Nama : Annas Fagiat

Kelas : D

Nim : L200170163

LAPORAN PRAKTIKUM

ALGORTIMA DAN STRUKTUR DATA

MODUL KE 6

Nomor 1

nomor1.py - C:\Users\USER\Documents\6_D_163\nomor1.py (3.6.3)

File Edit Format Run Options Window Help

```
from tugas2s import mahasiswa
from latihan import urut

mh1 = mahasiswa("ANNAS FAGIAT", 163, "BYL", 10000)
mh2 = mahasiswa("PUR", 153, "WNG", 13000)
mh3 = mahasiswa("WAN", 142, "SLTG", 5000)
mh4 = mahasiswa("TO", 175, "SLO", 12000)
mh5 = mahasiswa("RO", 180, "SKH", 2000)

nimMH = [mh1.NIM, mh2.NIM, mh3.NIM, mh4.NIM, mh5.NIM]
usMH = [mh1.us, mh2.us, mh3.us, mh4.us, mh5.us]

a1 = urut(nimMH)
b2 = urut(usMH)

a1.printMerge(nimMH)
b2.printMerge(usMH)

a1.printQuick(nimMH)
b2.printQuick(usMH)
```

Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
============ RESTART: C:\Users\USER\Documents\6_D_163\nomor1.py ============
ini merge sort
142 153 163 175 180

ini merge sort
2000 5000 10000 12000 13000

ini quick sort
142 153 163 175 180

ini quick sort
2000 5000 10000 12000 13000

>>>
```

Nomor 2

-

Nomor 3

nomor3.py - C:\Users\USER\Documents\6_D_163\nomor3.py (3.6.3)

File Edit Format Run Options Window Help

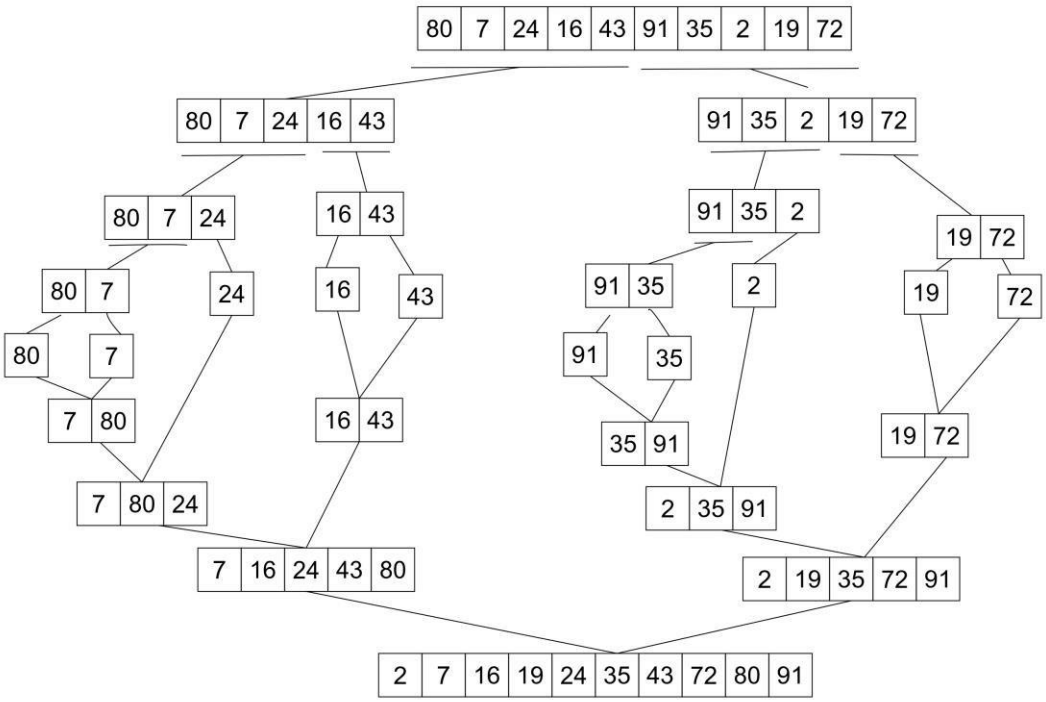
```
from time import time as detak
from random import shuffle as kocok
import time
k = [i for i in range(1,6001)]
kocok(k)

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

def sele(A):
    for i in range(len(A)):
        min_idx = i
        for j in range(i+1, len(A)):
            if A[min_idx] > A[j]:
                min_idx = j
        A[i], A[min_idx] = A[min_idx], A[i]

def inse(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i-1
        while j >=0 and key < arr[j] :
                arr[j+1] = arr[j]
                j -= 1
        arr[j+1] = key
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]:
```

Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
============ RESTART: C:\Users\USER\Documents\6_D_163\nomor3.py ============
bubble : 4.28377 detik
selection : 2.0518 detik
insertion : 2.05578 detik
merge : 0.0312176 detik
quick : 0.0312424 detik
>>>
```

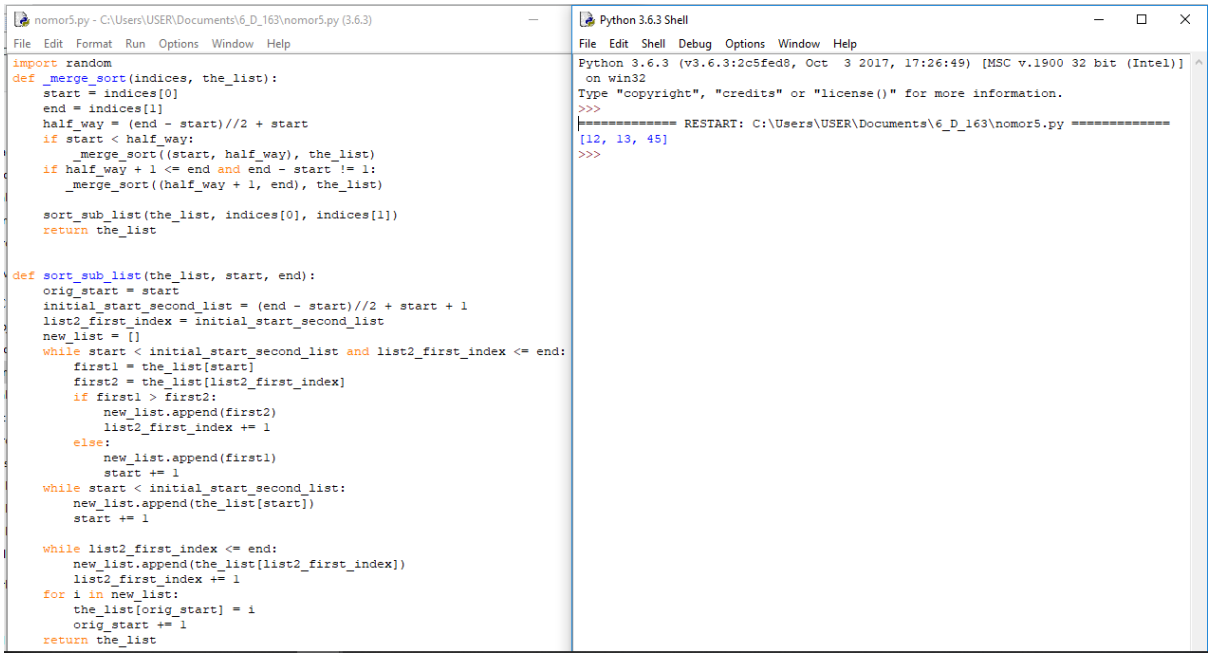
Nomor 4 A (Tracing Algorithm Merge Sort)



Nomor 4 B (Tracing Algorithm Quick Sort)

-

Nomor 5 (Merge Sort tanpa Slicing, menggunakan recursive)



nomor5.py - C:\Users\USER\Documents\6_D_163\nomor5.py (3.6.3)

File Edit Format Run Options Window Help

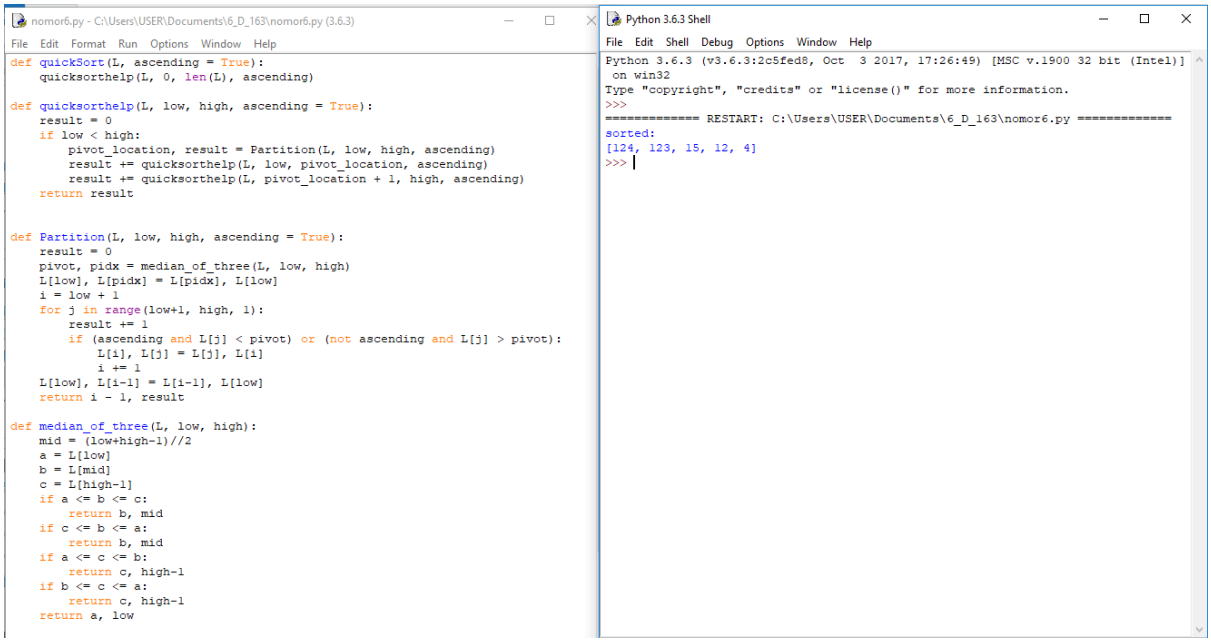
```
import random
def _merge_sort(indices, the_list):
    start = indices[0]
    end = indices[1]
    half_way = (end - start)//2 + start
    if start < half_way:
        _merge_sort((start, half_way), the_list)
    if half_way + 1 <= end and end - start != 1:
        _merge_sort((half_way + 1, end), 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
```

Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\USER\Documents\6_D_163\nomor5.py ==========
[12, 13, 45]
>>>
```

Nomor 6 (Quick Sort dengan Median of Three)



nomor6.py - C:\Users\USER\Documents\6_D_163\nomor6.py (3.6.3)

File Edit Format Run Options Window Help

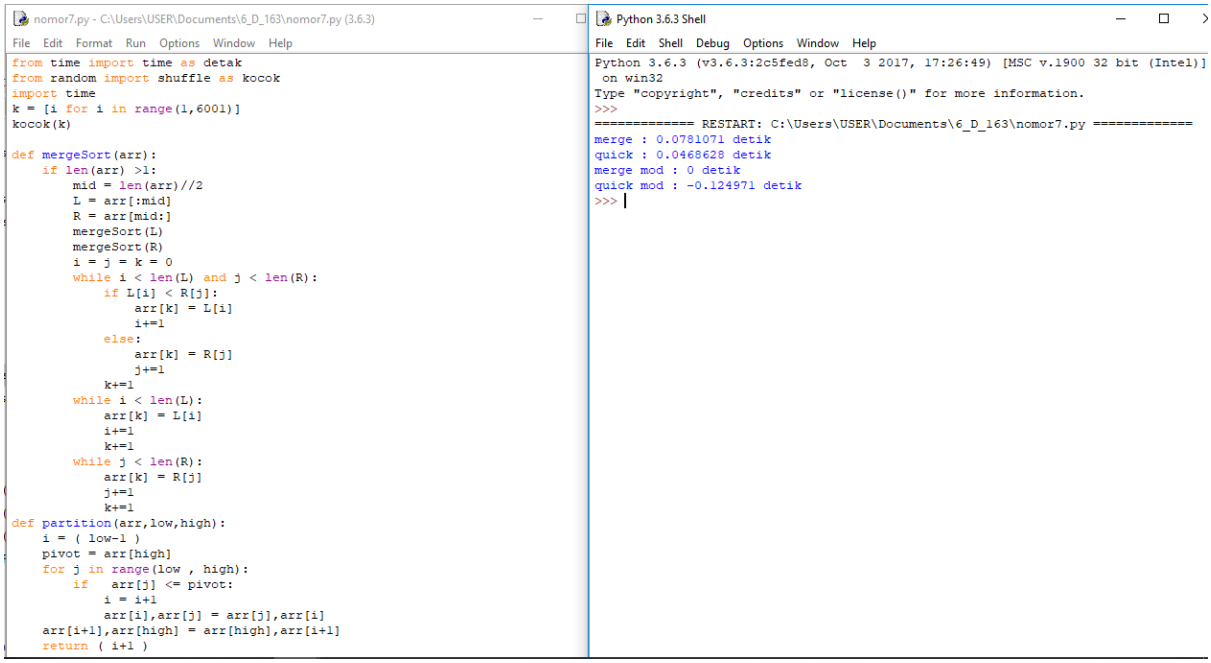
```
def quickSort(L, ascending = True):
    quicksorthelp(L, 0, len(L), ascending)

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
```

Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\USER\Documents\6_D_163\nomor6.py ==========
sorted:
[124, 123, 15, 12, 4]
>>>
```

Nomor 7



nomor7.py - C:\Users\USER\Documents\6_D_163\nomor7.py (3.6.3)

File Edit Format Run Options Window Help

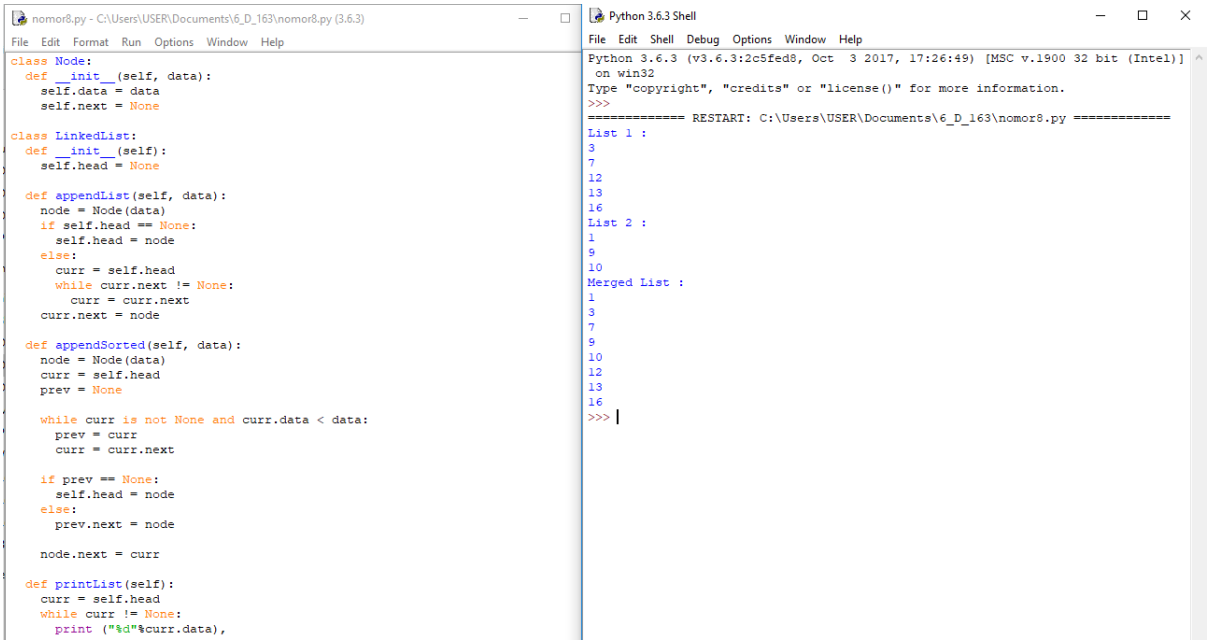
```
from time import time as detak
from random import shuffle as kocok
import time
k = [i for i in range(1,6001)]
kocok(k)

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]
    return ( i+1 )
```

Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\USER\Documents\6_D_163\nomor7.py ==========
merge : 0.0781071 detik
quick : 0.0468628 detik
merge mod : 0 detik
quick mod : -0.124971 detik
>>>
```

Nomor 8 (Merge Sort dengan Linked List)



nomor8.py - C:\Users\USER\Documents\6_D_163\nomor8.py (3.6.3)

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
    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),
```

Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
============ RESTART: C:\Users\USER\Documents\6_D_163\nomor8.py ============
List 1 :
3
7
12
13
16
List 2 :
1
9
10
Merged List :
1
3
7
9
10
12
13
16
>>>
```