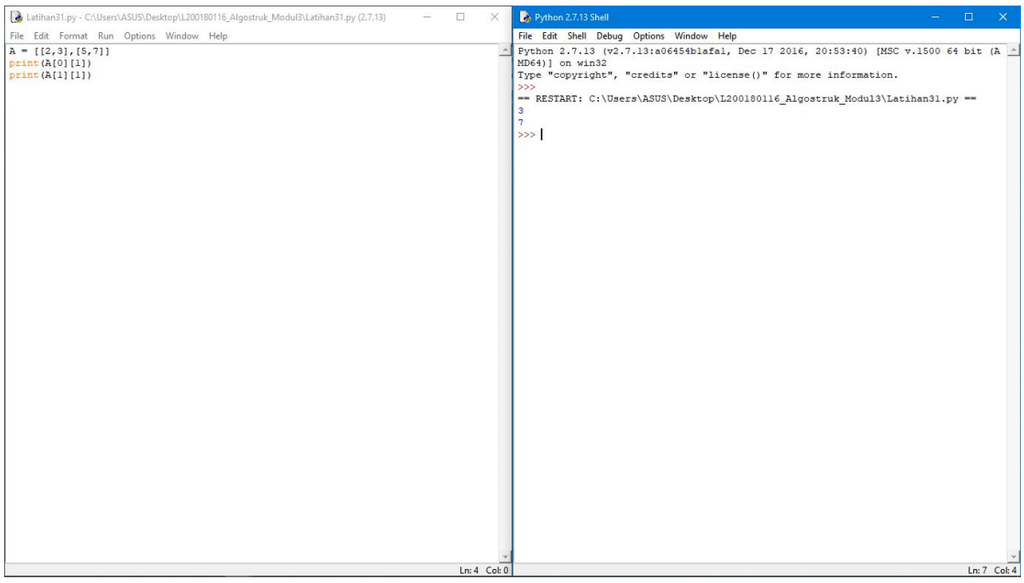
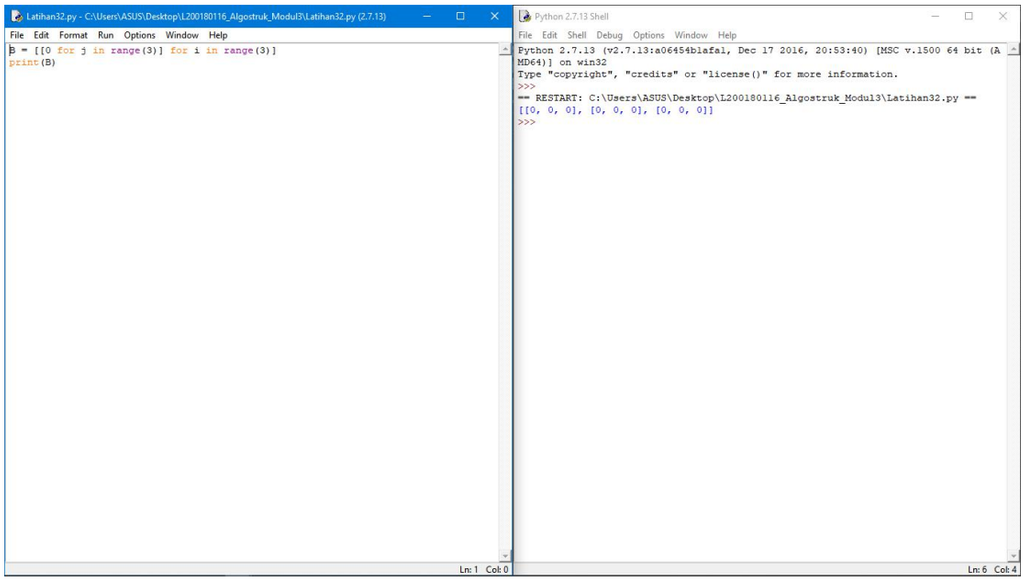
Ilyas Raihan Nadhif
L200180119
E

Modul 3

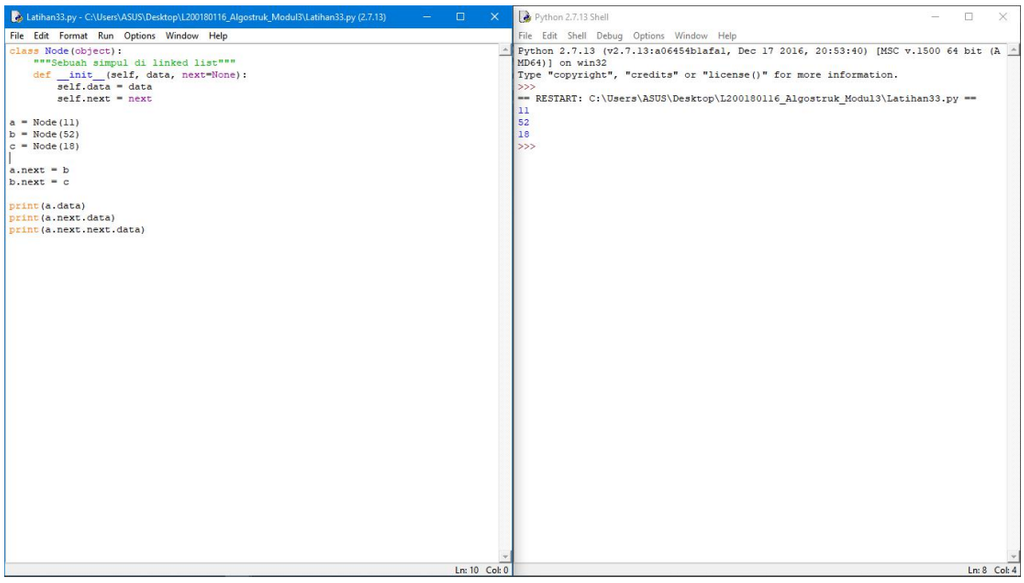
Latihan 1



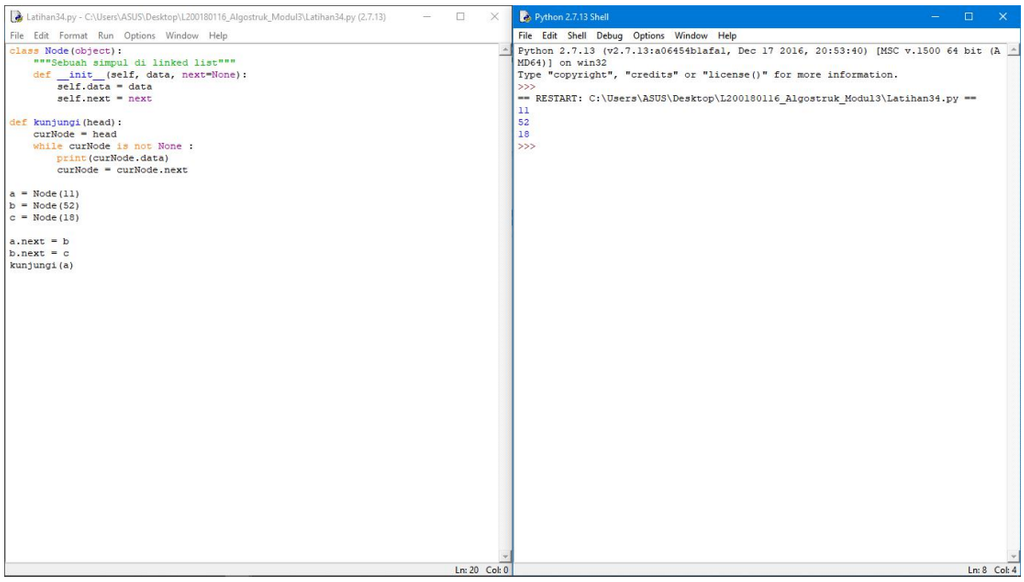
Latihan 2



Latihan 3



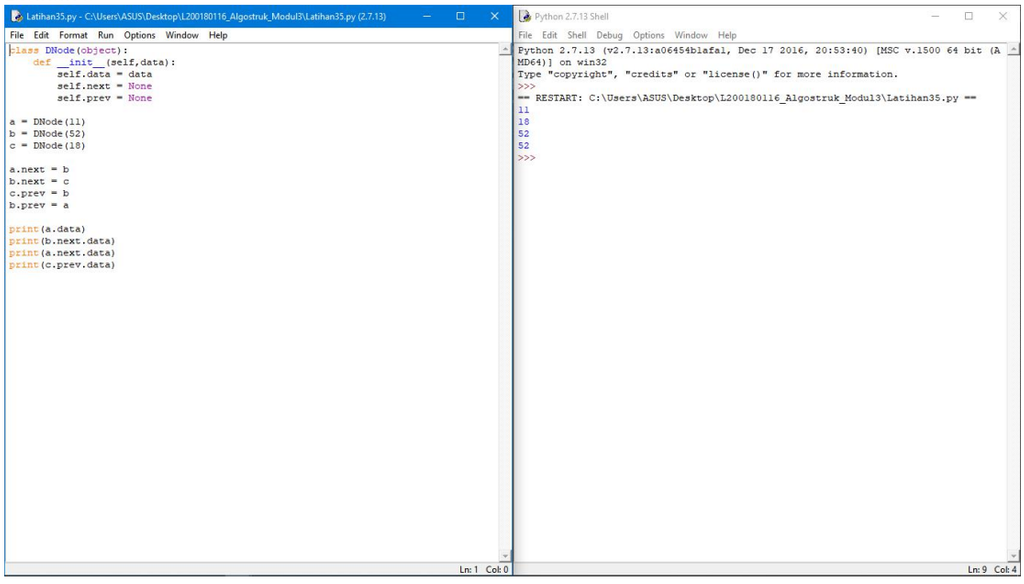
Latihan 4



```
Latihan34.py - C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Latihan34.py (2.7.13)
File  Edit  Format  Run  Options  Window  Help
class Node(object):
    """Sebuah simpul di linked list"""
    def __init__(self, data, next=None):
        self.data = data
        self.next = next

def kunjungi(head):
    curNode = head
    while curNode is not None :
        print(curNode.data)
        curNode = curNode.next

a = Node(11)
b = Node(52)
c = Node(18)

a.next = b
b.next = c
kunjungi(a)
Ln: 20  Col: 0
```

```
Python 2.7.13 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (A
MD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Latihan34.py ==
11
52
18
>>>
Ln: 8  Col: 4
```

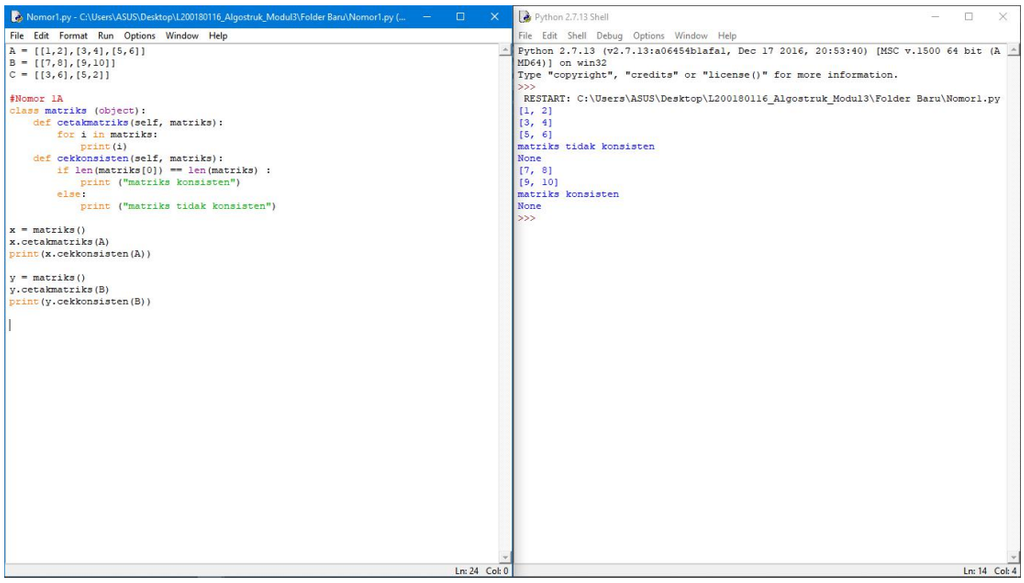
Latihan 5



```
Latihan35.py - C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Latihan35.py (2.7.13)
File  Edit  Format  Run  Options  Window  Help
class DNode(object):
    def __init__(self,data):
        self.data = data
        self.next = None
        self.prev = None

a = DNode(11)
b = DNode(52)
c = DNode(18)

a.next = b
b.next = c
c.prev = b
b.prev = a

print(a.data)
print(b.next.data)
print(a.next.data)
print(c.prev.data)
Ln: 1  Col: 0
```

```
Python 2.7.13 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (A
MD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Latihan35.py ==
11
18
52
52
>>>
Ln: 9  Col: 4
```

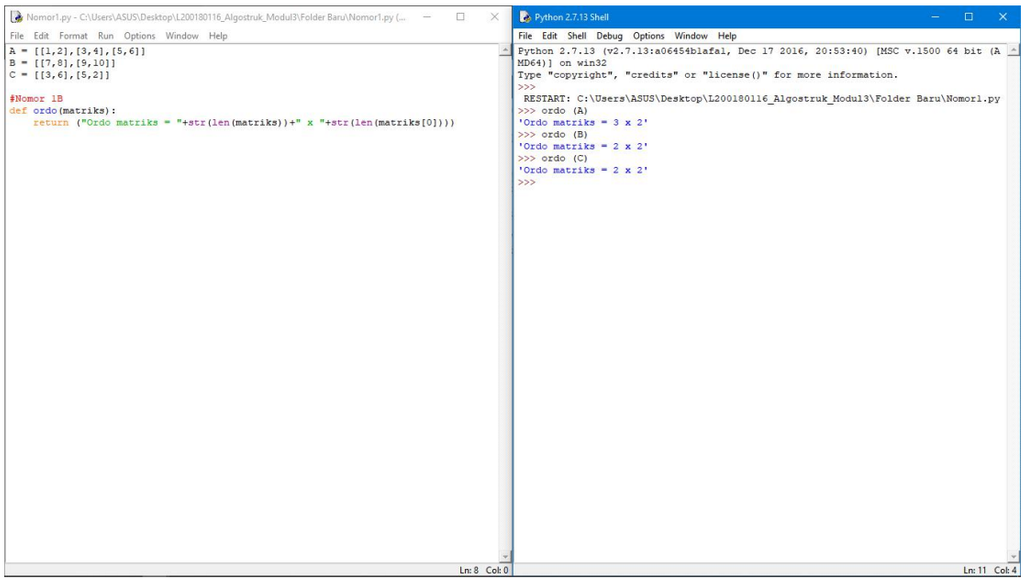
Tugas 1a



```
A = [[1,2],[3,4],[5,6]]
B = [[7,8],[9,10]]
C = [[3,6],[5,2]]

#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) :
            print ("matriks konsisten")
        else:
            print ("matriks tidak konsisten")

x = matriks()
x.cetakmatriks(A)
print(x.cekkonsisten(A))

y = matriks()
y.cetakmatriks(B)
print(y.cekkonsisten(B))
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor1.py
[1, 2]
[3, 4]
[5, 6]
matriks tidak konsisten
None
[7, 8]
[9, 10]
matriks konsisten
None
>>>
```

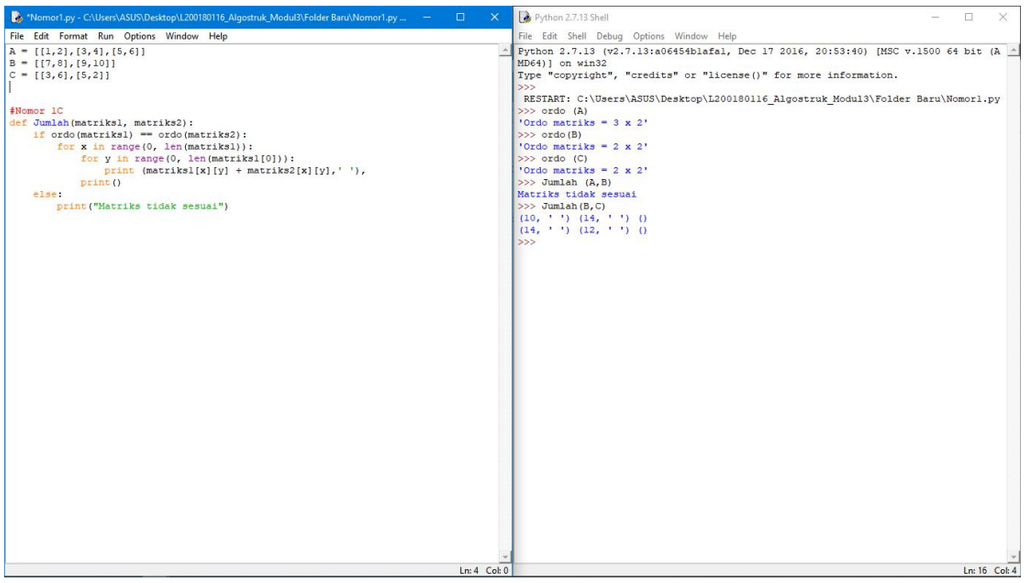
Tugas 1b



```
A = [[1,2],[3,4],[5,6]]
B = [[7,8],[9,10]]
C = [[3,6],[5,2]]

#Nomor 1B
def ordo(matriks):
    return ("Ordo matriks = "+str(len(matriks))+" x "+str(len(matriks[0])))
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor1.py
>>> ordo (A)
'Ordo matriks = 3 x 2'
>>> ordo (B)
'Ordo matriks = 2 x 2'
>>> ordo (C)
'Ordo matriks = 2 x 2'
>>>
```

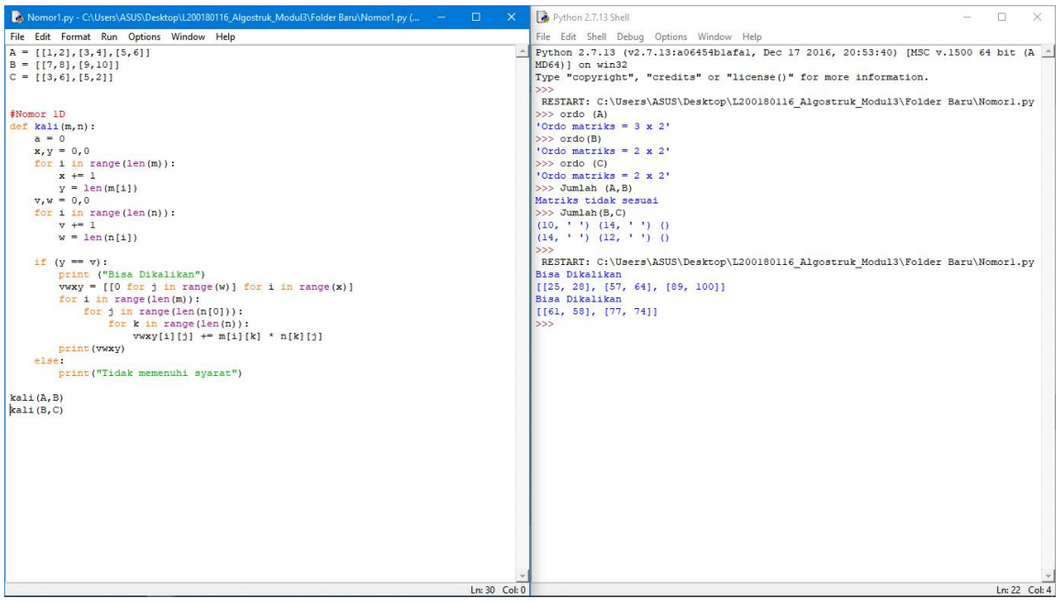
Tugas 1c



```
A = [[1,2],[3,4],[5,6]]
B = [[7,8],[9,10]]
C = [[3,6],[5,2]]

#Nomor 1C
def Jumlah(matriks1, matriks2):
    if ordo(matriks1) == ordo(matriks2):
        for x in range(0, len(matriks1)):
            for y in range(0, len(matriks1[0])):
                print (matriks1[x][y] + matriks2[x][y],' '),
            print()
    else:
        print("Matriks tidak sesuai")
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor1.py
>>> ordo (A)
'Ordo matriks = 3 x 2'
>>> ordo(B)
'Ordo matriks = 2 x 2'
>>> ordo (C)
'Ordo matriks = 2 x 2'
>>> Jumlah (A,B)
Matriks tidak sesuai
>>> Jumlah(B,C)
(10, ' ') (14, ' ') ()
(14, ' ') (12, ' ') ()
>>>
```

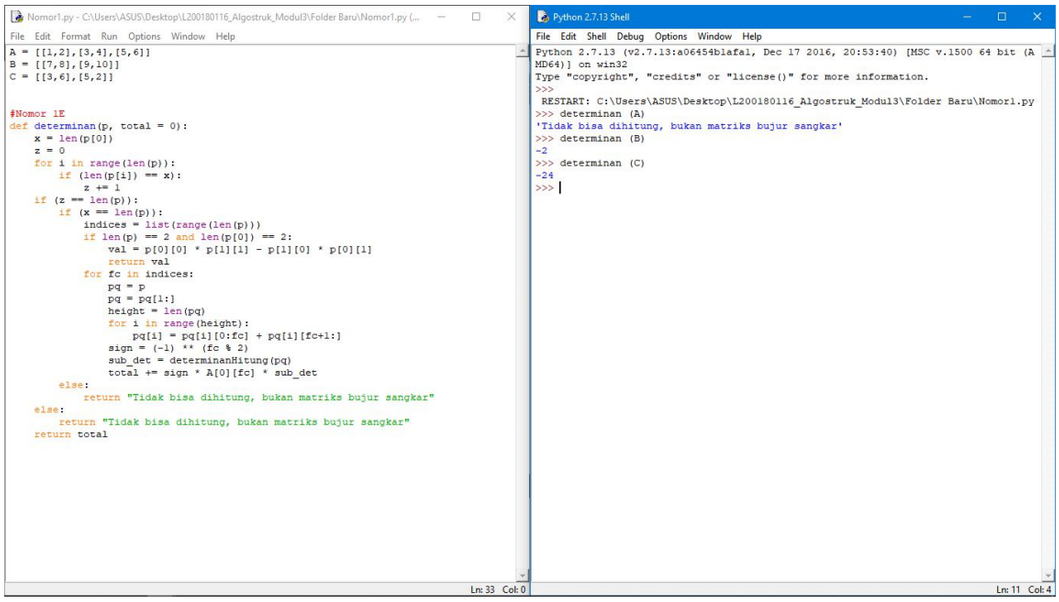
Tugas 1d



```
A = [[1,2],[3,4],[5,6]]
B = [[7,8],[9,10]]
C = [[3,6],[5,2]]


#Nomor 1D
def kali(m,n):
    a = 0
    x,y = 0,0
    for i in range(len(m)):
        x += 1
        y = len(m[i])
    v,w = 0,0
    for i in range(len(n)):
        v += 1
        w = len(n[i])

    if (y == v):
        print ("Bisa Dikalikan")
        vwxy = [[0 for j in range(w)] for i in range(x)]
        for i in range(len(m)):
            for j in range(len(n[0])):
                for k in range(len(n)):
                    vwxy[i][j] += m[i][k] * n[k][j]
        print(vwxy)
    else:
        print("Tidak memenuhi syarat")

kali(A,B)
kali(B,C)
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (A
MD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor1.py
>>> ordo (A)
'Ordo matriks = 3 x 2'
>>> ordo(B)
'Ordo matriks = 2 x 2'
>>> ordo (C)
'Ordo matriks = 2 x 2'
>>> Jumlah (A,B)
Matriks tidak sesuai
>>> Jumlah(B,C)
(10, ' ') (14, ' ') ()
(14, ' ') (12, ' ') ()
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor1.py
Bisa Dikalikan
[[25, 28], [57, 64], [89, 100]]
Bisa Dikalikan
[[61, 58], [77, 74]]
>>>
```

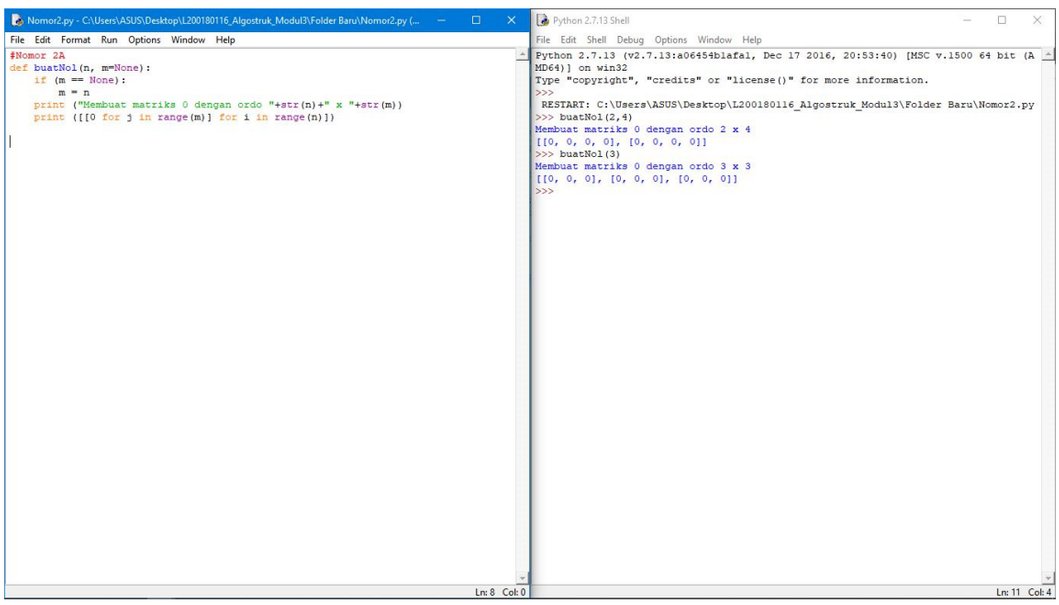
Tugas 1e



```
A = [[1,2],[3,4],[5,6]]
B = [[7,8],[9,10]]
C = [[3,6],[5,2]]


#Nomor 1E
def determinan(p, total = 0):
    x = len(p[0])
    z = 0
    for i in range(len(p)):
        if (len(p[i]) == x):
            z += 1
    if (z == len(p)):
        if (x == len(p)):
            indices = list(range(len(p)))
            if len(p) == 2 and len(p[0]) == 2:
                val = p[0][0] * p[1][1] - p[1][0] * p[0][1]
                return val
            for fc in indices:
                pq = p
                pq = pq[1:]
                height = len(pq)
                for i in range(height):
                    pq[i] = pq[i][0:fc] + pq[i][fc+1:]
                sign = (-1) ** (fc % 2)
                sub_det = determinanHitung(pq)
                total += sign * A[0][fc] * sub_det
        else:
            return "Tidak bisa dihitung, bukan matriks bujur sangkar"
    else:
        return "Tidak bisa dihitung, bukan matriks bujur sangkar"
    return total
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (A
MD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor1.py
>>> determinan (A)
'Tidak bisa dihitung, bukan matriks bujur sangkar'
>>> determinan (B)
-2
>>> determinan (C)
-24
>>>
```

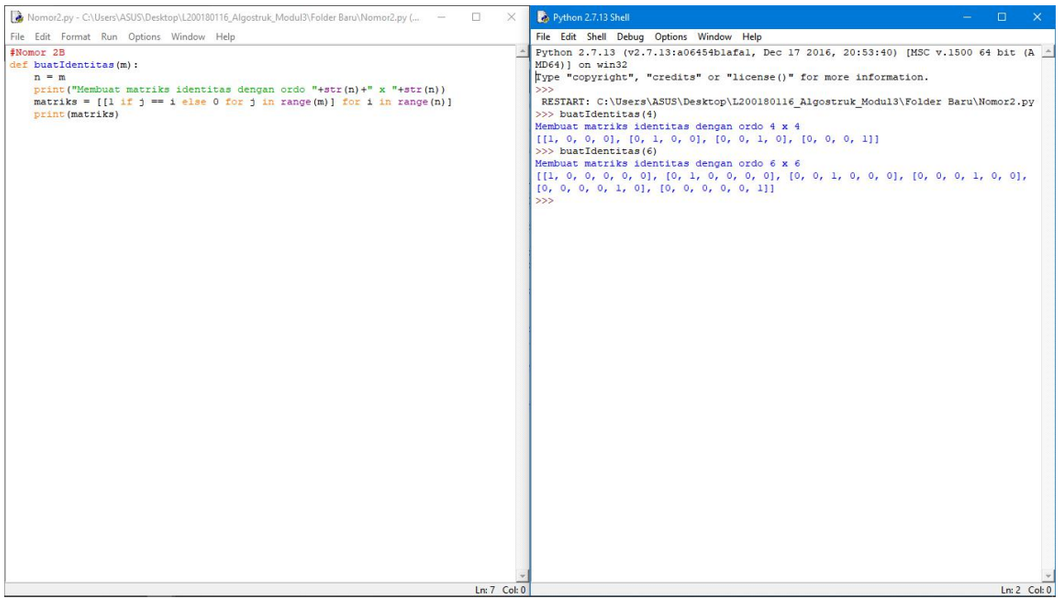
Tugas 2a



```
#Nomor 2A
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)])
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (A
MD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor2.py
>>> buatNol(2,4)
Membuat matriks 0 dengan ordo 2 x 4
[[0, 0, 0, 0], [0, 0, 0, 0]]
>>> buatNol(3)
Membuat matriks 0 dengan ordo 3 x 3
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]
>>>
```

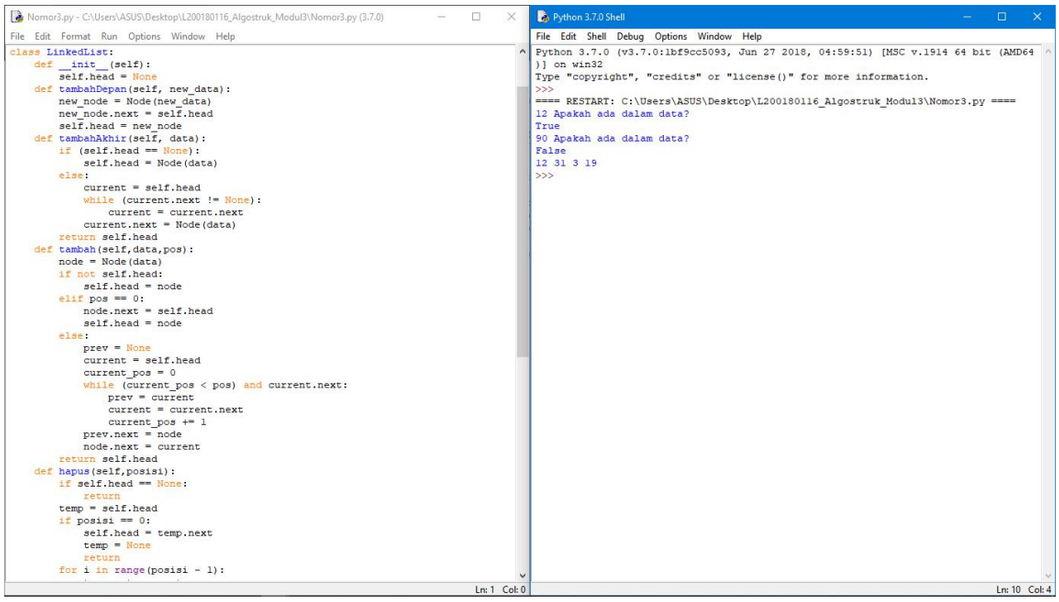
Tugas 2b



```
#Nomor 2B
def buatIdentitas(m):
    n = m
    print("Membuat matriks identitas dengan ordo "+str(n)+" x "+str(n))
    matriks = [[1 if j == i else 0 for j in range(m)] for i in range(n)]
    print(matriks)
```

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Folder Baru\Nomor2.py
>>> buatIdentitas(4)
Membuat matriks identitas dengan ordo 4 x 4
[[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]
>>> buatIdentitas(6)
Membuat matriks identitas dengan ordo 6 x 6
[[1, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 1]]
>>>
```

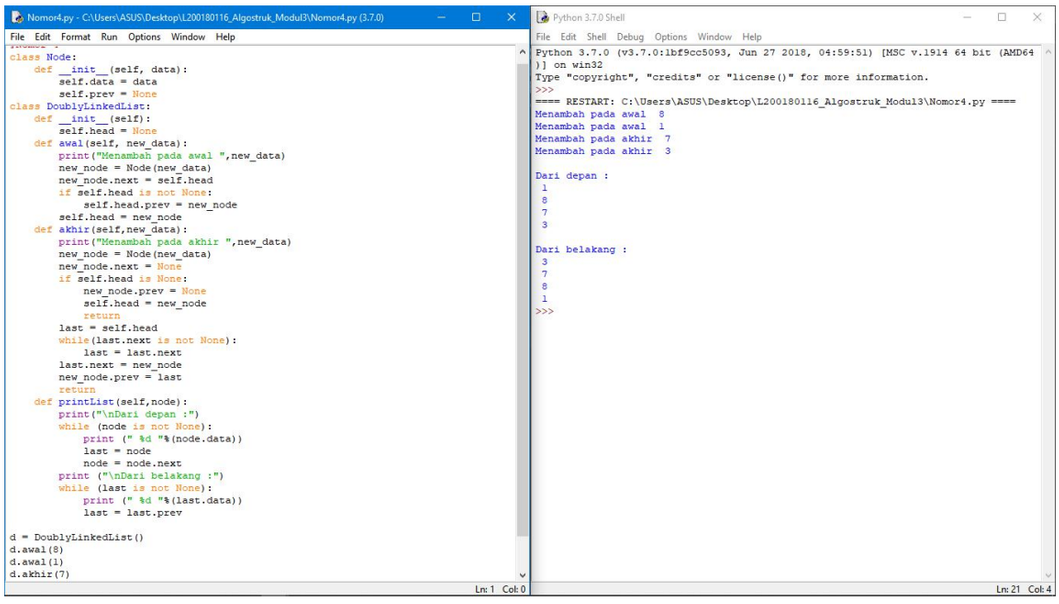
Tugas 3



```
class LinkedList:
    def __init__(self):
        self.head = None
    def tambahDepan(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node
    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
    def tambah(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 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):
```

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Nomor3.py ====
12 Apakah ada dalam data?
True
90 Apakah ada dalam data?
False
12 31 3 19
>>>
```

Tugas 4



```
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

d = DoublyLinkedList()
d.awal(8)
d.awal(1)
d.akhir(7)
```

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\ASUS\Desktop\L200180116_Algostruk_Modul3\Nomor4.py ====
Menambah pada awal  8
Menambah pada awal  1
Menambah pada akhir  7
Menambah pada akhir  3

Dari depan :
 1
 8
 7
 3

Dari belakang :
 3
 7
 8
 1
>>>
```