhir
    selesai = False

    while not selesai:
        while penandaKiri <= penandaKanan and A[penandaKiri].nim <= nilaiPivot:
            penandaKiri = penandaKiri + 1

        while A[penandaKanan].nim >= nilaiPivot and penandaKanan >= penandaKiri :
            penandaKanan = 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]
```

1.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Prak Algostruk\Modul_6\1.py (3.7.0)

File Edit Format Run Options Window Help

```
    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].nim
    penandaKiri = awal + 1
    penandaKanan = akhir
    selesai = False

    while not selesai:
        while penandaKiri <= penandaKanan and A[penandaKiri].nim <= nilaiPivot:
            penandaKiri = penandaKiri + 1

        while A[penandaKanan].nim >= nilaiPivot and penandaKanan >= penandaKiri :
            penandaKanan = 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

    return penandaKanan

cek(Daftar)
print("====================================")
print("mergesort :")
print("====================================")
mergeSort(Daftar)
cek(Daftar)
print("====================================")
print("quicksort :")
quickSort(Daftar)
print("====================================")
cek(Daftar)
```

Ln: 41 Col: 0

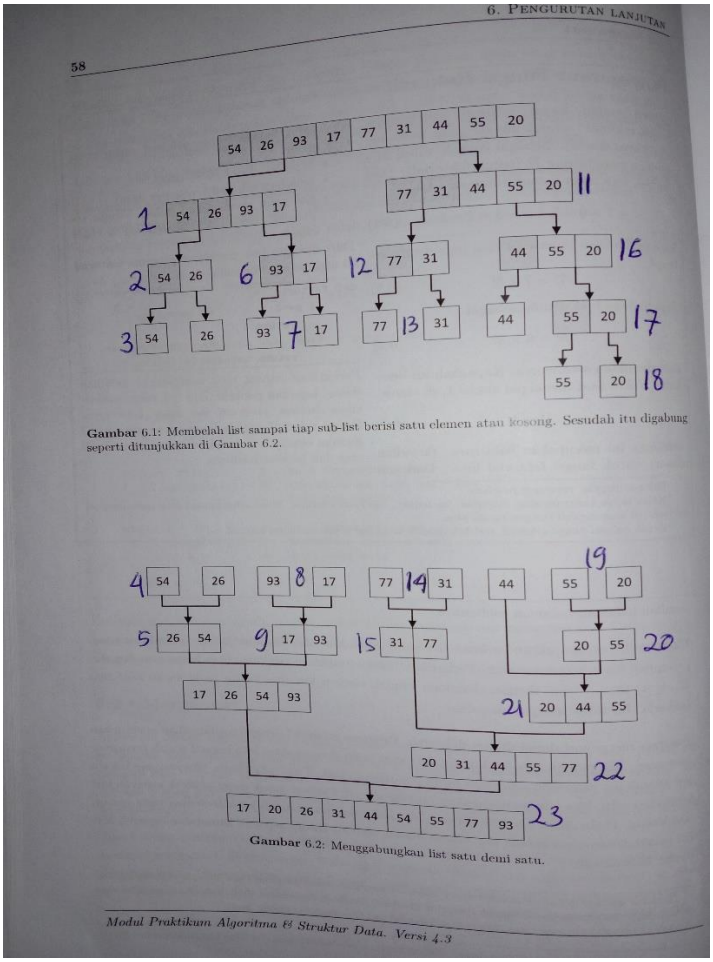
Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
 RESTART: E:\Diah Ramadhani's\KULIAHH\Semester 4\Prak Algostruk\Modul_6\1.py
diah 12 Sragen
siwi 41 Tegal
amanda 8 Surakarta
Chandra 17 Magelang
brian 62 Boyolali
nisa 51 Salatiga
fandi 15 Klaten
fauzi 64 Wonogiri
putri 43 Klaten
nimas 74 Karanganyar
viola 24 Purwodadi
====================================
mergesort :
====================================
amanda 8 Surakarta
diah 12 Sragen
fandi 15 Klaten
Chandra 17 Magelang
viola 24 Purwodadi
siwi 41 Tegal
putri 43 Klaten
nisa 51 Salatiga
brian 62 Boyolali
fauzi 64 Wonogiri
nimas 74 Karanganyar
====================================
quicksort :
====================================
amanda 8 Surakarta
diah 12 Sragen
fandi 15 Klaten
Chandra 17 Magelang
viola 24 Purwodadi
siwi 41 Tegal
putri 43 Klaten
nisa 51 Salatiga
brian 62 Boyolali
fauzi 64 Wonogiri
nimas 74 Karanganyar
>>>
```

Ln: 44 Col: 4

No. 2



6. PENGURUTAN LANJUTAN

58

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.

Modul Praktikum Algoritma & Struktur Data, Versi 4.3

No. 3

No3.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No3.py (3.7.0)

File Edit Format Run Options Window Help

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

k = [i for i in range(1,6001)]
kocok(k)

def bubbleSort(X) :
    n = len(X)
    for i in range(n):
        for j in range(0, n-i-1):
            if X[j] > X[j+1] :
                X[j], X[j+1] = X[j+1], X[j]

def selectionSort(X) :
    for i in range(len(X)):
        min_idk = i
        for j in range(i+1, len(X)):
            if X[min_idk] > X[j]:
                min_idk = j
        X[i], X[min_idk] = X[min_idk], X[i]

def insertSort(X) :
    n = len (X)
    for i in range (1, n) :
        nilai = X[i]
        abc = i-1
        while abc >= 0 and nilai < X[abc-1] :
            X[abc] = X[abc+1]
            abc -=1
        X[abc+1] = nilai

def mergeSort(X):
    if len(X) >1:
        mid = len(X)//2
        L = X[:mid]
        R = X[mid:]
        mergeSort(L)
        mergeSort(R)
        i = j = k = 0
        while i < len(L) and j < len(R):
```

Ln: 1 Col: 0

No3.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No3.py (3.7.0)

File Edit Format Run Options Window Help

```
        while i < len(L) and j < len(R):
            if L[i] < R[j]:
                X[k] = L[i]
                i+=1
            else:
                X[k] = R[j]
                j+=1
            k+=1
        while i < len(L):
            X[k] = L[i]
            i+=1
            k+=1
        while j < len(R):
            X[k] = R[j]
            j+=1
            k+=1

def partition(X,low,high):
    i = ( low-1 )
    pivot = X[high]
    for j in range(low , high):
        if   X[j] <= pivot:
            i = i+1
            X[i],X[j] = X[j],X[i]
    X[i+1],X[high] = X[high],X[i+1]
    return ( i+1 )

def quickSort(X,low,high):
    if low < high:
        pi = partition(X,low,high)
        quickSort(X, low, pi-1)
        quickSort(X, pi+1, high)

u_bub = k[:]
u_sel = k[:]
u_ins = k[:]
u_mer = 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)) ;
```

Ln: 1 Col: 0

```
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 () ; insertSort (u_ins) ; ak = detak() ; print('insert : % g detik' % (ak - aw)) ;
aw = detak () ; mergeSort (u_mer) ; ak = detak() ; print('merge : % g detik' % (ak - aw)) ;
aw = detak () ; quickSort (u_qck, 0, len(u_qck)-1) ; ak = detak() ; print('quick : % g detik' % (ak - aw)) ;
```

Ln: 1 Col: 0

Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No3.py =======
bubble :  9.68386 detik
selection :  4.56142 detik
insert :  0.0468521 detik
merge :  0.078146 detik
quick :  0.0467677 detik
>>>
```

No. 4

a) merge sort

80	7	24	16	43	91	35	2	19	72

Langkah 1

80	7	24	16	43	91	35	2	19	72

Langkah 2

7	16	24	80	2	35	43	91	19	72

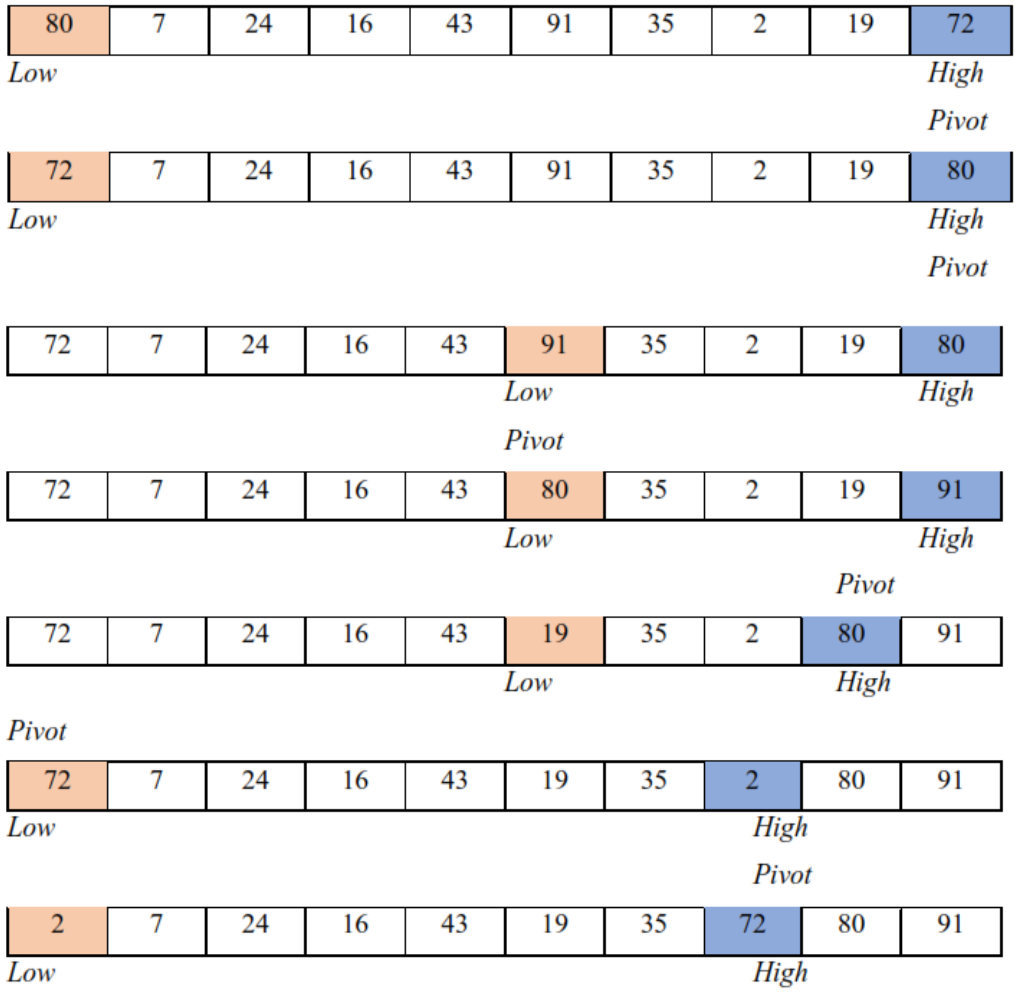
Langkah 3

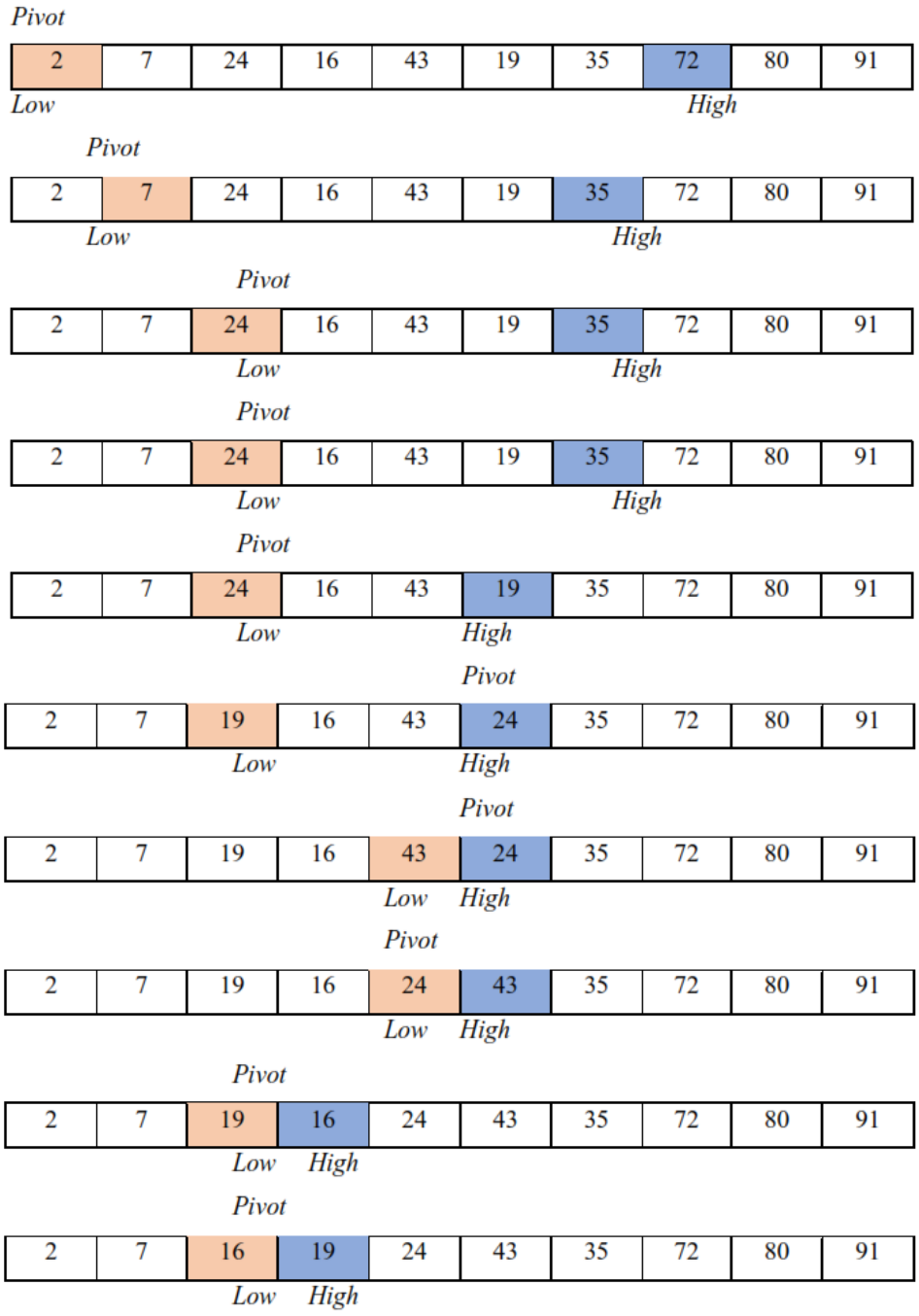
2	7	16	24	35	43	80	91	19	72

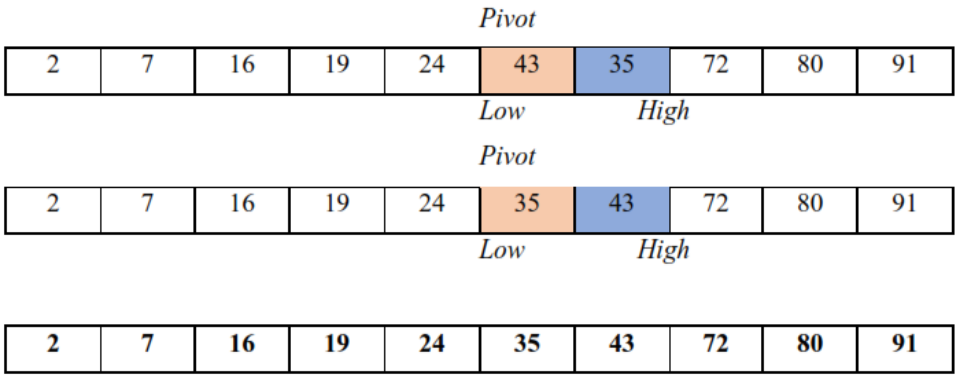
Langkah 4

2	7	16	19	24	35	43	72	80	91

b) quick sort


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


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



No. 5

No5.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No5.py (3.7.0)

File Edit Format Run Options Window Help

```
import random
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
```

Ln: 14 Col: 4

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

print(mergeSort([33,21,9,52, 45, 49]))
```

Ln: 14 Col: 4

Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No5.py =======
[9, 21, 33, 45, 49, 52]
>>>
```

No. 6

No6.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No6.py (3.7.0)

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
    return a, low
```

Ln: 1 Col: 0

```
list1 = list([36,1,4,95,47,68,50,5])

quickSort(list1, False)
print('sorted:')
print(list1)
```

Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No6.py =======
sorted:
[95, 68, 50, 47, 36, 5, 4, 1]
>>>
```

No. 7

No7.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No7.py (3.7.0)

File Edit Format Run Options Window Help

```
from time import time as detak
from random import shuffle as kocok
import time
k = [i for i in range(1,6001)]
kocok(k)


def mergeSort(arr):
    if len(arr) >1:
        mid = len(arr)//2
        L = arr[:mid]
        R = arr[mid:]
        mergeSort(L)
        mergeSort(R)
        i = j = k = 0
        while i < len(L) and j < len(R):
            if L[i] < R[j]:
                arr[k] = L[i]
                i+=1
            else:
                arr[k] = R[j]
                j+=1
            k+=1
        while i < len(L):
            arr[k] = L[i]
            i+=1
            k+=1
        while j < len(R):
            arr[k] = R[j]
            j+=1
            k+=1


def partition(arr,low,high):
    i = ( low-1 )
    pivot = arr[high]
    for j in range(low , high):
        if   arr[j] <= pivot:
            i = i+1
            arr[i],arr[j] = arr[j],arr[i]
    arr[i+1],arr[high] = arr[high],arr[i+1]
```

Ln: 39 Col: 20

No7.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No7.py (3.7.0)

File Edit Format Run Options Window Help

```
def partition(arr,low,high):
    i = ( low-1 )
    pivot = arr[high]
    for j in range(low , high):
        if   arr[j] <= pivot:
            i = i+1
            arr[i],arr[j] = arr[j],arr[i]
    arr[i+1],arr[high] = arr[high],arr[i+1]
    return ( i+1 )


def quickSort(arr,low,high):
    if low < high:
        pi = partition(arr,low,high)
        quickSort(arr, low, pi-1)
        quickSort(arr, pi+1, high)


import random
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])


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:
```

Ln: 39 Col: 20

No7.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No7.py (3.7.0)

File Edit Format Run Options Window Help

```
        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


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


def quickSortMOD(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
```

Ln: 39 Col: 20

No7.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No7.py (3.7.0)

File Edit Format Run Options Window Help

```
    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

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));
```

Ln: 144 Col: 76

Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No7.py =======
merge : 0.0936809 detik
quick : 0.0468645 detik
merge mod : -0.0156217 detik
quick mod : -0.140684 detik
>>>
```

No. 8

No8.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No8.py (3.7.0)

File Edit Format Run Options Window Help

```
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(data)
        else:
            ent = self.head
            currentent = self.head
            while currentent.next != None:
                currentent = currentent.next
            currentent.next = Node(data)
        return self.head

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

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

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

        node.next = currentent

    def printList(self):
        currentent = self.head
        while currentent != None:
```

Ln: 1 Col: 0

No8.py - E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No8.py (3.7.0)

File Edit Format Run Options Window Help

```
    def printList(self):
        currentent = self.head
        while currentent != None:
            print("%d" % currentent.data),
            currentent = currentent.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(12)
list1.appendSorted(35)
list1.appendSorted(23)
list1.appendSorted(9)
list1.appendSorted(1)
print("List 1:"),
list1.printList()

list2 = LinkedList()
list2.appendSorted(30)
list2.appendSorted(17)
list2.appendSorted(3)
print("List 2:"),
list2.printList()

list3 = LinkedList()
list3.head = list3.mergeSorted(list1.head, list2.head)
print("Merged List:"),
list3.printList()
```

Ln: 79 Col: 17

Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: E:\Diah Ramadhani's\KULIAHH\Semester 4\Modul 6\No8.py =======
List 1:
1
9
12
23
35
List 2:
3
17
30
Merged List:
1
3
9
12
17
23
30
35
>>>
```