**LAPORAN PRAKTIKUM ALGORITMA STRUKTUR DATA**

**MODUL 3**

**“COLLECTIONS, ARRAYS, AND LINKED**

**STRUCTURES”**



**Oleh:**

**NAMA : Daffa Putra Alwansyah**

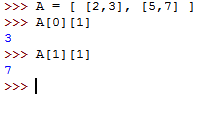
**NIM : L200190031**

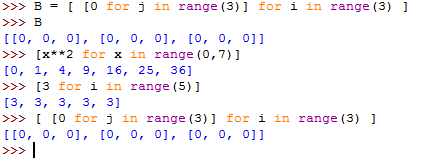
**KELAS : B**

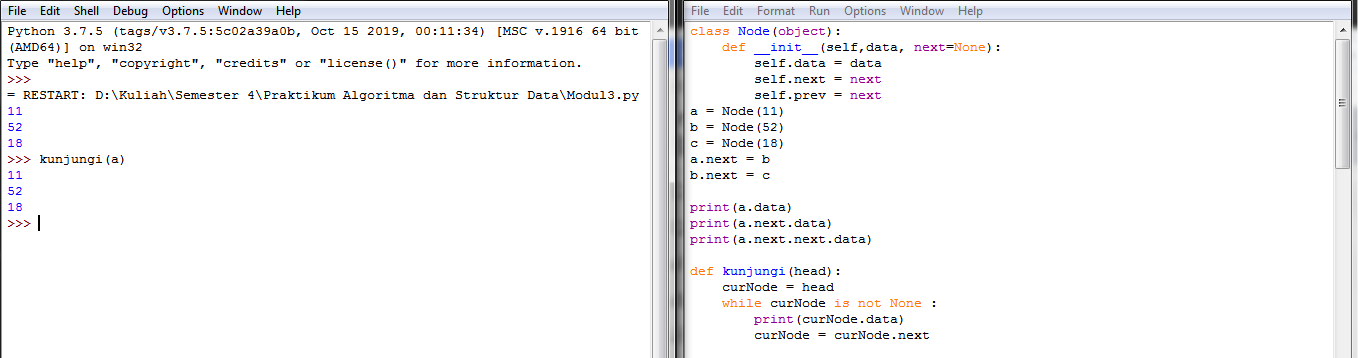
**PRODI : INFORMATIKA**

**Fakultas Komunikasi dan Informatika Universitas Muhammadiyah Surakarta**

**Latihan**

**Latihan3.1  
**

**Latihan3.2  
**

**Latihan3.3**  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**3.4 Soal-soal untuk Mahasiswa**  
A = [[1,2],[3,4],[5,'3']]

B = [[9,4],[2,1]]

C = [[8,5],[1,3]]

**#Nomor 1A**

class Matriks (object):

def cetakMatriks(self, matriks):

for i in matriks:

print(i)

def cekKonsisten(self, matriks):

if len(matriks[0]) == len(matriks) :

return ("Matriks konsisten, ordo sama")

else:

return ("Matriks tidak konsisten, ordo berbeda ")

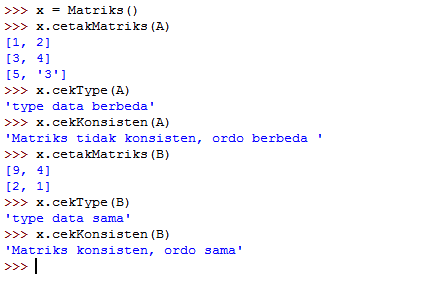
def cekType(self, matriks):

for i in matriks:

for x in i:

if type(x) != int:

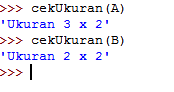
return("type data berbeda")

return("type data sama")  


**#Nomor 1B**

def cekUkuran(matriks):

return ("Ukuran "+str(len(matriks))+" x "+str(len(matriks[0])))



**#Nomor 1C**

def Jumlah(m1, m2):

if cekUkuran(m1) == cekUkuran(m2):

for x in range(0, len(m1)):

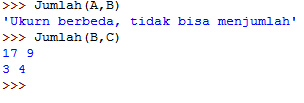
for y in range(0, len(m1[0])):

print (m1[x][y] + m2[x][y],end=' '),

print()

else:

return("Ukurn berbeda, tidak bisa menjumlah")



**##Nomor 1D**

i =[]

def Perkalian(m1,m2):

if cekUkuran(m1) == cekUkuran(m2):

for x in range(0, len(m1)):

row = []

for y in range (0, len(m1[0])):

total = 0

for z in range (0, len(m1)):

total = total + (m1[x][y]\*m2[z][y])

row.append(total)

i.append(row)

for x in range (0, len(i)):

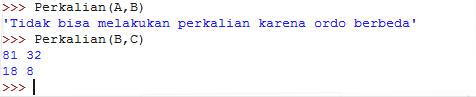
for y in range(0, len(i[0])):

print (i[x][y], end=' '),

print()

else:

return("Tidak bisa melakukan perkalian karena ordo berbeda")



**##Nomor 1E**

def Determinan(x):

for i in range(2):

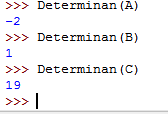
if i == 0:

ad = x[i][i]\*x[i+1][i+1]

elif i == 1:

bc = x[i-1][i]\*x[i][i-1]

return ad-bc



**##Nomor 2A**

def buatNol(n, m=None):

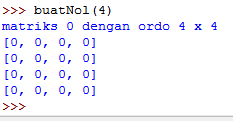
if (m == None):

m = n

print ("matriks 0 dengan ordo "+str(n)+" x "+str(m))

x = ([[0 for j in range(m)] for i in range(n)])

for i in x:

print(i)  


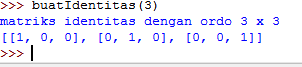
**##Nomor 2B**

def buatIdentitas(m):

print("matriks identitas dengan ordo "+str(m)+" x "+str(m))

matriks = [[1 if j == i else 0 for j in range(m)] for i in range(m)]

print(matriks)



**###Nomor 3**

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

class LinkedList:

def \_\_init\_\_(self):

self.head = None

**#menammbah suatu simpul di awal**

def tambahDepan(self, new\_data):

new\_node = Node(new\_data)

new\_node.next = self.head

self.head = new\_node

**#menambah suatu simpul di akhir**

def tambahAkhir(self, data):

if (self.head == None):

self.head = Node(data)

else:

current = self.head

while (current.next != None):

current = current.next

current.next = Node(data)

return self.head

**#menyisipkan suatu simpul di mana saja**

def tambah(self,data,posisi):

node = Node(data)

if not self.head:

self.head = node

elif posisi == 0:

node.next = self.head

self.head = node

else:

prev = None

current = self.head

current\_posisi = 0

while (current\_posisi < posisi) and current.next:

prev = current

current = current.next

current\_posisi += 1

prev.next = node

node.next = current

return self.head

**#menghapus suatu simpul di awal, di akhir, atau di mana saja**

def hapus(self,posisi):

if self.head == None:

return

temp = self.head

if posisi == 0:

self.head = temp.next

temp = None

return

for i in range(posisi - 1):

temp = temp.next

if temp is None:

break

if temp is None:

return

if temp.next is None:

return

next = temp.next.next

temp.next = None

temp.next = next

**##mencari data yang isinya tertentu**

def cari(self,x):

current = self.head

while current != None:

if current.data == x:

return True

current = current.next

return False

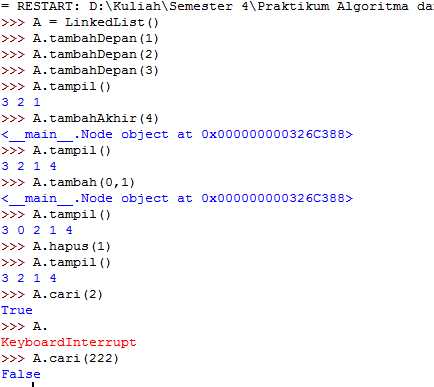
def tampil(self):

current = self.head

while current is not None:

print(current.data, end = ' ')

current = current.next



**#Nomor 4**

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.prev = None

class DoublyLinkedList:

def \_\_init\_\_(self):

self.head = None

**#menambah suatu simpul di awal**

def awal(self, new\_data):

print("Menambah awal ",new\_data)

new\_node = Node(new\_data)

new\_node.next = self.head

if self.head is not None:

self.head.prev = new\_node

self.head = new\_node

**#menambah suatu simpul di akhir**

def akhir(self,new\_data):

print("Menambah akhir ",new\_data)

new\_node = Node(new\_data)

new\_node.next = None

if self.head is None:

new\_node.prev = None

self.head = new\_node

return

last = self.head

while(last.next is not None):

last = last.next

last.next = new\_node

new\_node.prev = last

return

**#mengunjungi dan mencetak data tiap simpul dari depan dan dari belakang**

def tampil(self,node):

print("\ntampilan depan :")

while (node is not None):

print (" %d "%(node.data))

last = node

node = node.next

print ("\ntampilan dbelakang :")

while (last is not None):

print (" %d "%(last.data))

last = last.prev

