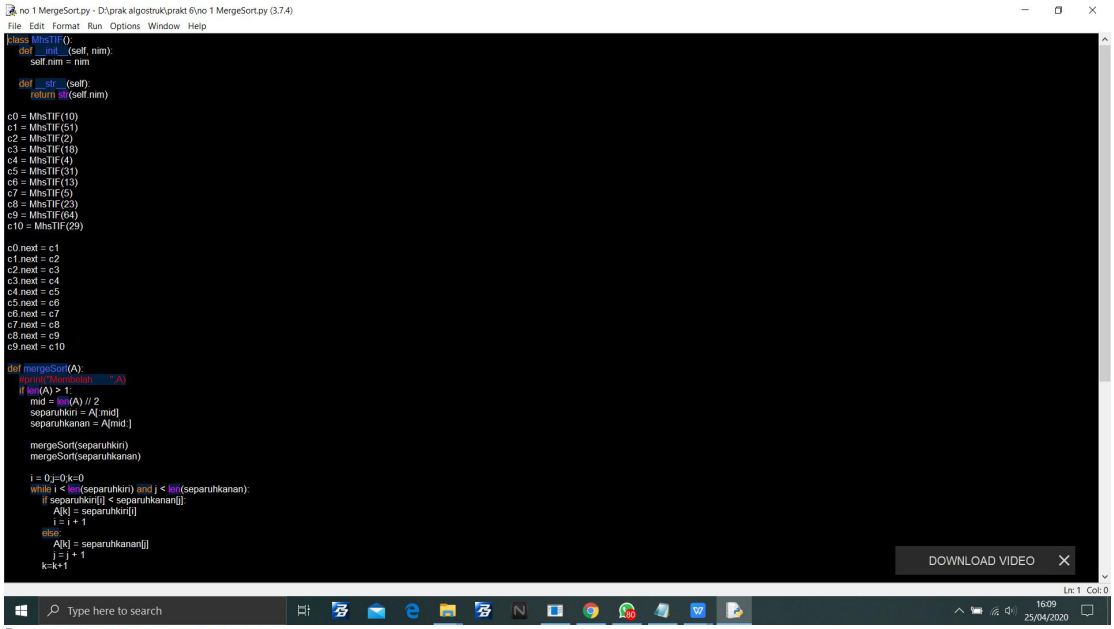
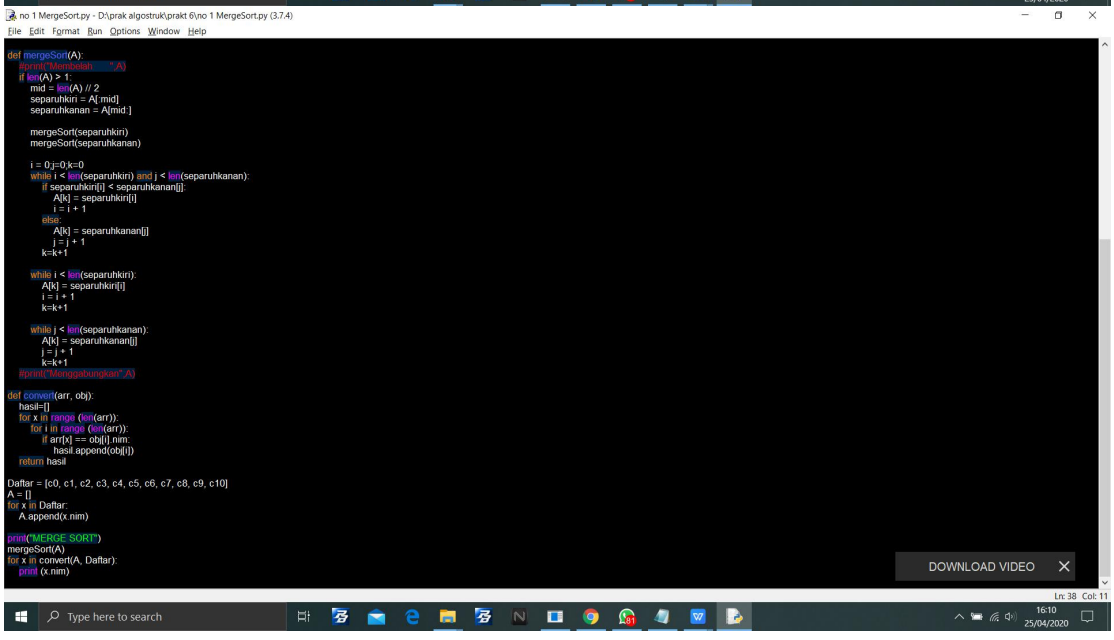
Nama = Fawwaz Haidar A.K
NIM = L200183143
Kelas = H

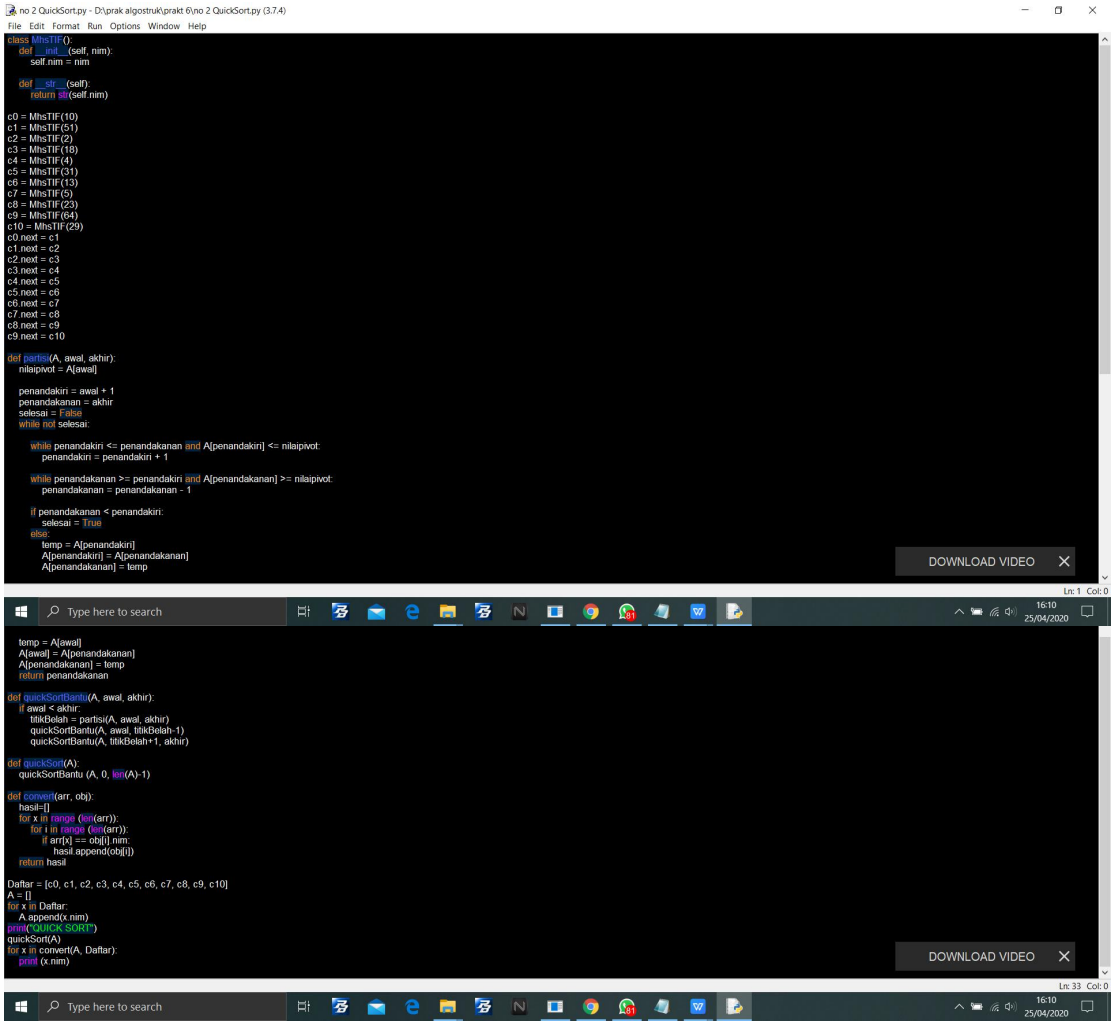
Modul 6
Praktikum Algostruk

1. MergeSort dan QuickSort
   a. MergeSort





b. QuickSort



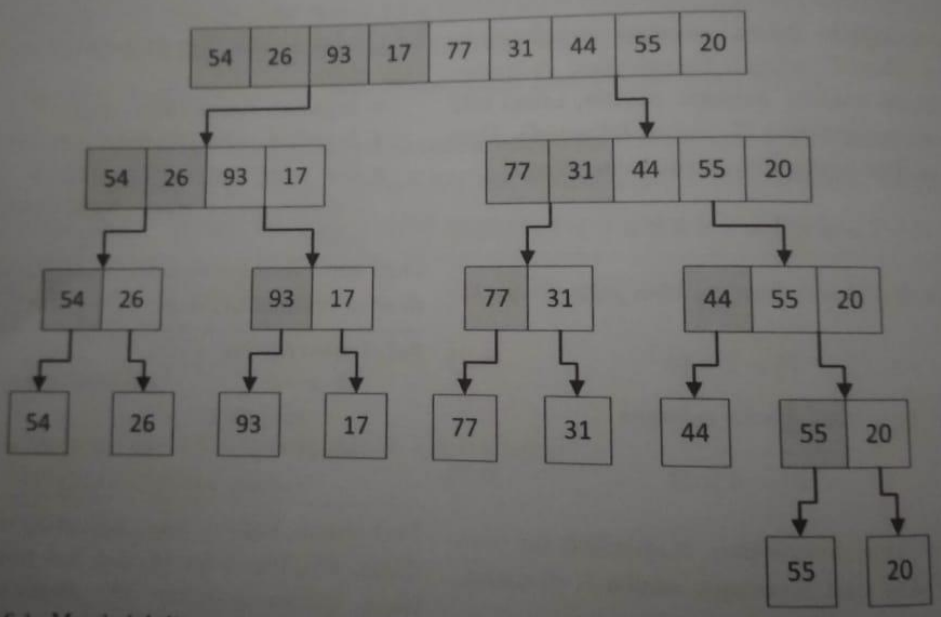
2. Proses MergeSort

```
def mergeSort(A):
    #print("Membelah      ", A)
    if len(A) > 1:
        mid = len(A) // 2          # Membelah list.
        separuhKiri = A[:mid]      # Slicing ini langkah yang expensive sebenarnya,
        separuhKanan = A[mid:]     #   bisakah kamu membuatnya lebih baik?

        mergeSort(separuhKiri)  # Ini rekursi. Memanggil lebih lanjut mergeSort
        mergeSort(separuhKanan) #   untuk separuhKiri dan separuhKanan.

        # Di bawah ini adalah proses penggabungan.
        i=0  ;  j=0  ;  k=0
        while i < len(separuhKiri) and j < len(separuhKanan):
            if separuhKiri[i] < separuhKanan[j]:  # while-loop ini
                A[k] = separuhKiri[i]             #   menggabungkan kedua list, yakni
                i = i + 1                         #   separuhKiri dan separuhKanan,
            else:                                 #   sampai salah satu kosong.
                A[k] = separuhKanan[j]            # Perhatikan kesamaan strukturnya
                j = j + 1                         #   dengan proses penggabungan
            k=k+1                                 #   dua list urut.

        while i < len(separuhKiri):   # Jika separuhKiri mempunyai sisa
            A[k] = separuhKiri[i]     #   tumpukkan ke A
            i = i + 1                 #   satu demi satu.
            k = k + 1                 #

        while j < len(separuhKanan):  # Jika separuhKanan mempunyai sisa
            A[k] = separuhKanan[j]    #   tumpukkan ke A
            j = j + 1                 #   satu demi satu.
            k = k + 1
    #print("Menggabungkan", A)
```

Larikan program di atas dengan memanggilnya seperti ini

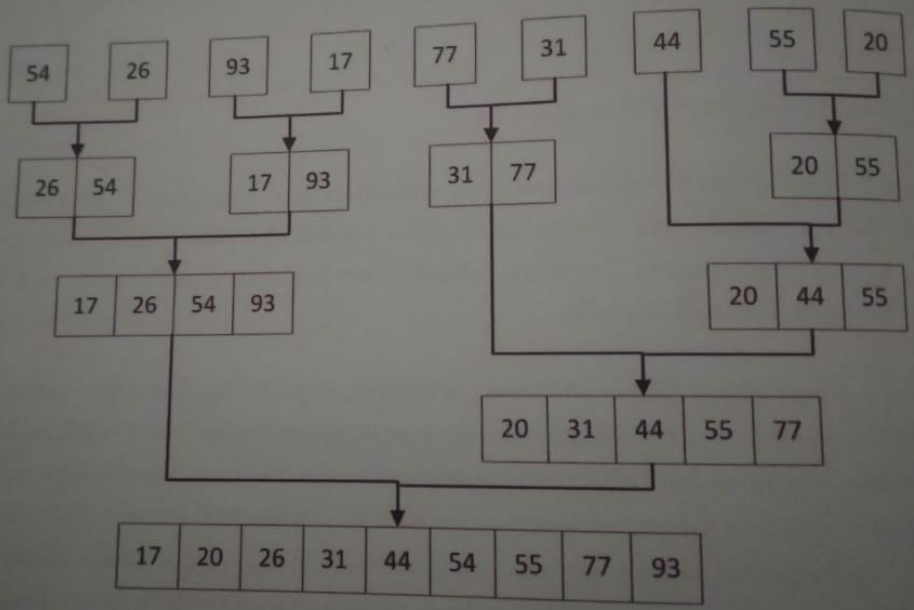
```
alist = [54,26,93,17,77,31,44,55,20]
mergeSort(alist)
print(alist)
```

6.1



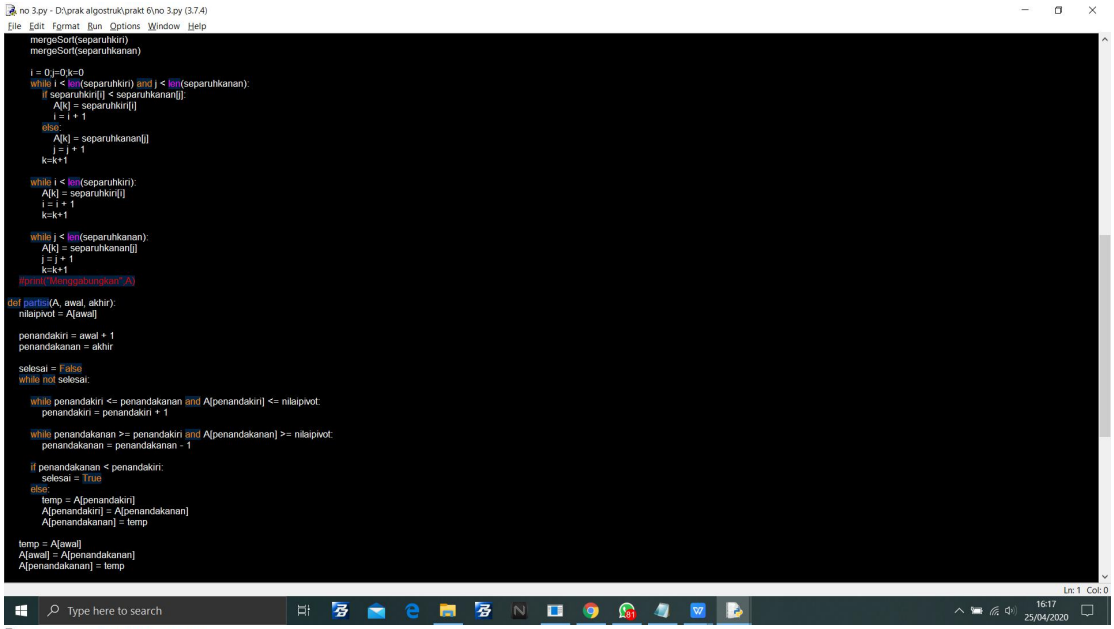
Gambar 6.1: Membelah list sampai tiap sub-list berisi satu elemen atau kosong. Sesudah itu digabu
seperti ditunjukkan di Gambar 6.2.

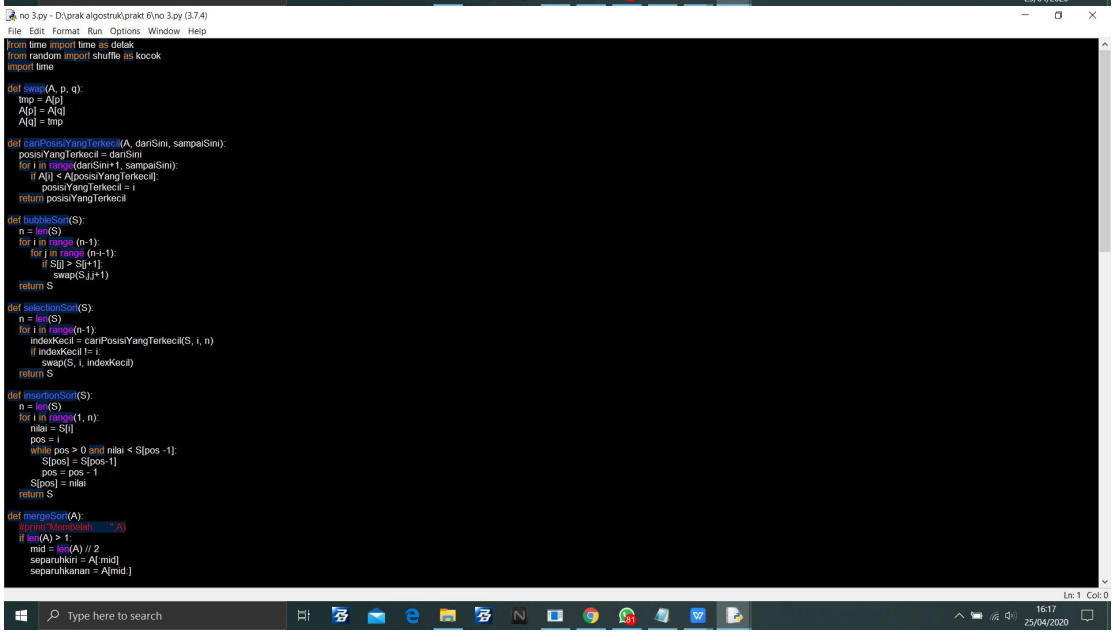
6.2

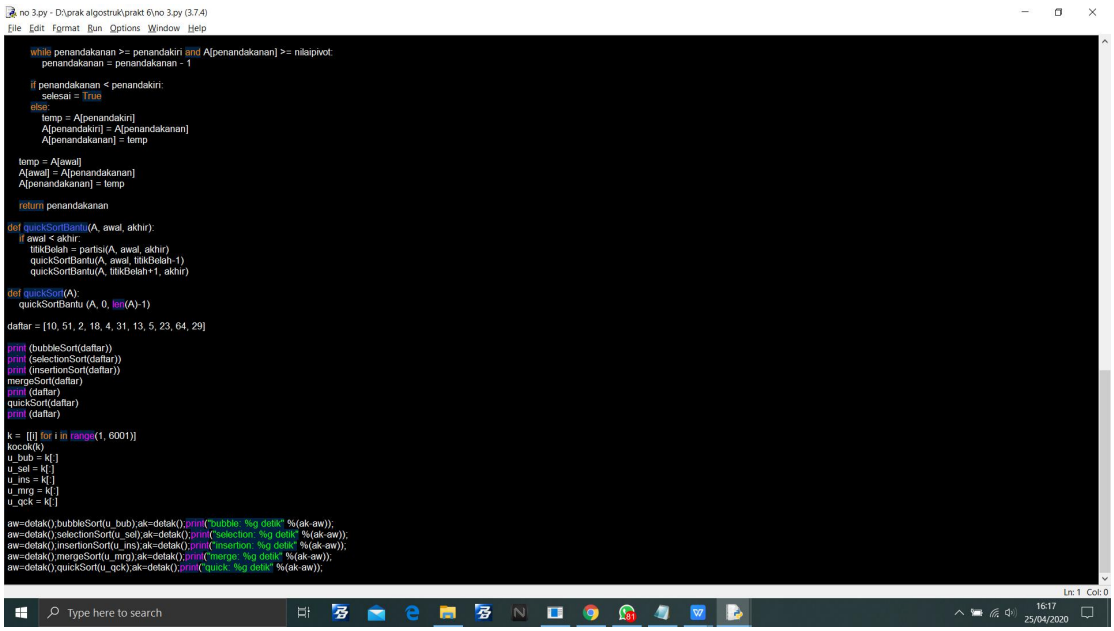


Gambar 6.2: Menggabungkan list satu demi satu.

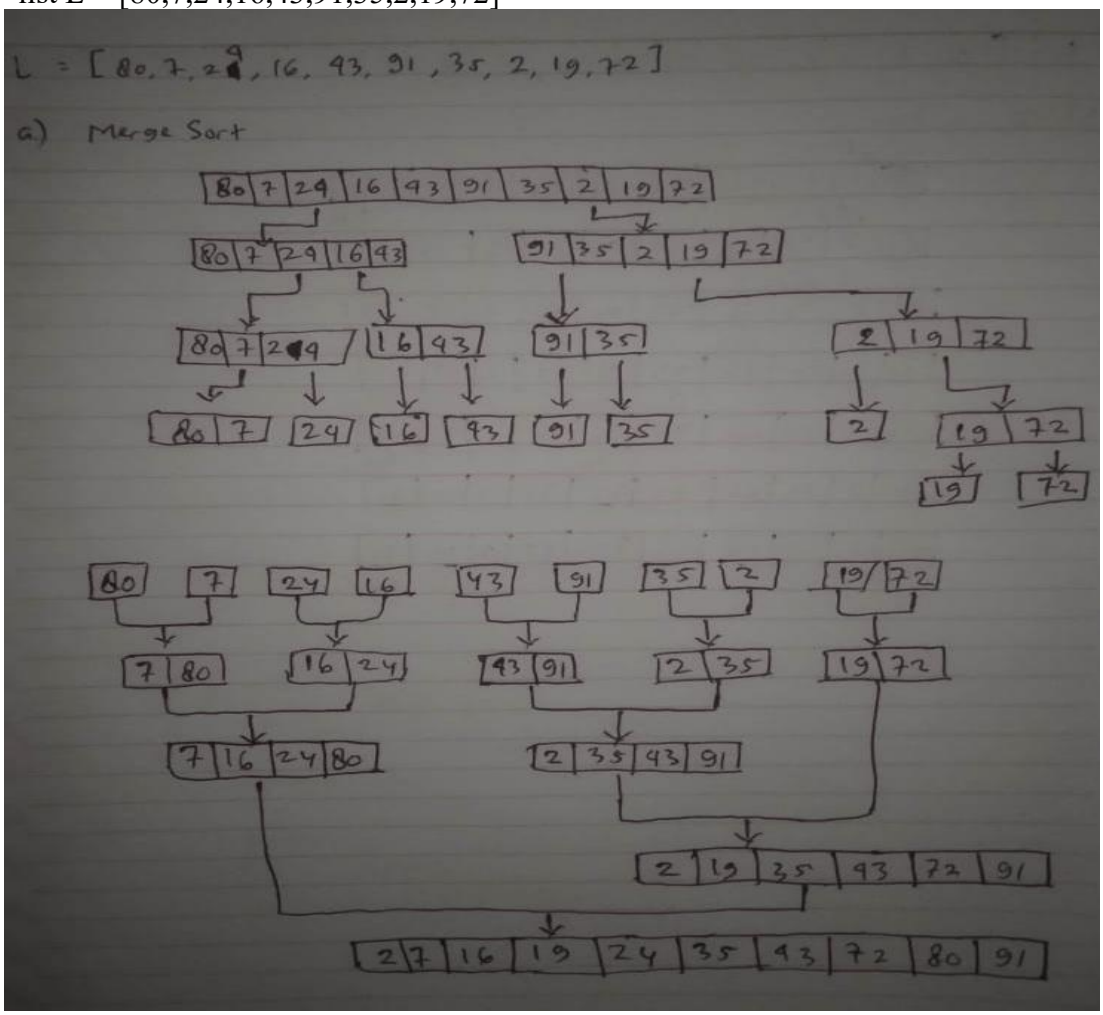
3. Uji kecepatan MergeSort dan QuickSort

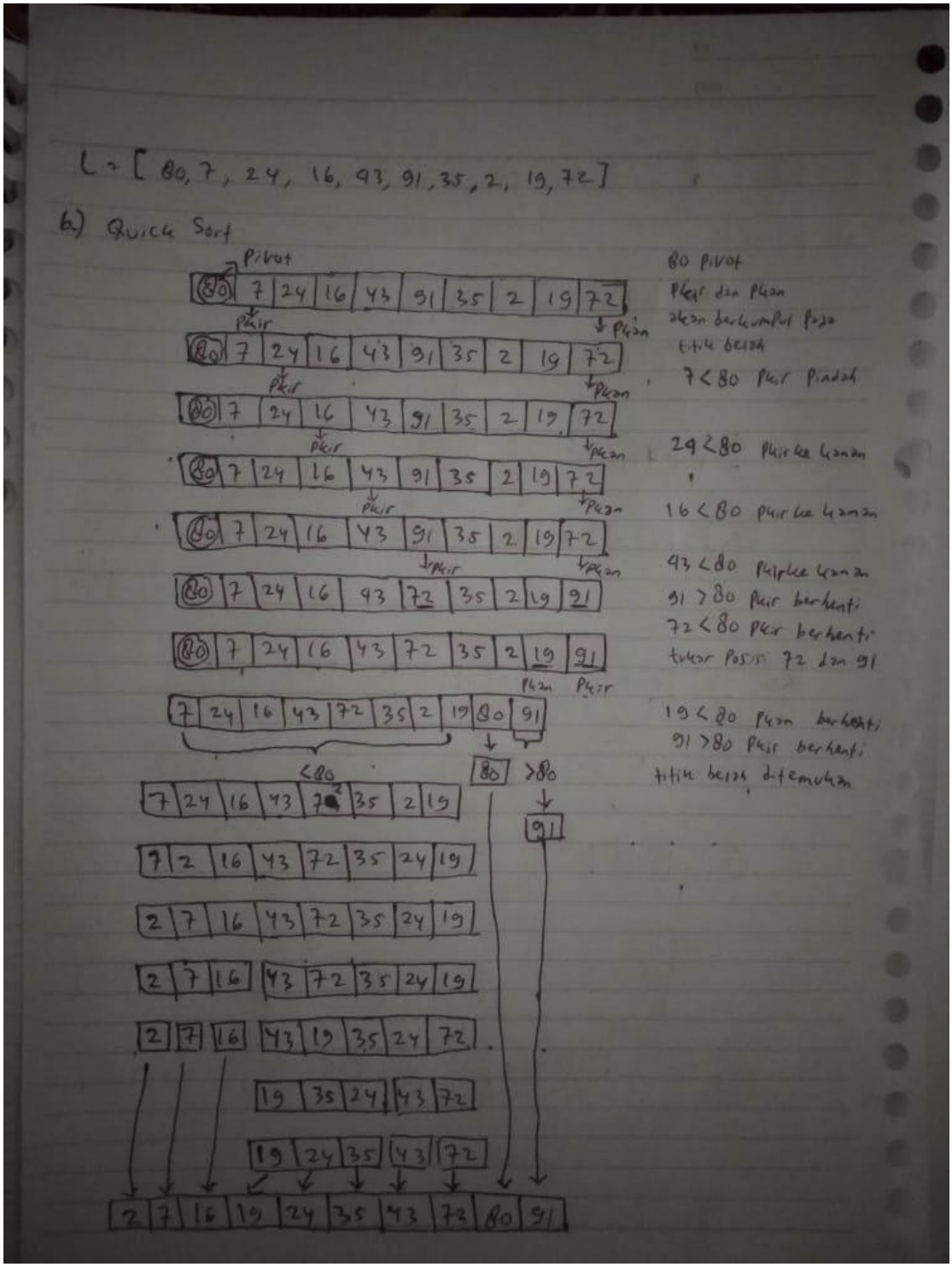




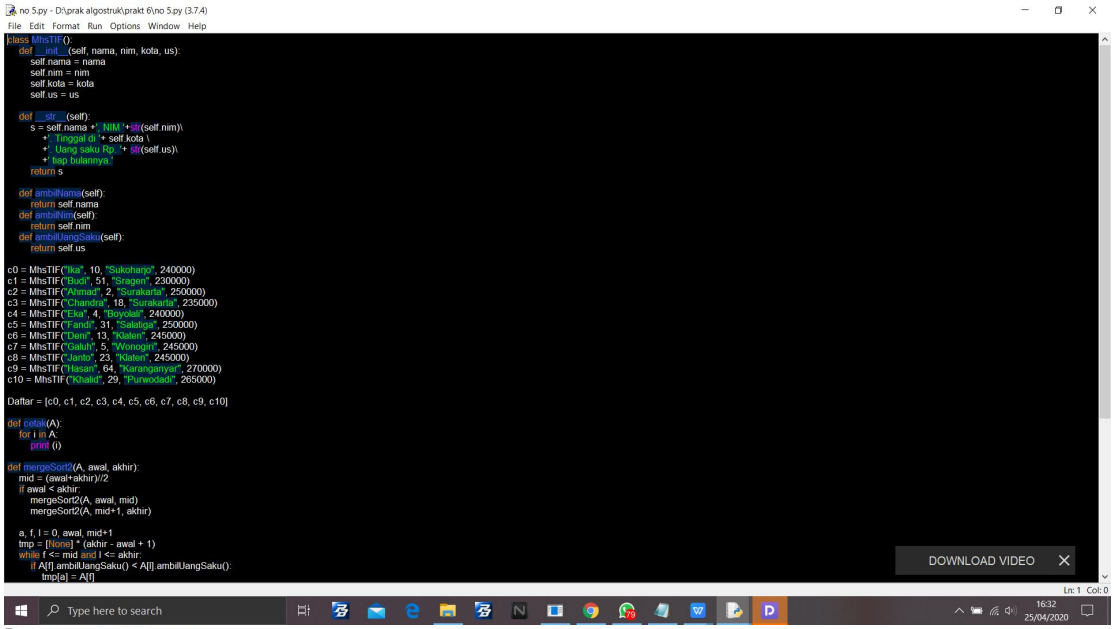


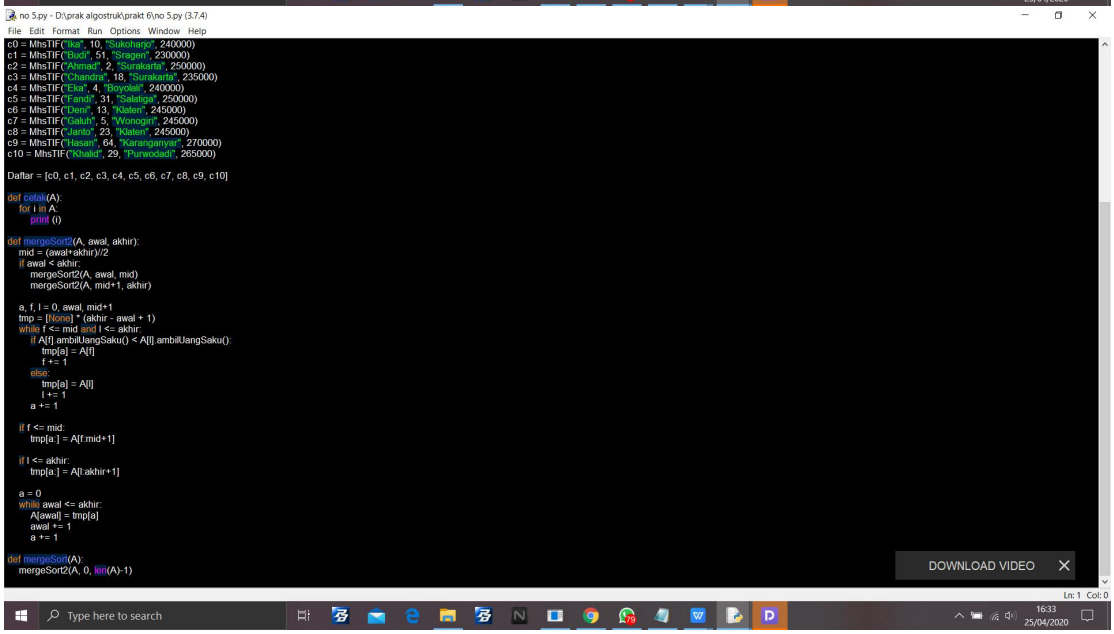
4. Gambar trace pengurutan dengan merge sort dan quick sort pada list L = [80,7,24,16,43,91,35,2,19,72]



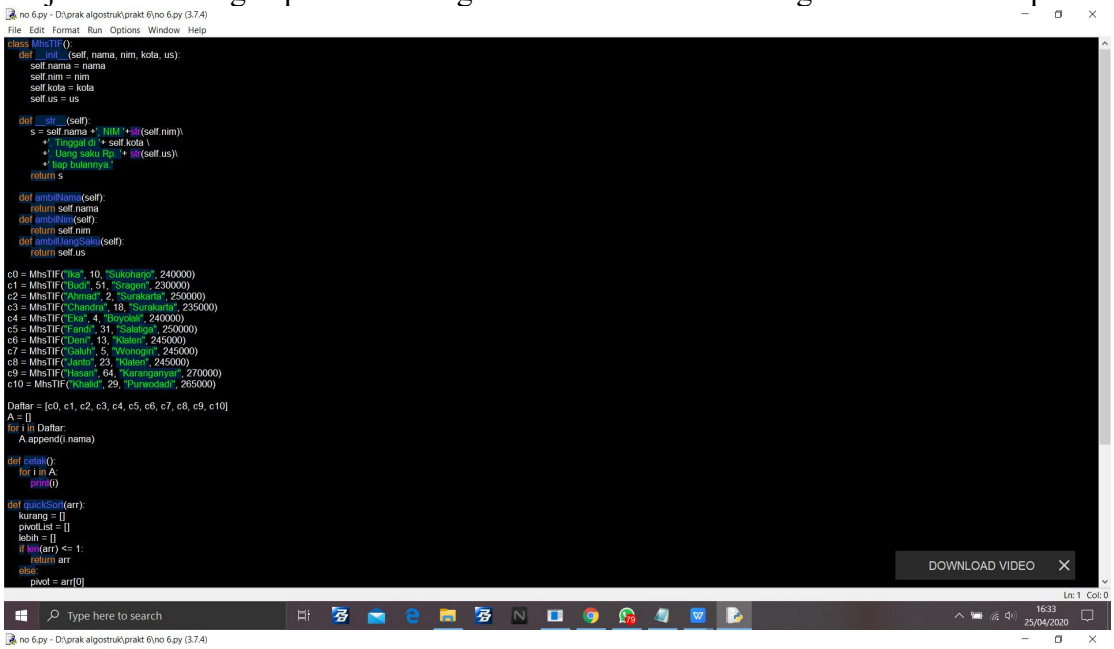


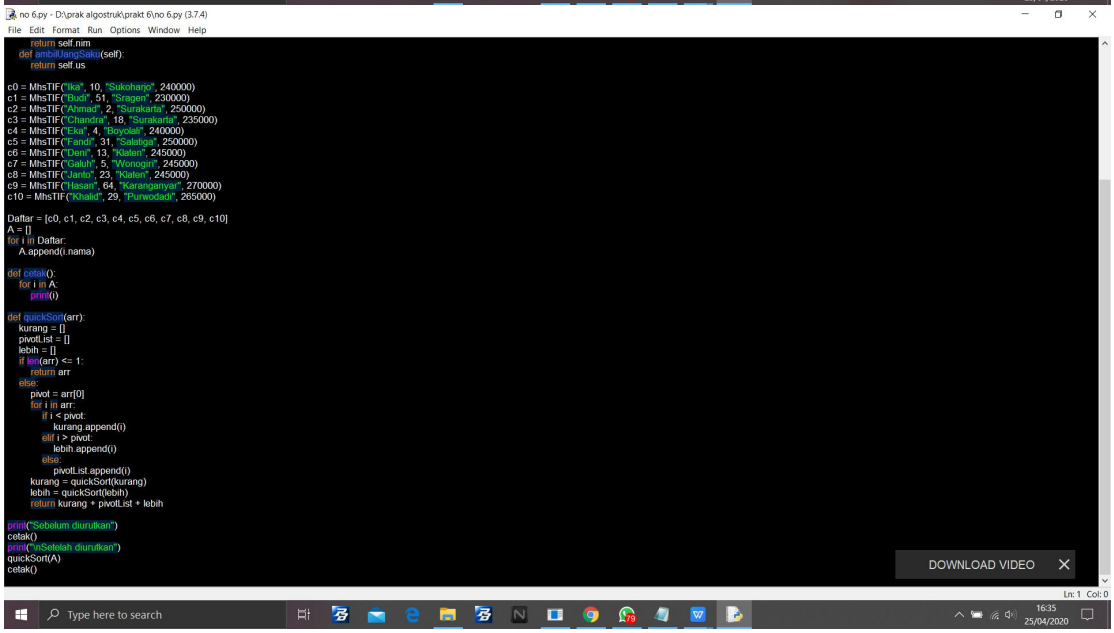
5. Mengefisienkan fungsi MergeSort


no 5.py - D:\prak algostruk\prakt 6\no 5.py (3.7.4)
File Edit Format Run Options Window Help
class MhsTIF():
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 \
+' Uang saku Rp.'+ str(self.us)\
+' tiap bulannya.'
return s
def ambilNama(self):
return self.nama
def ambilNim(self):
return self.nim
def ambilUangSaku(self):
return self.us
c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 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]
DOWNLOAD VIDEO
Ln: 1 Col: 0
Type here to search
16:32
25/04/2020

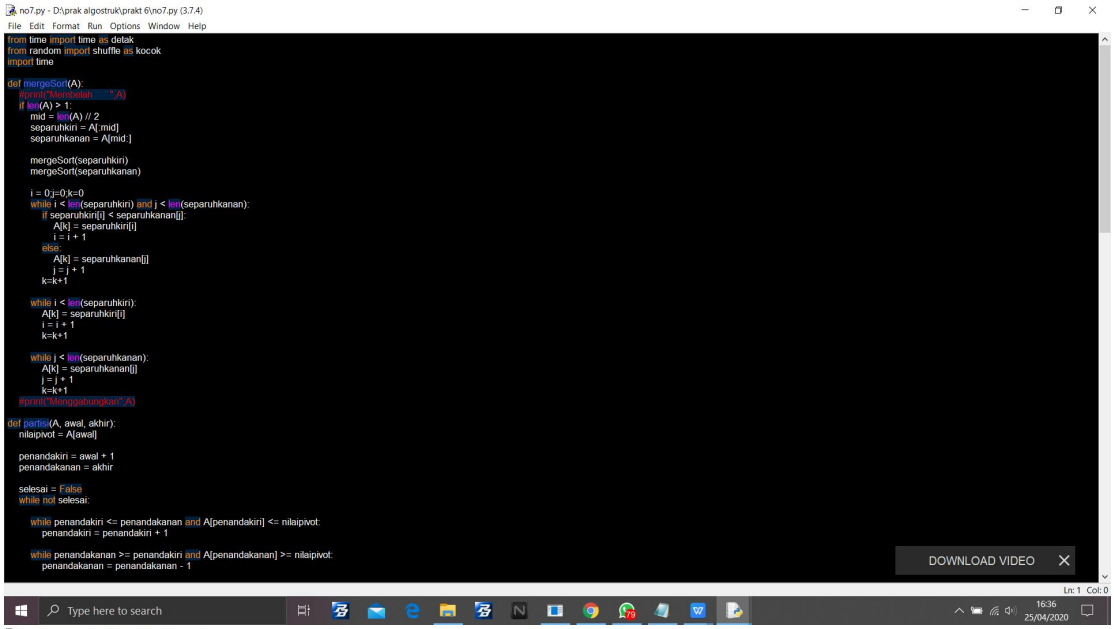

no 5.py - D:\prak algostruk\prakt 6\no 5.py (3.7.4)
File Edit Format Run Options Window Help
c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 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]
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)
DOWNLOAD VIDEO
Ln: 1 Col: 0
Type here to search
16:33
25/04/2020

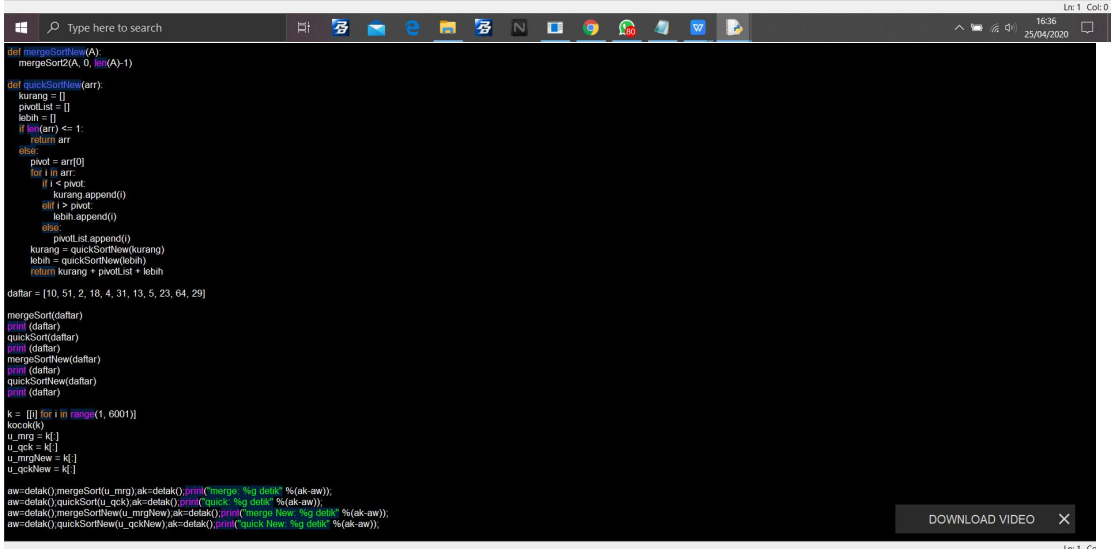
6. Uji efisiensi fungsi quick sort dengan metode median-dari-tiga untuk memilih pivot



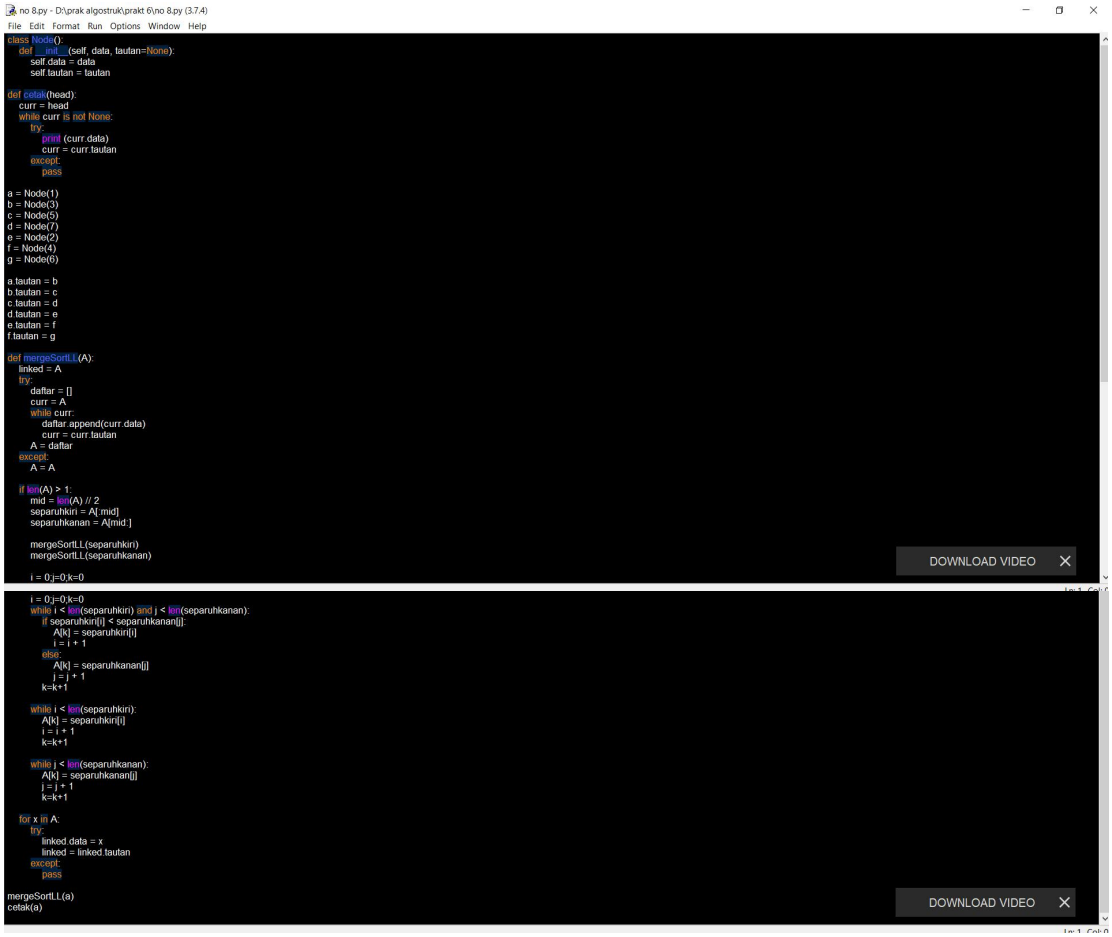


7. Menguji kecepatan sebelum efisiensi dan setelah diefesiensi kan





8. Versi merge sort untuk linked list


no 8.py - D:\prak algostruk\prakt 6\no 8.py (3.7.4)
File Edit Format Run Options Window Help
class Node():
def __init__(self, data, tautan=None):
self.data = data
self.tautan = tautan
def cetak(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
DOWNLOAD VIDEO

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)
DOWNLOAD VIDEO