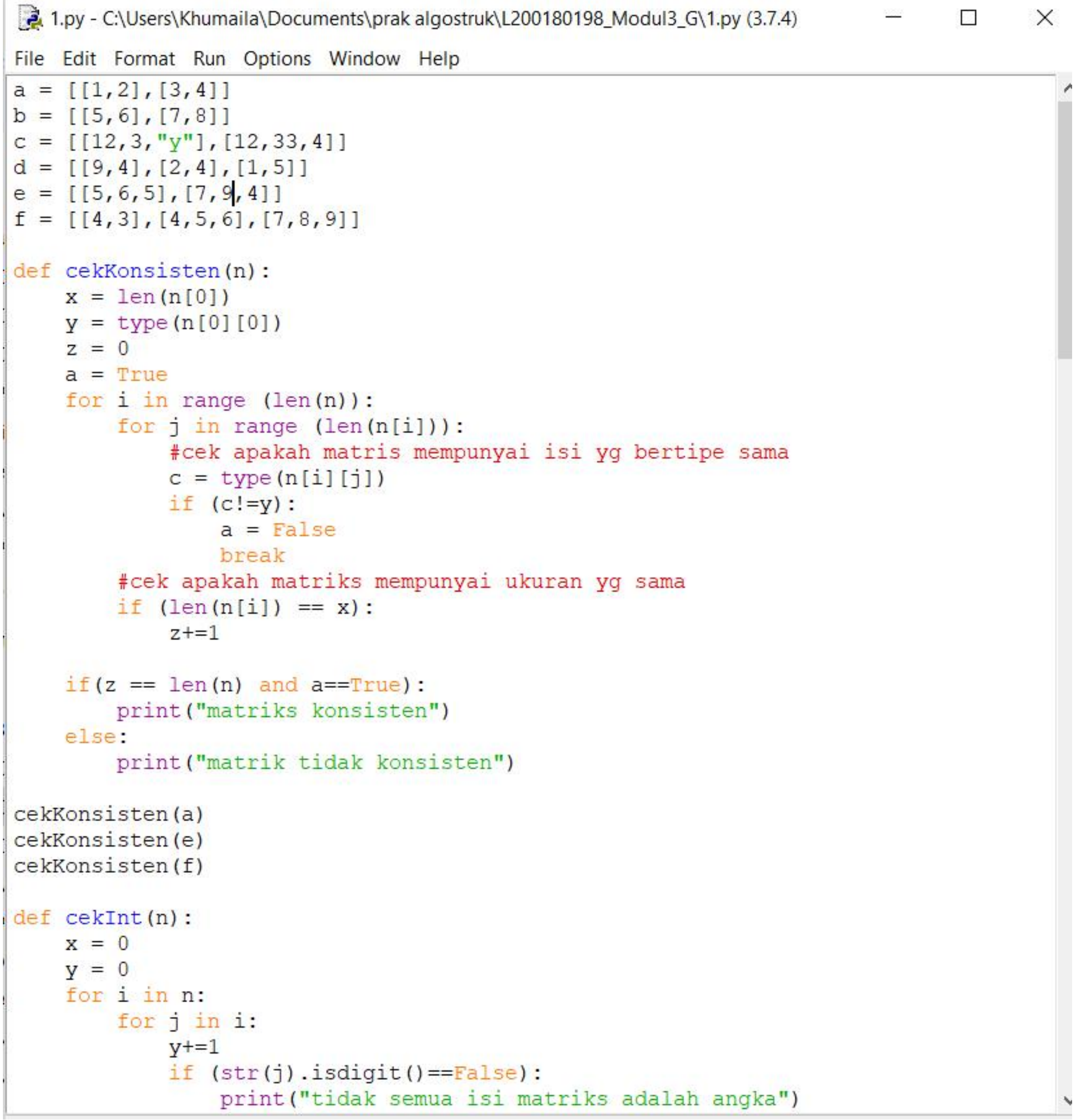
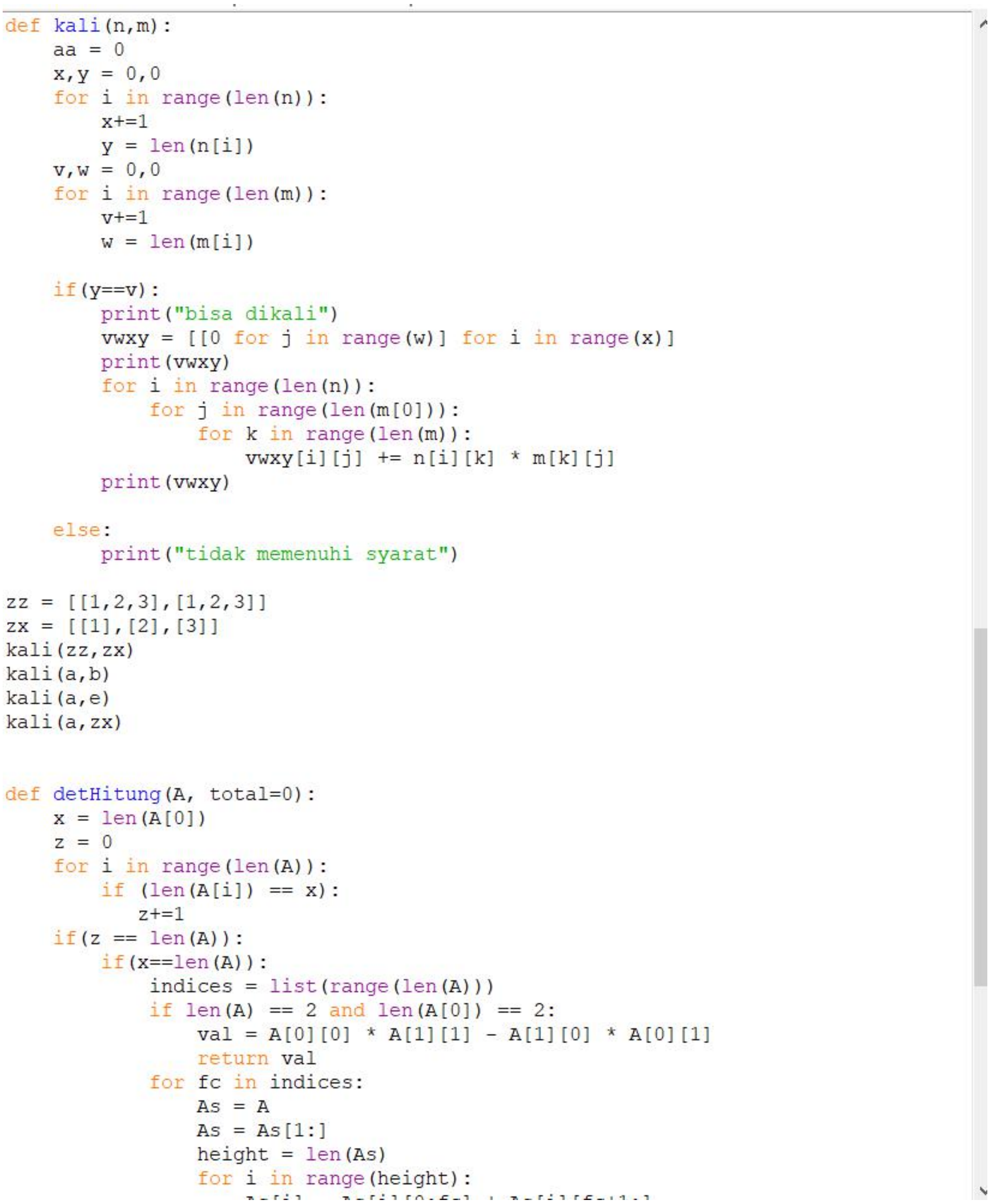
Nama: Khumaila Masfarina Yusrifa
NIM:L200180198/G#
Modul 3

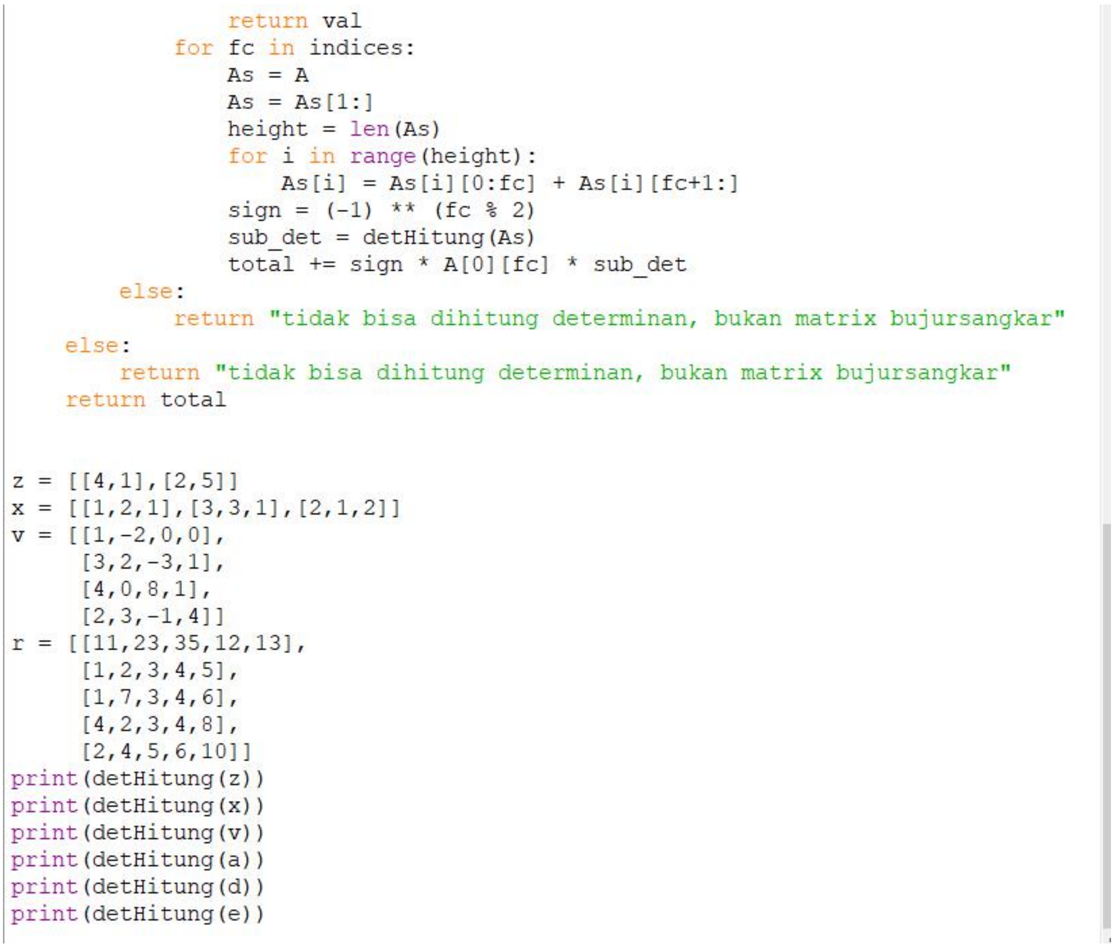
NO 1


1.py - C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\1.py (3.7.4)
File Edit Format Run Options Window Help

```python
a = [[1,2],[3,4]]
b = [[5,6],[7,8]]
c = [[12,3,"y"],[12,33,4]]
d = [[9,4],[2,4],[1,5]]
e = [[5,6,5],[7,9,4]]
f = [[4,3],[4,5,6],[7,8,9]]

def cekKonsisten(n):
    x = len(n[0])
    y = type(n[0][0])
    z = 0
    a = True
    for i in range (len(n)):
        for j in range (len(n[i])):
            #cek apakah matris mempunyai isi yg bertipe sama
            c = type(n[i][j])
            if (c!=y):
                a = False
                break
        #cek apakah matriks mempunyai ukuran yg sama
        if (len(n[i]) == x):
            z+=1

    if(z == len(n) and a==True):
        print("matriks konsisten")
    else:
        print("matrik tidak konsisten")

cekKonsisten(a)
cekKonsisten(e)
cekKonsisten(f)

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")
                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 cekOrdo(n):
    x,y = 0,0
    for i in range(len(n)):
        x+=1
        y = len(n[i])
    print(len(n))
    print("mempunyai ordo "+str(x)+"x"+str(y))

cekOrdo(a)
cekOrdo(c)
cekOrdo(d)
cekOrdo(f)

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(e,b)
jumlah(a,d)
```

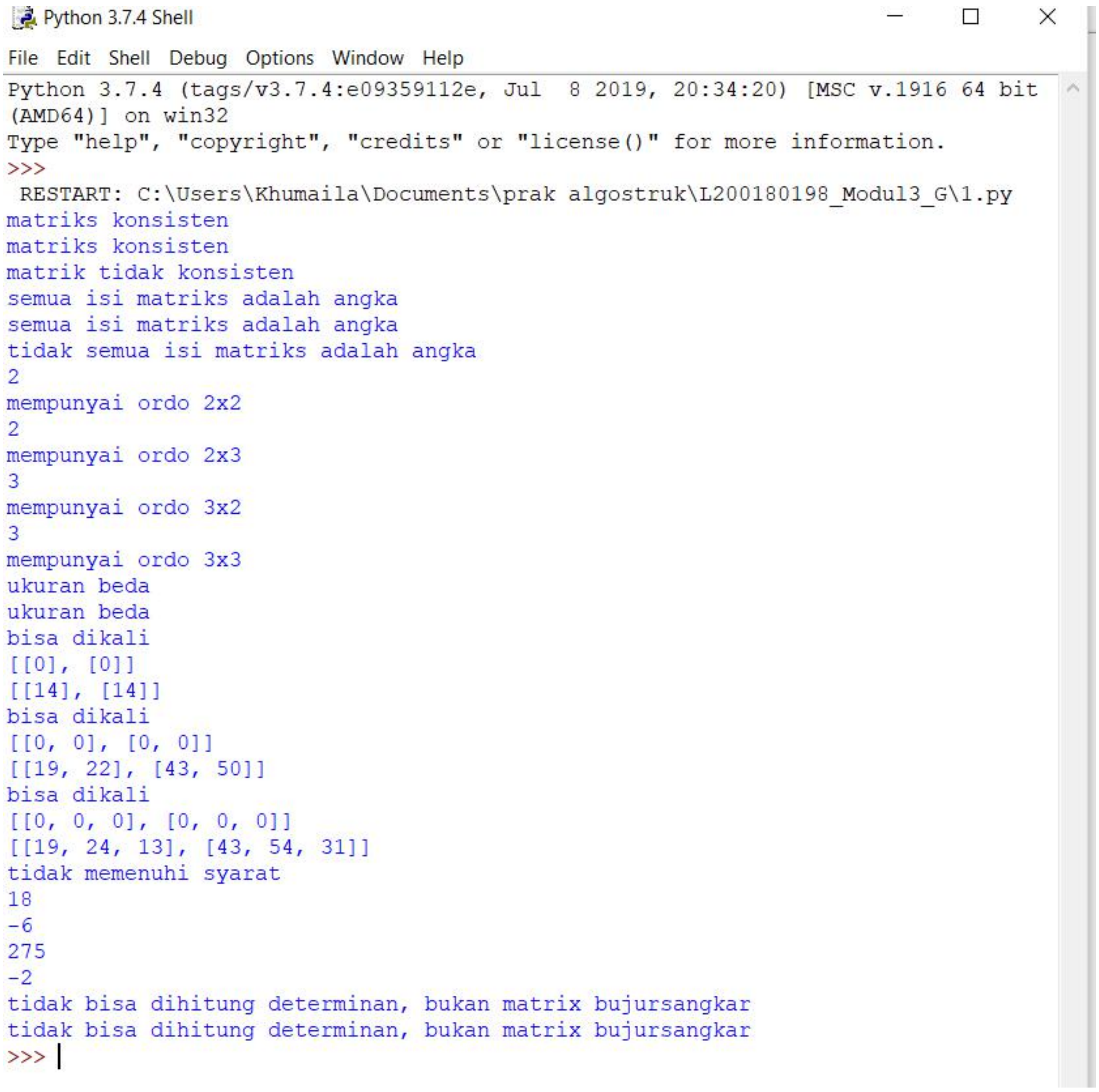


```
def kali(n,m):
    aa = 0
    x,y = 0,0
    for i in range(len(n)):
        x+=1
        y = len(n[i])
    v,w = 0,0
    for i in range(len(m)):
        v+=1
        w = len(m[i])

    if(y==v):
        print("bisa dikali")
        vwxy = [[0 for j in range(w)] for i in range(x)]
        print(vwxy)
        for i in range(len(n)):
            for j in range(len(m[0])):
                for k in range(len(m)):
                    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 detHitung(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:]
```



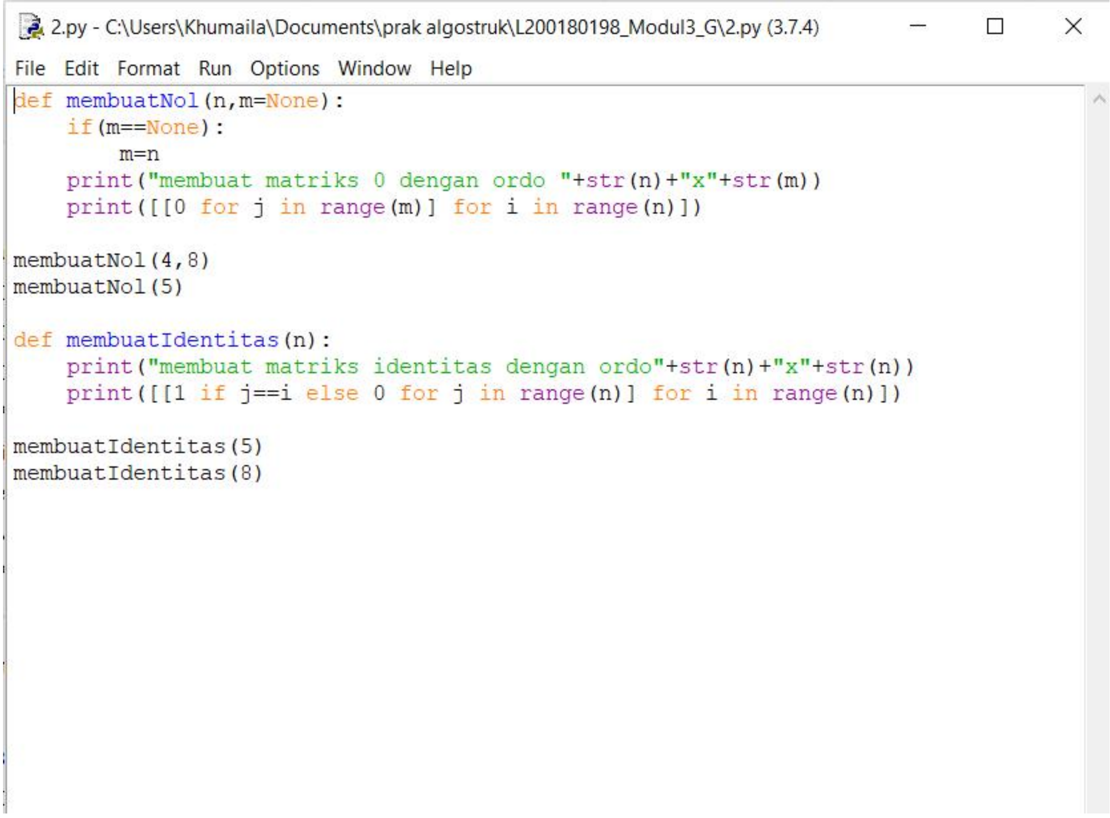
```
                return val
            for fc in indices:
                As = A
                As = As[1:]
                height = len(As)
                for i in range(height):
                    As[i] = As[i][0:fc] + As[i][fc+1:]
                sign = (-1) ** (fc % 2)
                sub_det = detHitung(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 = [[4,1],[2,5]]
x = [[1,2,1],[3,3,1],[2,1,2]]
v = [[1,-2,0,0],
     [3,2,-3,1],
     [4,0,8,1],
     [2,3,-1,4]]
r = [[11,23,35,12,13],
     [1,2,3,4,5],
     [1,7,3,4,6],
     [4,2,3,4,8],
     [2,4,5,6,10]]
print(detHitung(z))
print(detHitung(x))
print(detHitung(v))
print(detHitung(a))
print(detHitung(d))
print(detHitung(e))
```

Hasil:



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\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
2
mempunyai ordo 2x2
2
mempunyai ordo 2x3
3
mempunyai ordo 3x2
3
mempunyai ordo 3x3
ukuran beda
ukuran beda
bisa dikali
[[0], [0]]
[[14], [14]]
bisa dikali
[[0, 0], [0, 0]]
[[19, 22], [43, 50]]
bisa dikali
[[0, 0, 0], [0, 0, 0]]
[[19, 24, 13], [43, 54, 31]]
tidak memenuhi syarat
18
-6
275
-2
tidak bisa dihitung determinan, bukan matrix bujursangkar
tidak bisa dihitung determinan, bukan matrix bujursangkar
>>> 
```

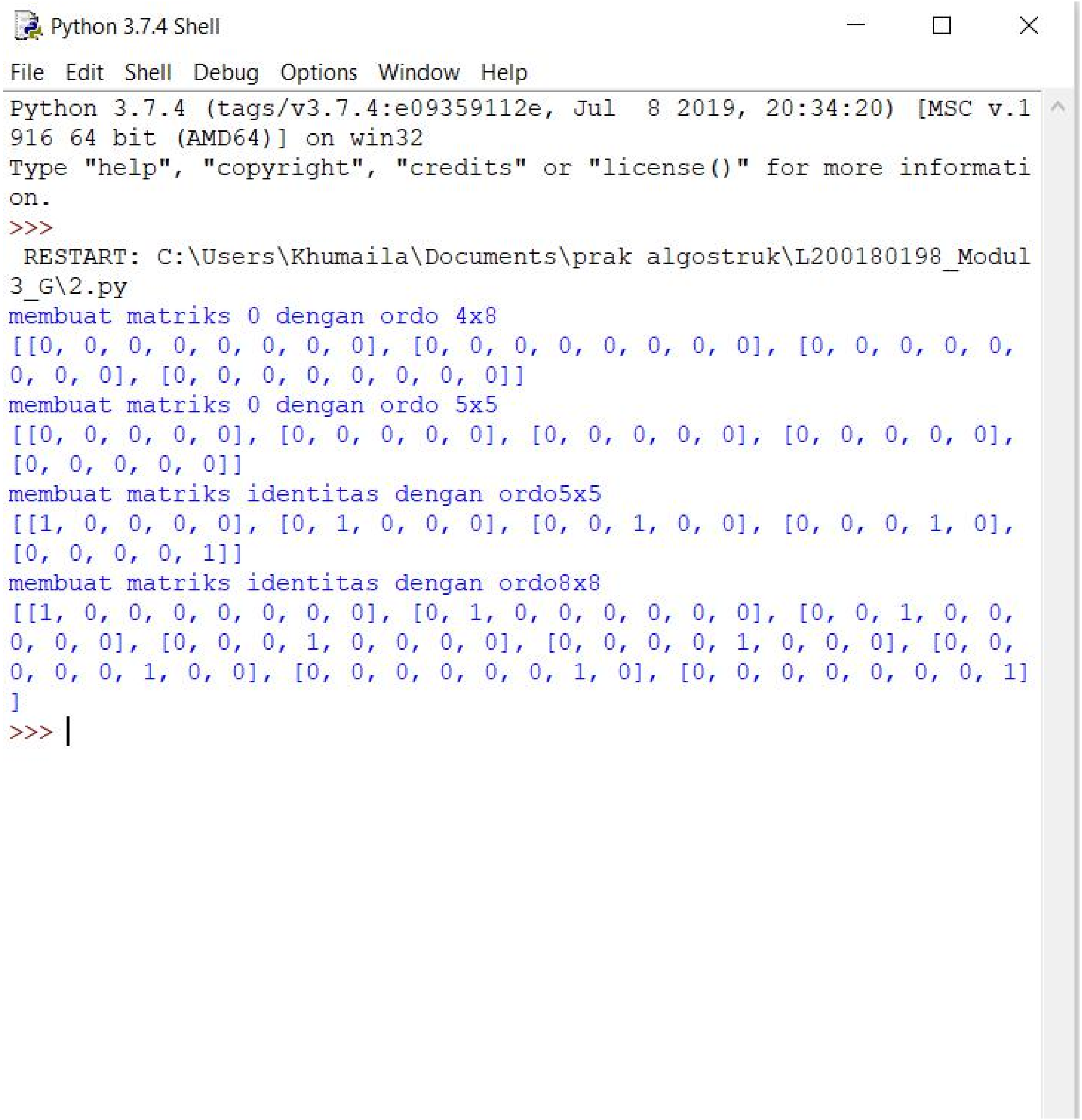
NO 2



2.py - C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\2.py (3.7.4)
File Edit Format Run Options Window Help

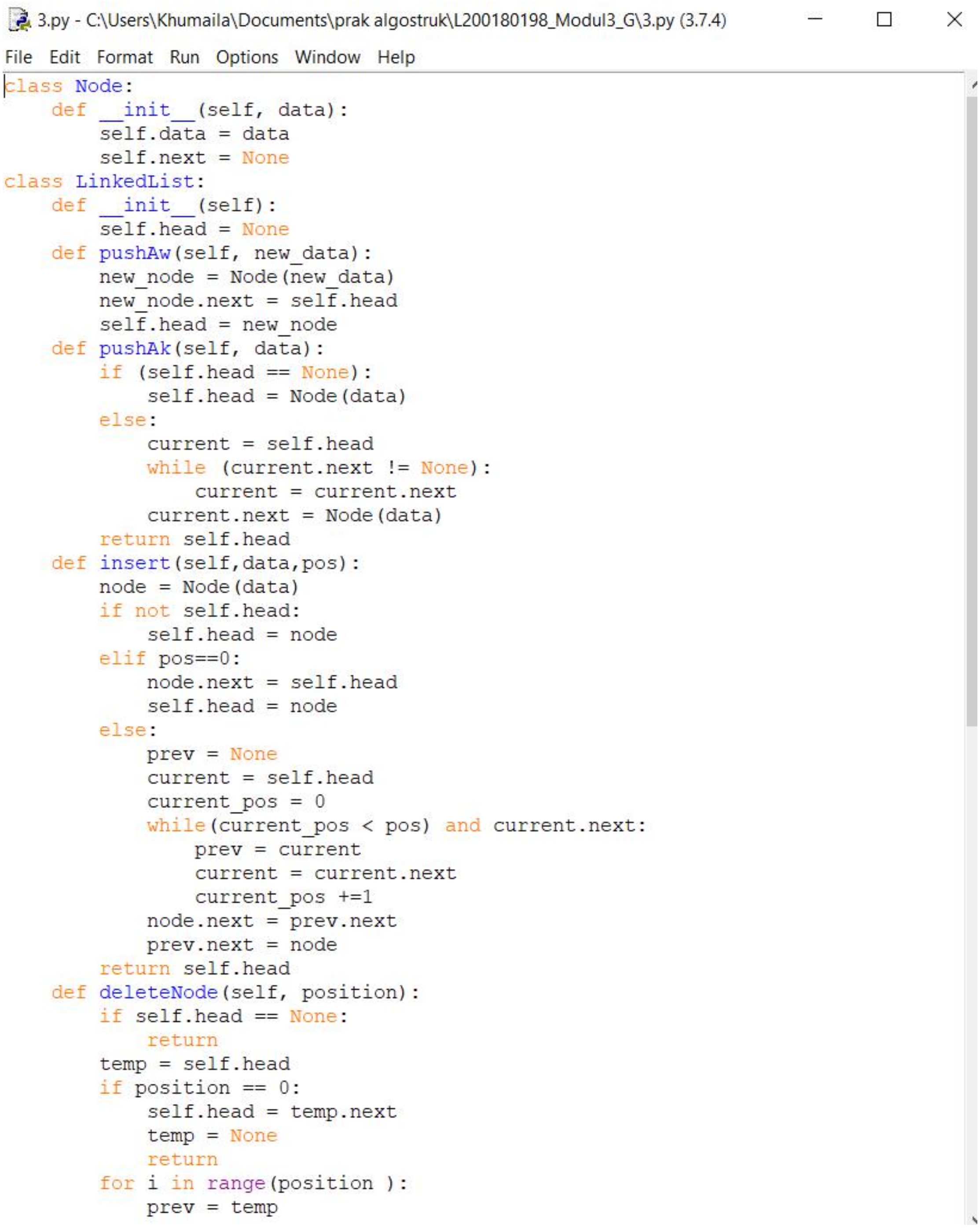
```
def membuatNol(n,m=None):
    if(m==None):
        m=n
    print("membuat matriks 0 dengan ordo "+str(n)+"x"+str(m))
    print([[0 for j in range(m)] for i in range(n)])

membuatNol(4,8)
membuatNol(5)

def membuatIdentitas(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)])

membuatIdentitas(5)
membuatIdentitas(8)
```

Hasil:



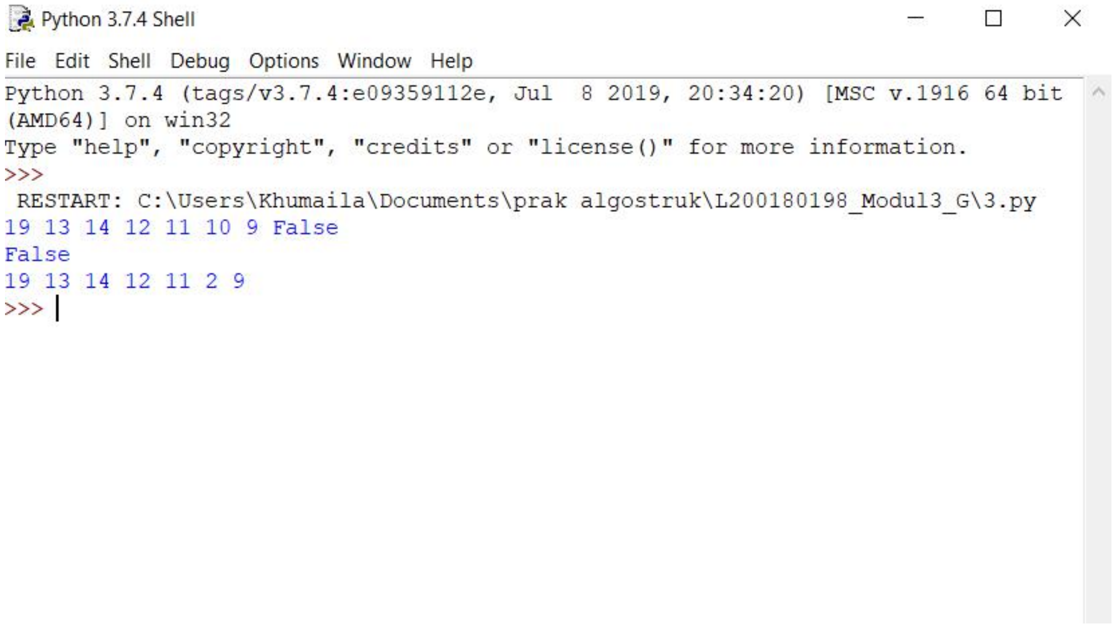
```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1
916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more informati
on.
>>> 
 RESTART: C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul
3_G\2.py
membuat matriks 0 dengan ordo 4x8
[[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0,
0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0]]
membuat matriks 0 dengan ordo 5x5
[[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0],
[0, 0, 0, 0, 0]]
membuat matriks identitas dengan ordo5x5
[[1, 0, 0, 0, 0], [0, 1, 0, 0, 0], [0, 0, 1, 0, 0], [0, 0, 0, 1, 0],
[0, 0, 0, 0, 1]]
membuat matriks identitas dengan ordo8x8
[[1, 0, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0,
0, 0, 0], [0, 0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0, 0], [0, 0,
0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 0, 0, 1]
]
>>> 
```

NO 3


3.py - C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\3.py (3.7.4)
File Edit Format Run Options Window Help

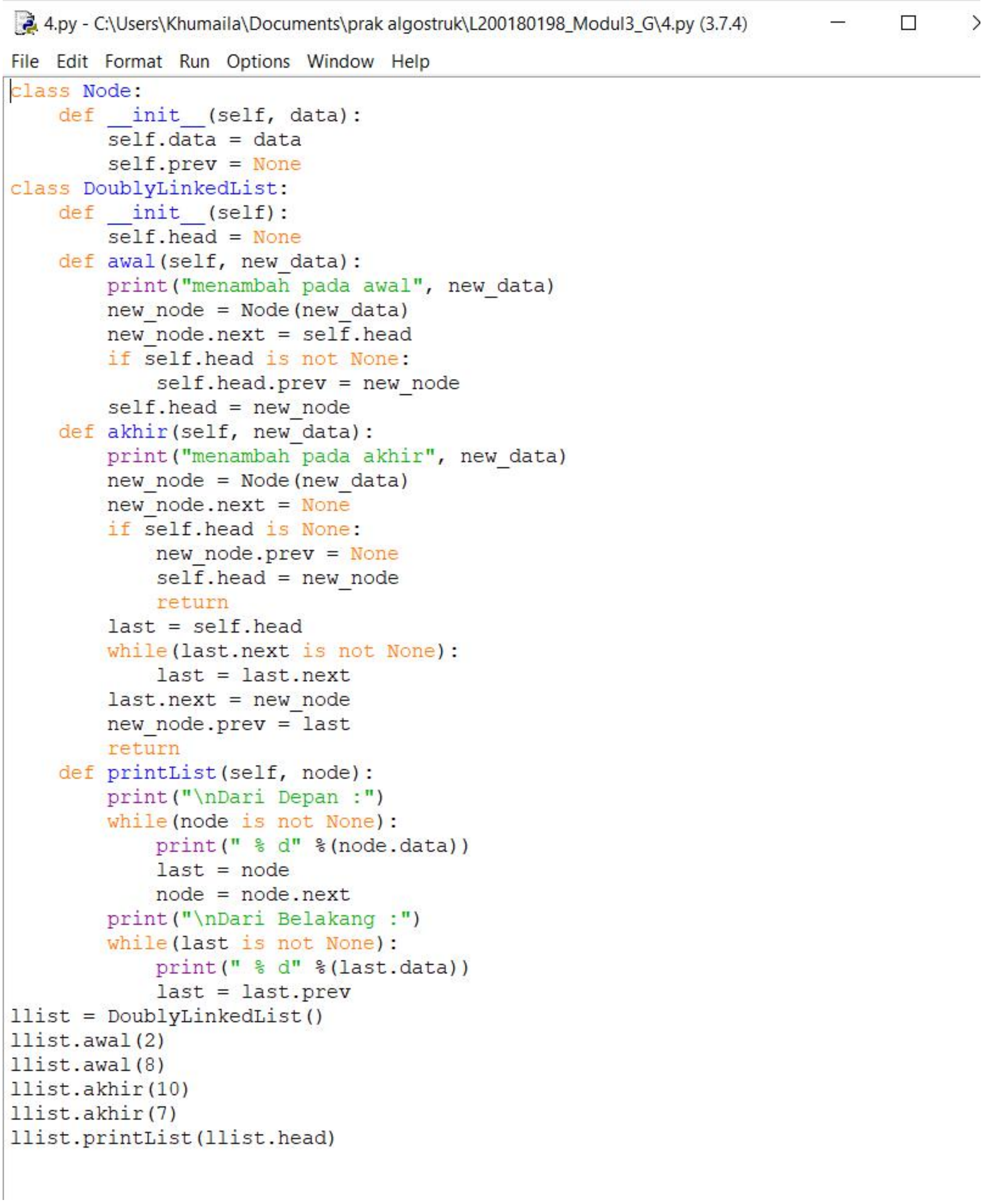
```python
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
class LinkedList:
    def __init__(self):
        self.head = None
    def pushAw(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node
    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
            node.next = prev.next
            prev.next = node
        return self.head
    def deleteNode(self, position):
        if self.head == None:
            return
        temp = self.head
        if position == 0:
            self.head = temp.next
            temp = None
            return
        for i in range(position ):
            prev = temp
            prev = temp
            temp = temp.next
            if temp is None:
                break
        if temp is None:
            return
        if temp.next is None:
            return
        prev.next = temp.next
        temp= None

    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(10)
llist.pushAw(11)
llist.pushAw(12)
llist.pushAw(14)
llist.pushAw(13)
llist.pushAw(19)
llist.pushAk(9)
llist.display()
llist.deleteNode(5)
llist.insert(2,5)
print(llist.search(21))
print(llist.search(29))
llist.display()
```

Hasil:



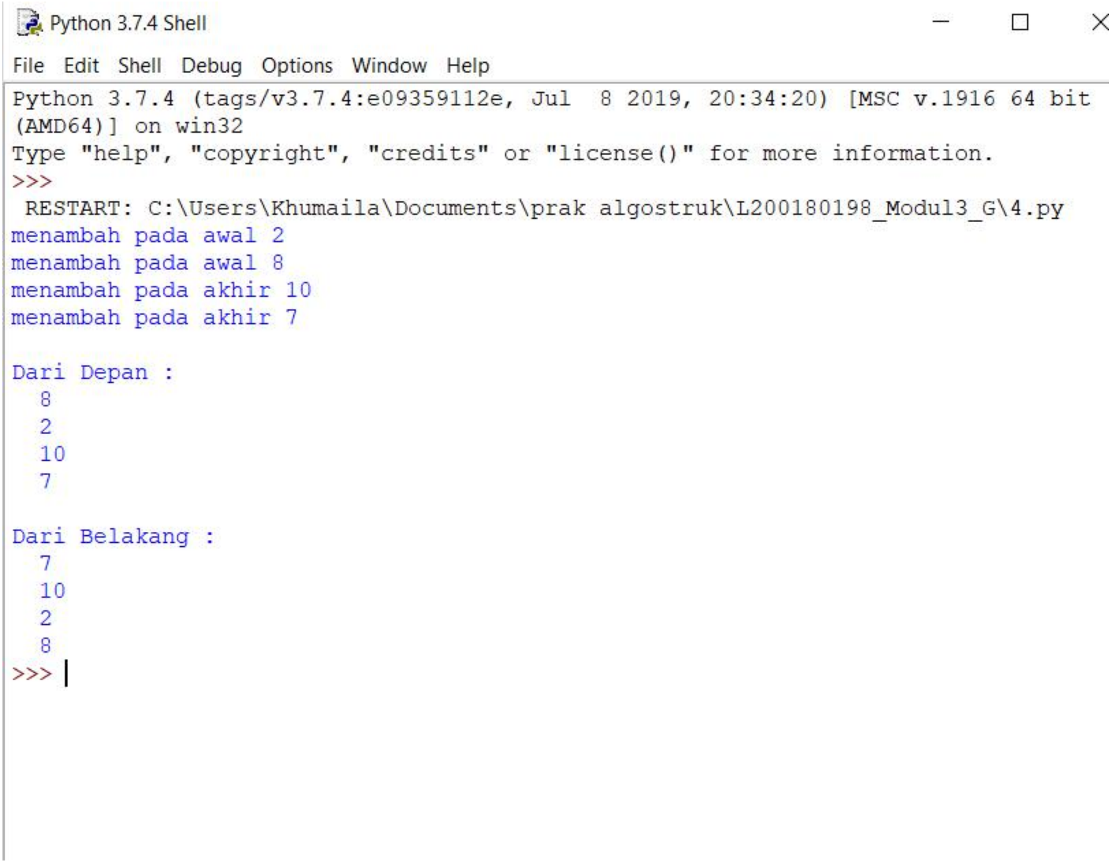
```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\3.py
19 13 14 12 11 10 9 False
False
19 13 14 12 11 2 9
>>> |
```

NO 4


4.py - C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\4.py (3.7.4)
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(2)
llist.awal(8)
llist.akhir(10)
llist.akhir(7)
llist.printList(llist.head)
```

Hasil:



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
 RESTART: C:\Users\Khumaila\Documents\prak algostruk\L200180198_Modul3_G\4.py
menambah pada awal 2
menambah pada awal 8
menambah pada akhir 10
menambah pada akhir 7

Dari Depan :
  8
  2
  10
  7

Dari Belakang :
  7
  10
  2
  8
>>> 
```