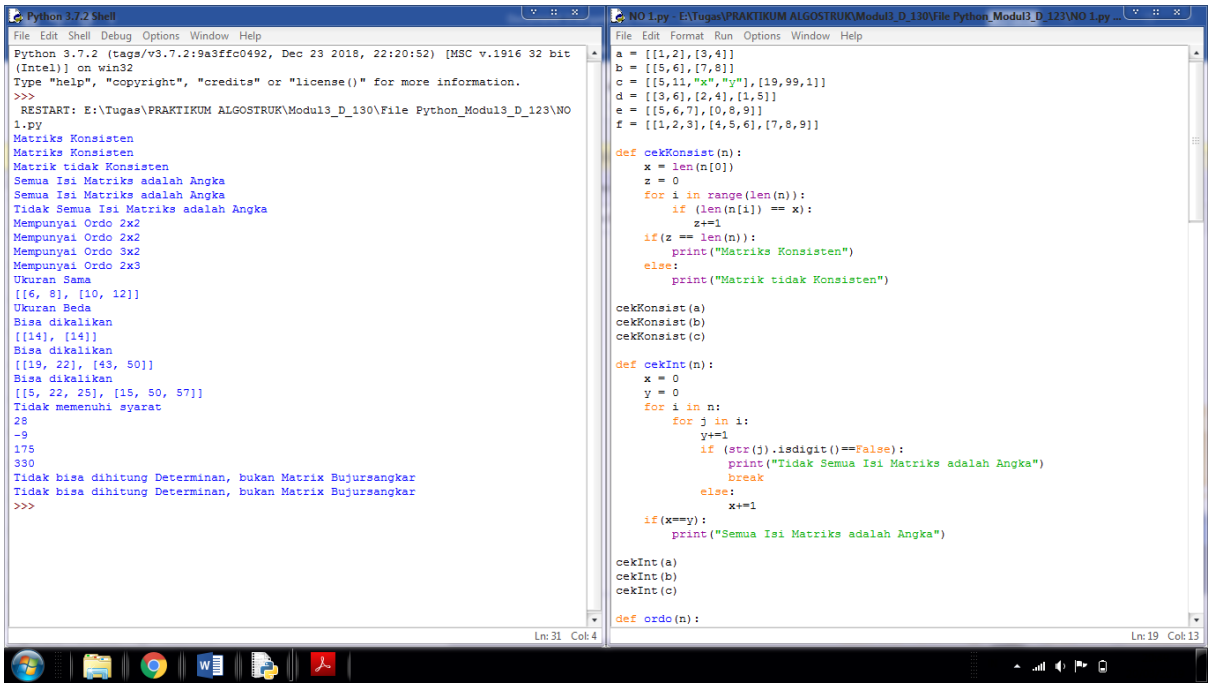
Nama : Puji Nugroho
NIM : L200170123
Kelas : D
Modul : 3

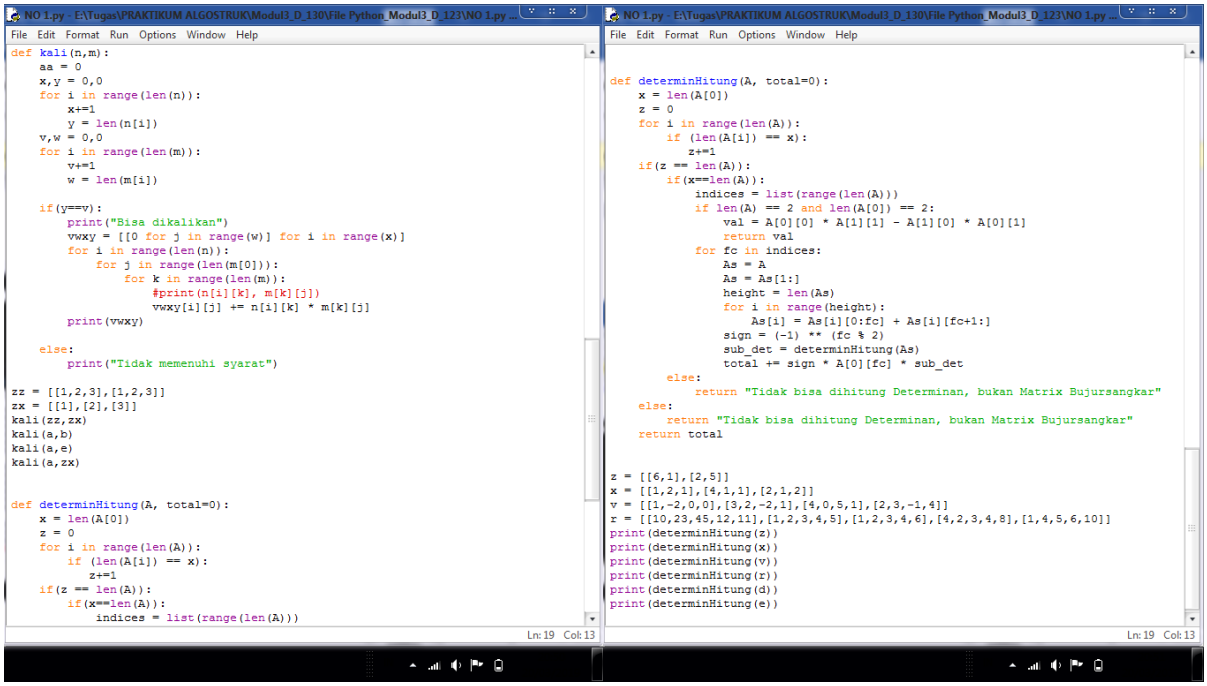
Nomor 1.



```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO
1.py
Matriks Konsisten
Matriks Konsisten
Matrik tidak Konsisten
Semua Isi Matriks adalah Angka
Semua Isi Matriks adalah Angka
Tidak Semua Isi Matriks adalah Angka
Mempunyai Ordo 2x2
Mempunyai Ordo 2x2
Mempunyai Ordo 3x2
Mempunyai Ordo 2x3
Ukuran Sama
[[6, 8], [10, 12]]
Ukuran Beda
Bisa dikalikan
[[14], [14]]
Bisa dikalikan
[[19, 22], [43, 50]]
Bisa dikalikan
[[5, 22, 25], [15, 50, 57]]
Tidak memenuhi syarat
28
-9
175
330
Tidak bisa dihitung Determinan, bukan Matrix Bujursangkar
Tidak bisa dihitung Determinan, bukan Matrix Bujursangkar
>>>
```

```
a = [[1,2],[3,4]]
b = [[5,6],[7,8]]
c = [[5,11,"x","y"],[19,99,1]]
d = [[3,6],[2,4],[1,5]]
e = [[5,6,7],[0,8,9]]
f = [[1,2,3],[4,5,6],[7,8,9]]

def cekKonsist(n):
    x = len(n[0])
    z = 0
    for i in range(len(n)):
        if (len(n[i]) == x):
            z+=1
    if(z == len(n)):
        print("Matriks Konsisten")
    else:
        print("Matrik tidak Konsisten")

cekKonsist(a)
cekKonsist(b)
cekKonsist(c)

def cekInt(n):
    x = 0
    y = 0
    for i in n:
        for j in i:
            y+=1
            if (str(j).isdigit()==False):
                print("Tidak Semua Isi Matriks adalah Angka")
                break
            else:
                x+=1
    if(x==y):
        print("Semua Isi Matriks adalah Angka")

cekInt(a)
cekInt(b)
cekInt(c)

def ordo(n):
```

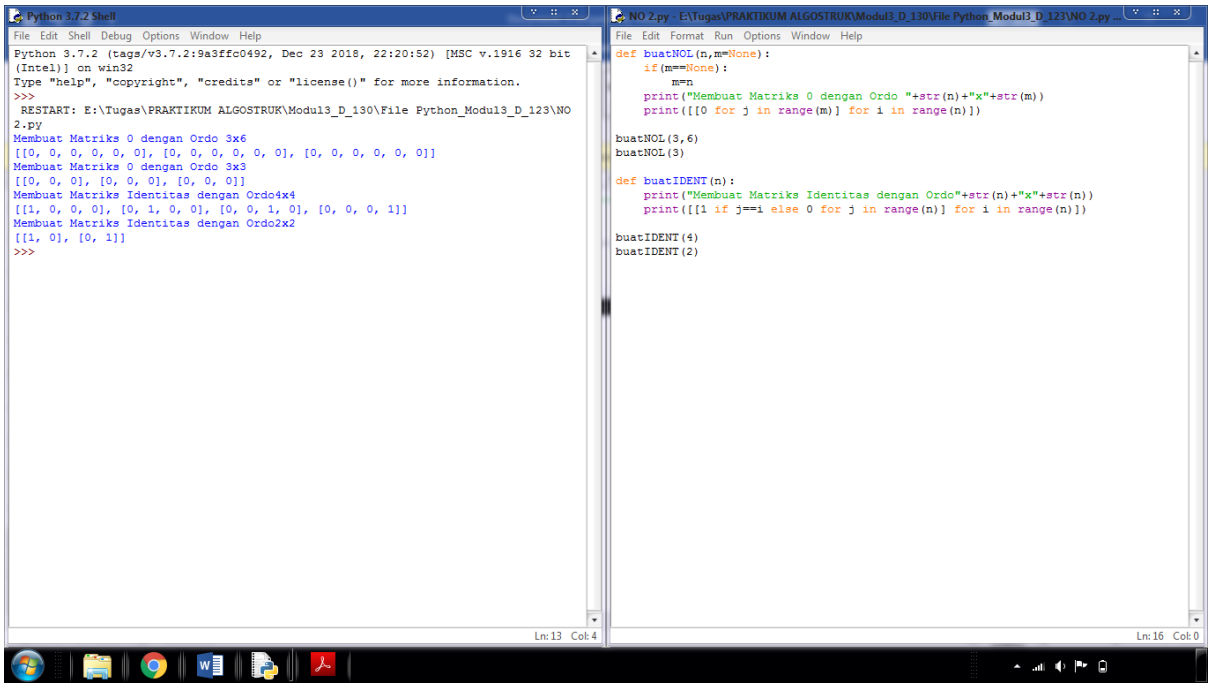
```
def ordo(n):
    x,y = 0,0
    for i in range(len(n)):
        x+=1
        y = len(n[i])
    print("Mempunyai Ordo "+str(x)+"x"+str(y))

ordo(a)
ordo(b)
ordo(d)
ordo(e)

def jumlah(n,m):
    x,y = 0,0
    for i in range(len(n)):
        x+=1
        y = len(n[i])
    xy = [[0 for j in range(x)] for i in range(y)]

    z = 0
    if(len(n)==len(m)):
        for i in range(len(n)):
            if(len(n[i]) == len(m[i])):
                z+=1
    if(z==len(n) and z==len(m)):
        print("Ukuran Sama")
        for i in range(len(n)):
            for j in range(len(n[i])):
                xy[i][j] = n[i][j] + m[i][j]
        print(xy)
    else:
        print("Ukuran Beda")

jumlah(a,b)
jumlah(a,d)


def kali(n,m):
    aa = 0
    x,y = 0,0
    for i in range(len(n)):
```



```
def kali(n,m):
    aa = 0
    x,y = 0,0
    for i in range(len(n)):
        x+=1
        y = len(n[i])
    v,w = 0,0
    for i in range(len(m)):
        v+=1
        w = len(m[i])

    if(y==v):
        print("Bisa dikalikan")
        vwxy = [[0 for j in range(w)] for i in range(x)]
        for i in range(len(n)):
            for j in range(len(m[0])):
                for k in range(len(m)):
                    #print(n[i][k], m[k][j])
                    vwxy[i][j] += n[i][k] * m[k][j]
        print(vwxy)

    else:
        print("Tidak memenuhi syarat")

zz = [[1,2,3],[1,2,3]]
zx = [[1],[2],[3]]
kali(zz,zx)
kali(a,b)
kali(a,e)
kali(a,zx)


def determinHitung(A, total=0):
    x = len(A[0])
    z = 0
    for i in range(len(A)):
        if (len(A[i]) == x):
            z+=1
    if(z == len(A)):
        if(x==len(A)):
            indices = list(range(len(A)))
```

```
def determinHitung(A, total=0):
    x = len(A[0])
    z = 0
    for i in range(len(A)):
        if (len(A[i]) == x):
            z+=1
    if(z == len(A)):
        if(x==len(A)):
            indices = list(range(len(A)))
            if len(A) == 2 and len(A[0]) == 2:
                val = A[0][0] * A[1][1] - A[1][0] * A[0][1]
                return val
            for fc in indices:
                As = A
                As = As[1:]
                height = len(As)
                for i in range(height):
                    As[i] = As[i][0:fc] + As[i][fc+1:]
                sign = (-1) ** (fc % 2)
                sub_det = determinHitung(As)
                total += sign * A[0][fc] * sub_det
        else:
            return "Tidak bisa dihitung Determinan, bukan Matrix Bujursangkar"
    else:
        return "Tidak bisa dihitung Determinan, bukan Matrix Bujursangkar"
    return total


z = [[6,1],[2,5]]
x = [[1,2,1],[4,1,1],[2,1,2]]
v = [[1,-2,0,0],[3,2,-2,1],[4,0,5,1],[2,3,-1,4]]
r = [[10,23,45,12,11],[1,2,3,4,5],[1,2,3,4,6],[4,2,3,4,8],[1,4,5,6,10]]
print(determinHitung(z))
print(determinHitung(x))
print(determinHitung(v))
print(determinHitung(r))
print(determinHitung(d))
print(determinHitung(e))
```

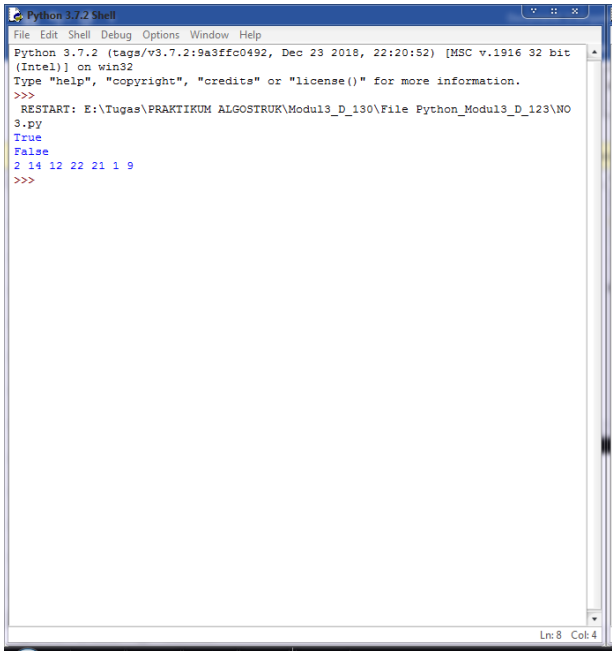
Nomor 2.

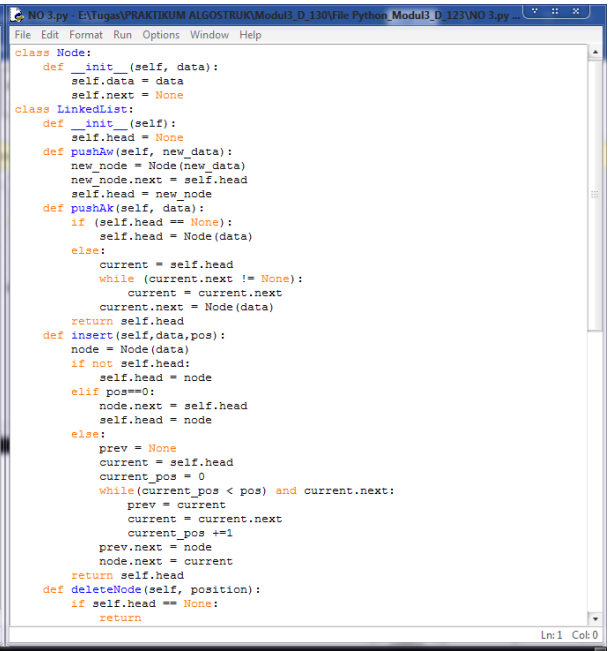


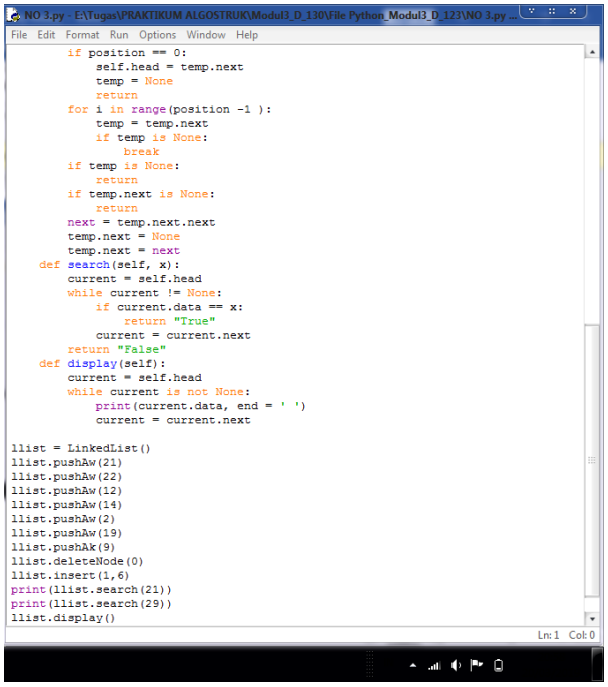
```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO
2.py
Membuat Matriks 0 dengan Ordo 3x6
[[0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0]]
Membuat Matriks 0 dengan Ordo 3x3
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]
Membuat Matriks Identitas dengan Ordo4x4
[[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]
Membuat Matriks Identitas dengan Ordo2x2
[[1, 0], [0, 1]]
>>>
```

```
def buatNOL(n,m=None):
    if(m==None):
        m=n
    print("Membuat Matriks 0 dengan Ordo "+str(n)+"x"+str(m))
    print([[0 for j in range(m)] for i in range(n)])

buatNOL(3,6)
buatNOL(3)

def buatIDENT(n):
    print("Membuat Matriks Identitas dengan Ordo"+str(n)+"x"+str(n))
    print([[1 if j==i else 0 for j in range(n)] for i in range(n)])

buatIDENT(4)
buatIDENT(2)
```

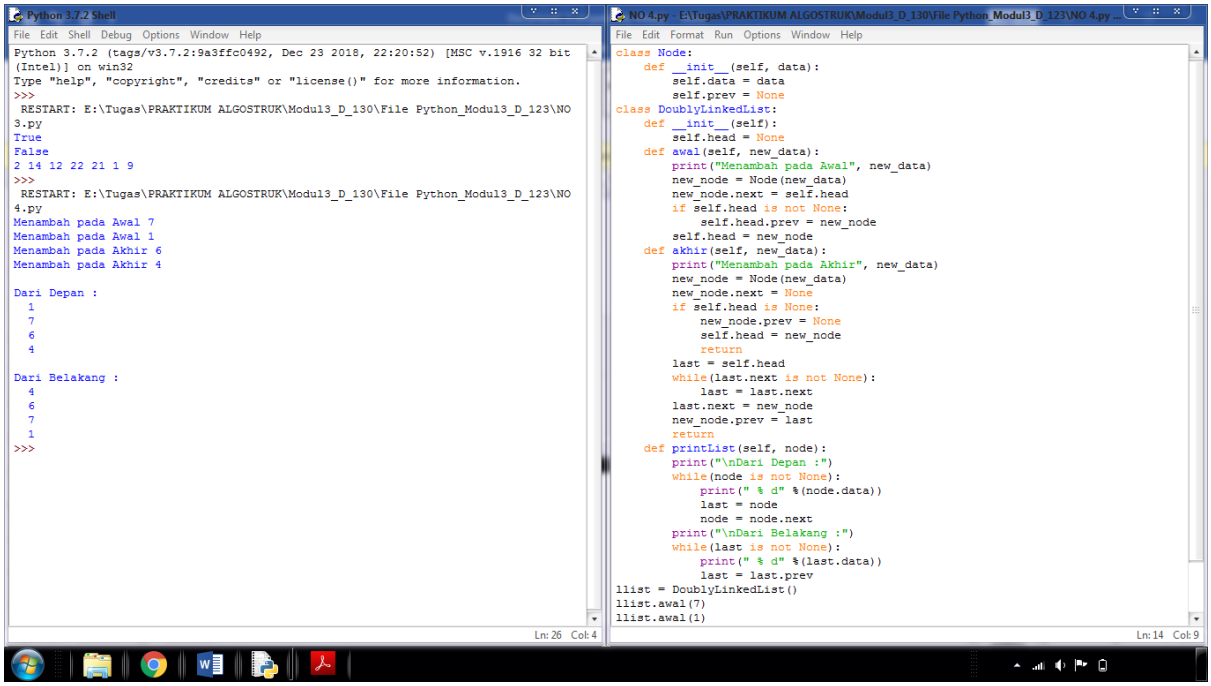
Nomor 3.


Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO 3.py
True
False
2 14 12 22 21 1 9
>>>
Ln: 8 Col: 4


NO 3.py - E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO 3.py
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 pushAw(self, new_data):
new_node = Node(new_data)
new_node.next = self.head
self.head = new_node
def pushAk(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
def insert(self,data,pos):
node = Node(data)
if not self.head:
self.head = node
elif pos==0:
node.next = self.head
self.head = node
else:
prev = None
current = self.head
current_pos = 0
while(current_pos < pos) and current.next:
prev = current
current = current.next
current_pos +=1
prev.next = node
node.next = current
return self.head
def deleteNode(self, position):
if self.head == None:
return
Ln: 1 Col: 0


NO 3.py - E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO 3.py
File Edit Format Run Options Window Help
if position == 0:
self.head = temp.next
temp = None
return
for i in range(position -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
def search(self, x):
current = self.head
while current != None:
if current.data == x:
return "True"
current = current.next
return "False"
def display(self):
current = self.head
while current is not None:
print(current.data, end = ' ')
current = current.next
llist = LinkedList()
llist.pushAw(21)
llist.pushAw(22)
llist.pushAw(12)
llist.pushAw(14)
llist.pushAw(2)
llist.pushAw(19)
llist.pushAk(9)
llist.deleteNode(0)
llist.insert(1,6)
print(llist.search(21))
print(llist.search(29))
llist.display()
Ln: 1 Col: 0

Nomor 4.


Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO 3.py
True
False
2 14 12 22 21 1 9
>>>
RESTART: E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO 4.py
Menambah pada Awal 7
Menambah pada Awal 1
Menambah pada Akhir 6
Menambah pada Akhir 4
Dari Depan :
1
7
6
4
Dari Belakang :
4
6
7
1
>>>
Ln: 26 Col: 4
NO 4.py - E:\Tugas\PRAKTIKUM ALGOSTRUK\Modul3_D_130\File Python_Modul3_D_123\NO 4.py ...
File Edit Format Run Options Window Help
class Node:
def __init__(self, data):
self.data = data
self.prev = None
class DoublyLinkedList:
def __init__(self):
self.head = None
def awal(self, new_data):
print("Menambah pada 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
def akhir(self, new_data):
print("Menambah pada 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
def printList(self, node):
print("\nDari Depan :")
while(node is not None):
print(" % d" %(node.data))
last = node
node = node.next
print("\nDari Belakang :")
while(last is not None):
print(" % d" %(last.data))
last = last.prev
llist = DoublyLinkedList()
llist.awal(7)
llist.awal(1)
Ln: 14 Col: 9