

Nama = Fawwaz Haidar A.K
NIM = L200183143
Kelas = H

Modul 6 Praktikum Algostruk

1. MergeSort dan QuickSort

a. MergeSort

```
no 1 MergeSort.py - D:\prak algostruk\prakt 6\no 1 MergeSort.py (3.7.4)
File Edit Format Run Options Window Help

class MhsTIF():
    def __init__(self, nim):
        self.nim = nim

    def __str__(self):
        return str(self.nim)

c0 = MhsTIF(10)
c1 = MhsTIF(51)
c2 = MhsTIF(2)
c3 = MhsTIF(18)
c4 = MhsTIF(4)
c5 = MhsTIF(51)
c6 = MhsTIF(13)
c7 = MhsTIF(5)
c8 = MhsTIF(23)
c9 = MhsTIF(64)
c10 = MhsTIF(29)

c0.next = c1
c1.next = c2
c2.next = c3
c3.next = c4
c4.next = c5
c5.next = c6
c6.next = c7
c7.next = c8
c8.next = c9
c9.next = c10

def mergeSort(A):
    print("Mengeksekusi mergeSort")
    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 array A")

def convert(arr, obj):
    hasil = []
    for x in range(len(arr)):
        for i in range(len(obj)):
            if arr[x] == obj[i].nim:
                hasil.append(obj[i])
    return hasil

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
A = []
for x in Daftar:
    A.append(x.nim)

print("MERGE SORT")
mergeSort(A)
for x in convert(A, Daftar):
    print(x.nim)
```

b. QuickSort

```
no 2 QuickSort.py - D:\praktik\struktur\praktik\2 QuickSort.py (3,7,4)
File Edit Format Run Options Window Help

class MhsTIF():
    def __init__(self, nim):
        self.nim = nim

    def __str__(self):
        return str(self.nim)

c0 = MhsTIF(10)
c1 = MhsTIF(51)
c2 = MhsTIF(2)
c3 = MhsTIF(18)
c4 = MhsTIF(4)
c5 = MhsTIF(31)
c6 = MhsTIF(13)
c7 = MhsTIF(6)
c8 = MhsTIF(23)
c9 = MhsTIF(64)
c10 = MhsTIF(29)
c0.next = c1
c1.next = c2
c2.next = c3
c3.next = c4
c4.next = c5
c5.next = c6
c6.next = c7
c7.next = c8
c8.next = c9
c9.next = c10

def partisi(A, awal, akhir):
    nilai_pivot = A[awal]

    penandakiri = awal + 1
    penandakanan = akhir
    selesai = False
    while not selesai:
        while penandakiri <= penandakanan and A[penandakiri] <= nilai_pivot:
            penandakiri = penandakiri + 1
        while penandakanan >= penandakiri and A[penandakanan] >= nilai_pivot:
            penandakanan = penandakanan - 1
        if penandakiri < penandakanan:
            selesai = True
            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:
        titik_dibelah = partisi(A, awal, akhir)
        quickSortBantu(A, awal, titik_dibelah - 1)
        quickSortBantu(A, titik_dibelah + 1, akhir)

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

def convert(arr, obj):
    hasil = []
    for i in range(len(arr)):
        for j in range(len(obj)):
            if arr[i] == obj[j].nim:
                hasil.append(obj[j])
    return hasil

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
A = []
for x in Daftar:
    A.append(x.nim)
print("Quick Sort")
quickSort(A)
for x in convert(A, Daftar):
    print(x.nim)
```

2. Proses MergeSort

```

1 def mergeSort(A):
2     #print("Membelah", A)
3     if len(A) > 1:
4         mid = len(A) // 2
5         separuhKiri = A[:mid]
6         separuhKanan = A[mid:]
7         # mergeSort(separuhKiri) # Ini rekursi. Memanggil lebih lanjut mergeSort
8         # mergeSort(separuhKanan) # untuk separuhKiri dan separuhKanan.
9
10    # Di bawah ini adalah proses penggabungan.
11    i=0 ; j=0 ; k=0
12    while i < len(separuhKiri) and j < len(separuhKanan):
13        if separuhKiri[i] < separuhKanan[j]: # while-loop ini
14            A[k] = separuhKiri[i]
15            i = i + 1
16        else:
17            A[k] = separuhKanan[j]
18            j = j + 1
19            k=k+1
20
21    while i < len(separuhKiri):
22        A[k] = separuhKiri[i]
23        i = i + 1
24        k = k + 1
25
26    while j < len(separuhKanan):
27        A[k] = separuhKanan[j]
28        j = j + 1
29        k = k + 1
30
31    #print("Menggabungkan", A)

```

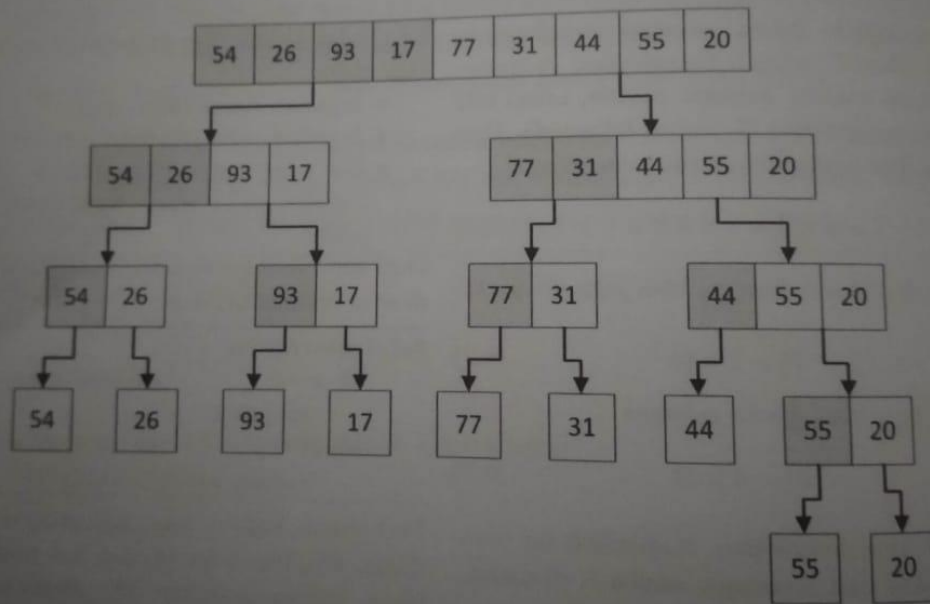
Larikan program di atas dengan memanggilnya seperti ini

```

alist = [54,26,93,17,77,31,44,55,20]
mergeSort(alist)
print(alist)

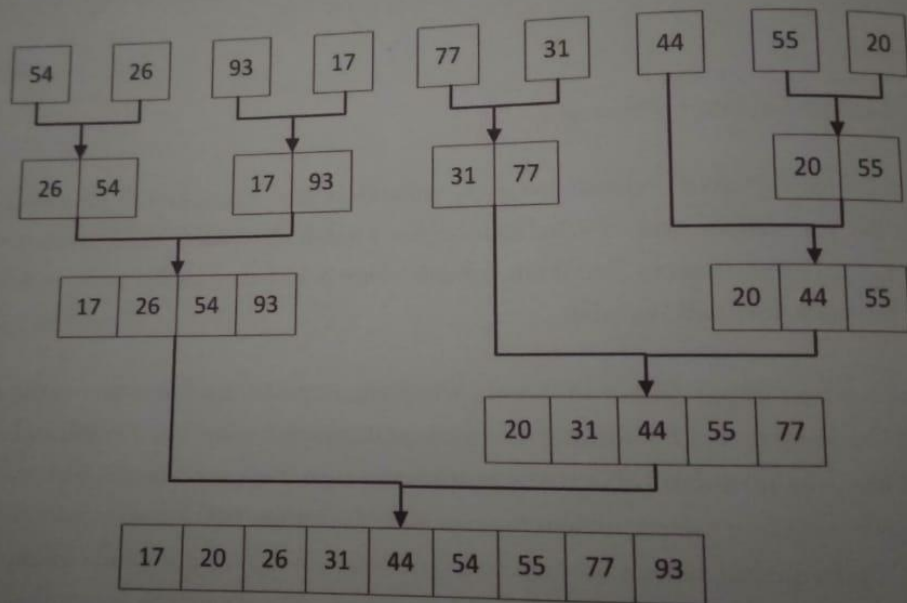
```

6.1



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

6.2



Gambar 6.2: Menggabungkan list satu demi satu.

3. Uji kecepatan MergeSort dan QuickSort

```
no 3.py - D:\prak algoritma\prakt 6\no 3.py (3,74)
File Edit Format Run Options Window Help

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:
            temp = A[penandakiri]
            A[penandakiri] = A[penandakanan]
            A[penandakanan] = temp

    temp = A[awal]
    A[awal] = A[penandakanan]
    A[penandakanan] = temp

no 3.py - D:\prak algoritma\prakt 6\no 3.py (3,74)
File Edit Format Run Options Window Help

from time import time as detik
from random import shuffle as kocok
import time

def temp(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], S[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

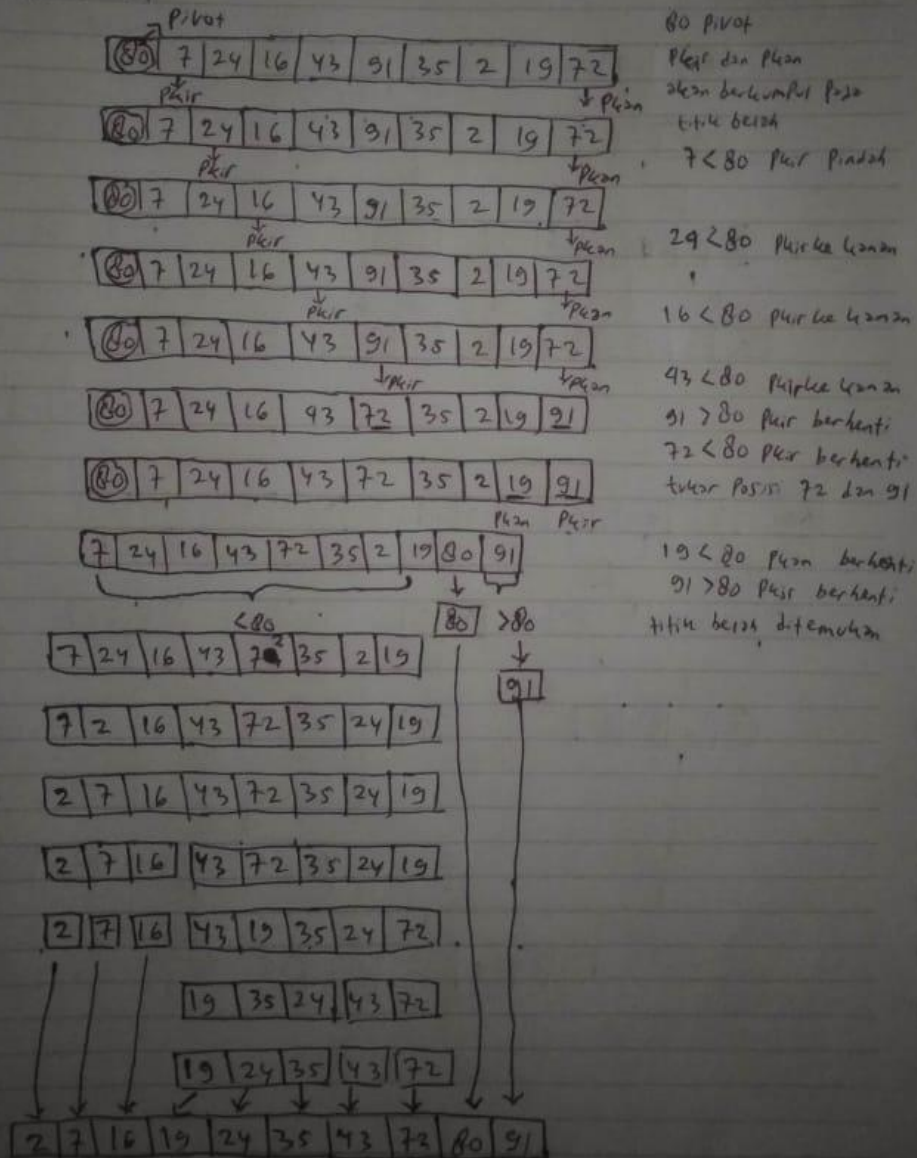
def insertiionSort(S):
    n = len(S)
    for i in range(1, n):
        nilai = S[i]
        pos = i
        while pos > 0 and nilai < S[pos-1]:
            S[pos] = S[pos-1]
            pos = pos - 1
        S[pos] = nilai
    return S

def mergeSort(A):
    print("Memanggil", A)
    if len(A) > 1:
        mid = len(A) // 2
        separuhkiri = A[:mid]
        separuhkanan = A[mid:]

no 3.py - D:\prak algoritma\prakt 6\no 3.py (3,74)
File Edit Format Run Options Window Help
```


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

b.) Quick Sort



5. Mengefisienkan fungsi MergeSort

```

no 5.py -D:\prak algoritma\praktik 6\no 5.py (3,74)
File Edit Format Run Options Window Help

class Mahasiswa():
    def __init__(self, nama, nim, kota, us):
        self.nama = nama
        self.nim = nim
        self.kota = kota
        self.us = us

    def __str__(self):
        s = self.nama + ' NIM'+str(self.nim)\
            + '\nTinggal di ' + self.kota + '\n' \
            + 'Uang Saku Rp. ' + str(self.us) \
            + '\Rp bulannya.'
        return s

    def ambilKota(self):
        return self.kota
    def ambilNama(self):
        return self.nama
    def ambilNim(self):
        return self.nim
    def ambilUangSaku(self):
        return self.us

c0 = Mahasiswa("Rika", 10, "Sukoharjo", 240000)
c1 = Mahasiswa("Budi", 51, "Sliparan", 230000)
c2 = Mahasiswa("Adama", 2, "Surakarta", 250000)
c3 = Mahasiswa("Chandra", 18, "Surakarta", 235000)
c4 = Mahasiswa("Eka", 4, "Boyolali", 240000)
c5 = Mahasiswa("Fandi", 31, "Salatiga", 245000)
c6 = Mahasiswa("Dian", 13, "Klaten", 245000)
c7 = Mahasiswa("Diah", 5, "Wonorejo", 245000)
c8 = Mahasiswa("Cemara", 23, "Nabun", 245000)
c9 = Mahasiswa("Hasan", 64, "Karanganyar", 270000)
c10 = Mahasiswa("Khaid", 29, "Purwodadi", 265000)

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

def cetak(A):
    for i in A:
        print(i)

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].ambilUangSaku() < A[l].ambilUangSaku():
        tmp[a] = A[f]

```

```

No 5.py - D:\prak algoritma\praktik 6\No 5.py (3.74)
File Edit Format View Options Window Help

c0 = MhsTIF("Rus", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Rusdi", 51, "Sragen", 230000)
c2 = MhsTIF("Adama", 2, "Surakarta", 260000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 39, "Solo", 240000)
c6 = MhsTIF("Dian", 13, "Klaten", 245000)
c7 = MhsTIF("Dianah", 5, "Wonorejo", 240000)
c8 = MhsTIF("Chani", 23, "Ngabon", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khaid", 29, "Purwodadi", 265000)

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

def cetak(A):
    for i in A:
        print(i)

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] ambilJangSaku() < A[l] ambilJangSaku():
            tmp[a] = A[f]
            f += 1
        else:
            tmp[a] = A[l]
            l += 1
        a += 1
    if f <= mid:
        tmp[a:] = A[f:mid+1]
    if l <= akhir:
        tmp[a:] = A[l:akhir+1]
    a = 0
    while awal < akhir:
        A[awal] = tmp[a]
        awal += 1
        a += 1
    def mergeSort(A):
        mergeSort2(A, 0, len(A)-1)

```


6. Uji efisiensi fungsi quick sort dengan metode median-dari-tiga untuk memilih pivot

```
no 6.py - D:\prak algoritma\prak 6\no 6.py (3,74)
File Edit Format Run Options Window Help

class MsisTIF():
    def __init__(self, nama, nim, kota, us):
        self.nama = nama
        self.nim = nim
        self.kota = kota
        self.us = us

    def __str__(self):
        s = self.nama + ", NIM:" + str(self.nim) + \
            " Tinggal di " + self.kota + \
            " Usia satu Rp.:" + str(self.us) + \
            " Ikon bulatnya:"
        return s

    def ambilNama(self):
        return self.nama
    def ambilNim(self):
        return self.nim
    def ambilKangSaku(self):
        return self.us

c0 = MsisTIF("Risa", 10, "Sukoharjo", 240000)
c1 = MsisTIF("Dendi", 51, "Surabaya", 230000)
c2 = MsisTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MsisTIF("Chandra", 18, "Surakarta", 235000)
c4 = MsisTIF("Eka", 4, "Boyolali", 240000)
c5 = MsisTIF("Fandi", 31, "Salaliga", 250000)
c6 = MsisTIF("Dini", 13, "Klaten", 245000)
c7 = MsisTIF("Cahya", 5, "Wonorejo", 245000)
c8 = MsisTIF("Janto", 23, "Klaten", 245000)
c9 = MsisTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MsisTIF("Ikbal", 29, "Purwodadi", 265000)

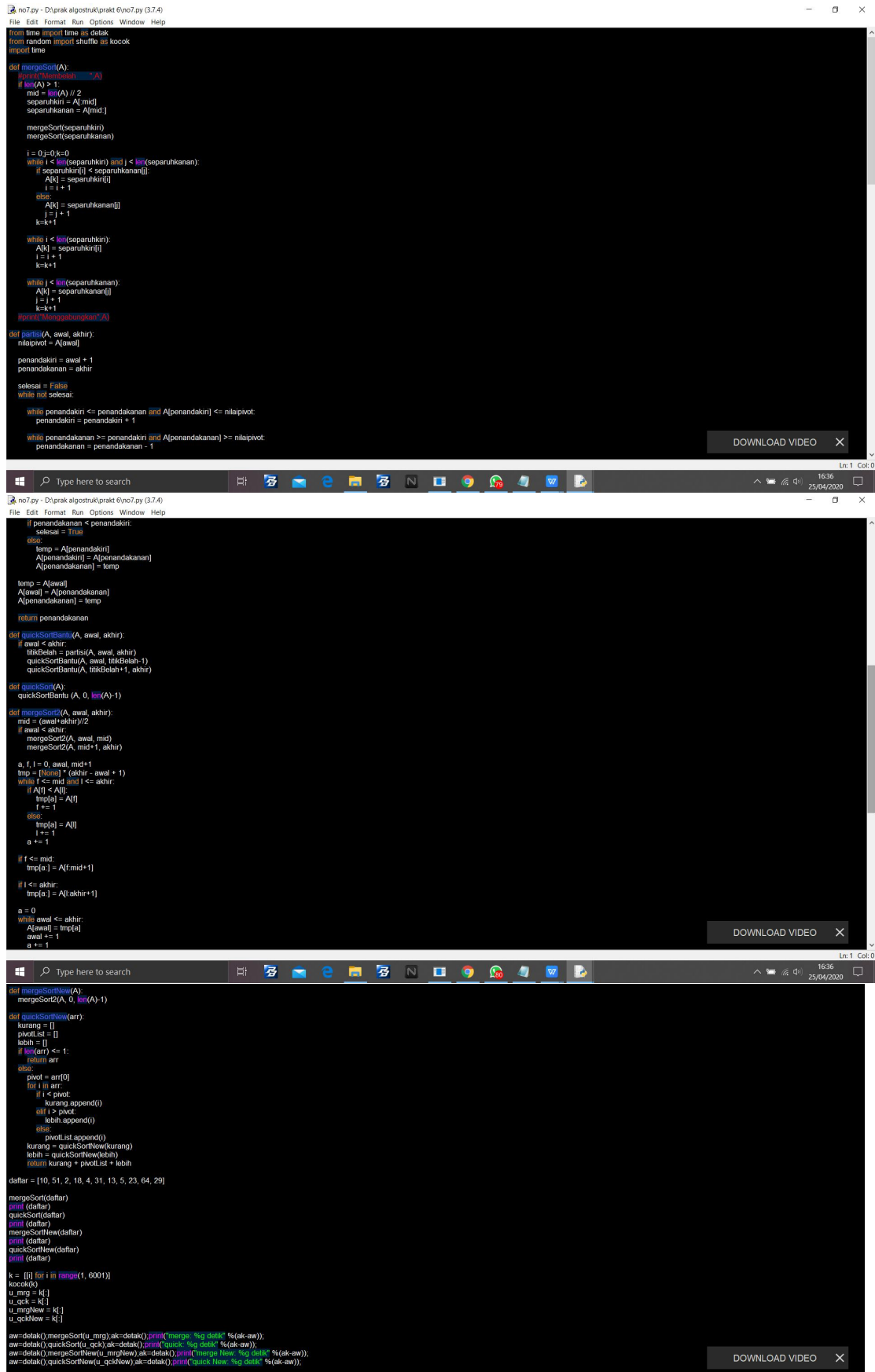
Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
A = []
for i in Daftar:
    A.append(i.nama)

def cetak():
    for i in A:
        print(i)

def quickSort(arr):
    kurang = []
    pivot_list = []
    lebih = []
    if len(arr) <= 1:
        return arr
    else:
        pivot = arr[0]
        for i in arr:
            if i < pivot:
                kurang.append(i)
            elif i > pivot:
                lebih.append(i)
            else:
                pivot_list.append(i)
        kurang = quickSort(kurang)
        lebih = quickSort(lebih)
        return kurang + pivot_list + lebih

print("Sebelum diurutkan")
cetak()
print("Setelah diurutkan")
quickSort(A)
cetak()
```

7. Menguji kecepatan sebelum efisiensi dan setelah efisiensi kan



8. Versi merge sort untuk linked list

```
File Edit Format Insert View Help
no 8.py - D:\prak algoritma\praktik no 8.py (3.74)

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

def print(head):
    curr = head
    while curr is not None:
        try:
            print(curr.data)
            curr = curr.tautan
        except:
            pass

a = Node(1)
b = Node(3)
c = Node(5)
d = Node(7)
e = Node(2)
f = Node(4)
g = Node(6)

a.tautan = b
b.tautan = c
c.tautan = d
d.tautan = e
e.tautan = f
f.tautan = g

def mergeSortLL(A):
    linked = A
    try:
        daftar = []
        curr = A
        while curr:
            daftar.append(curr.data)
            curr = curr.tautan
        A = daftar
    except:
        A = A

    if len(A) > 1:
        mid = len(A) // 2
        separuhkiri = A[:mid]
        separuhkanan = A[mid:]

        mergeSortLL(separuhkiri)
        mergeSortLL(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

    for x in A:
        try:
            linked.data = x
            linked = linked.tautan
        except:
            pass

mergeSortLL(a)
cetak(a)
```