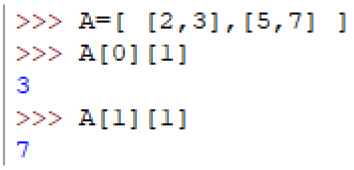
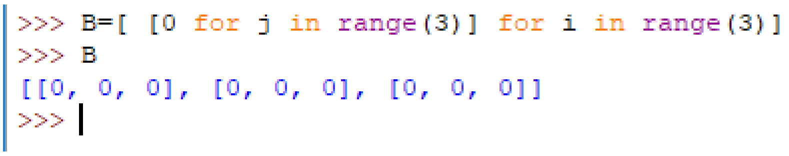
NAMA : DANANG AJI NUGROHO
NIM : L200180015
KELAS : A

LATIHAN 3.1



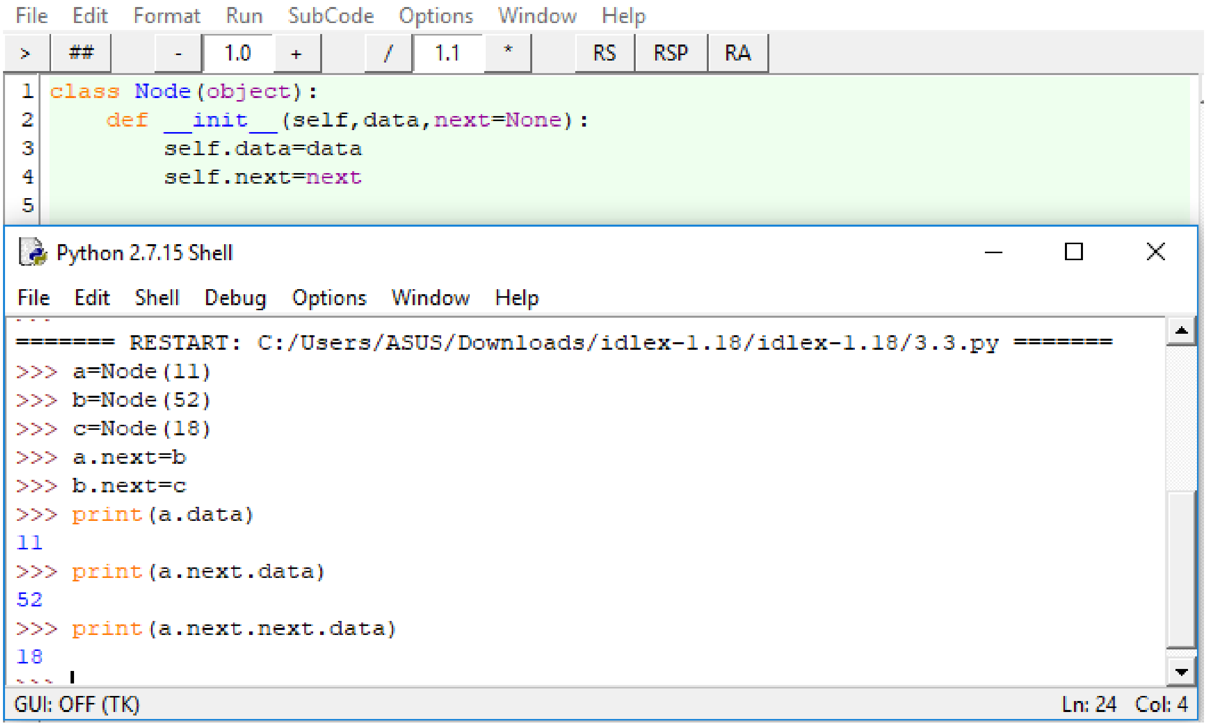
```
>>> A=[ [2,3],[5,7] ]
>>> A[0][1]
3
>>> A[1][1]
7
```

LATIHAN 3.2



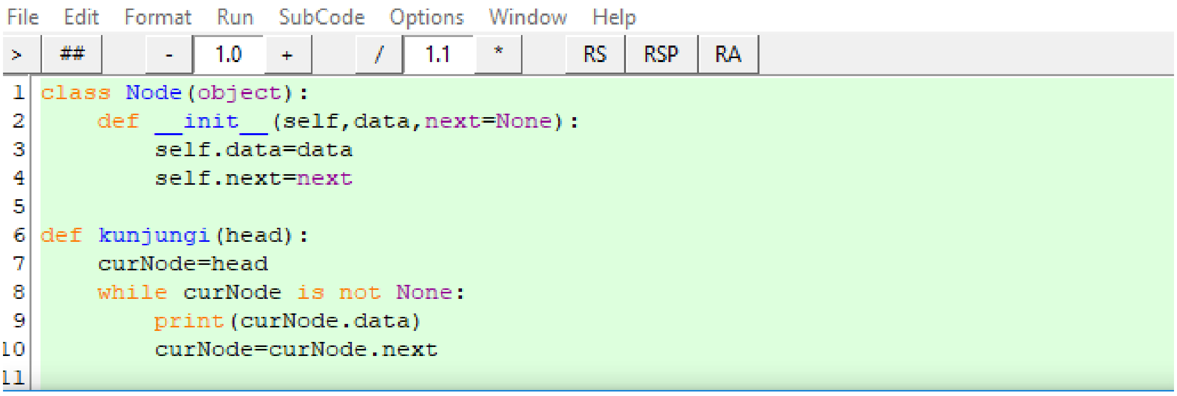
```
>>> B=[ [0 for j in range(3)] for i in range(3)]
>>> B
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]
>>> 
```

LATIHAN 3.3


File Edit Format Run SubCode Options Window Help
> ## - 1.0 + / 1.1 * RS RSP RA
Python 2.7.15 Shell
File Edit Shell Debug Options Window Help
GUI: OFF (TK)
Ln: 24 Col: 4

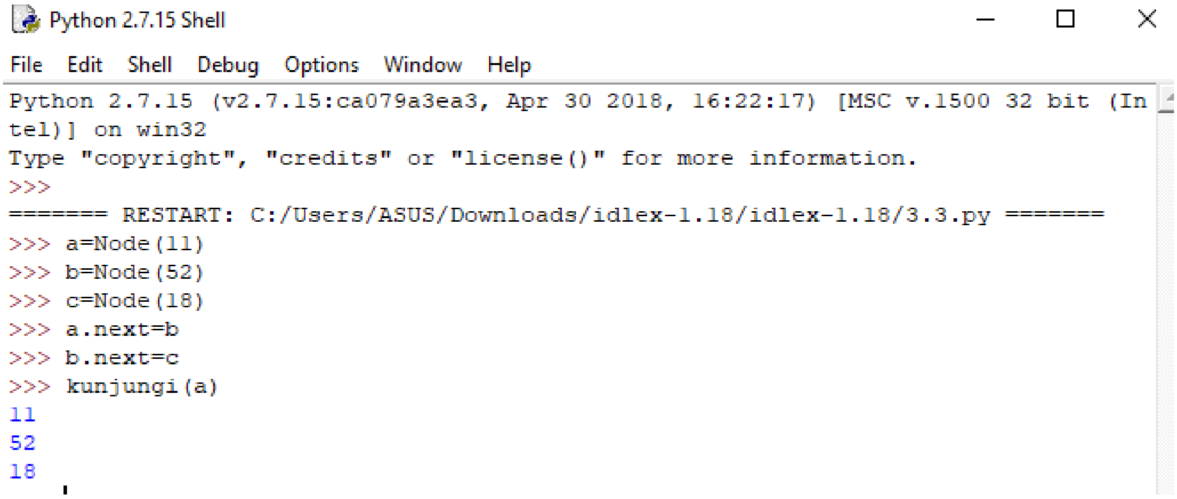
```
class Node(object):
    def __init__(self,data,next=None):
        self.data=data
        self.next=next

```

```
======= RESTART: C:/Users/ASUS/Downloads/idlex-1.18/idlex-1.18/3.3.py =======
>>> a=Node(11)
>>> b=Node(52)
>>> c=Node(18)
>>> a.next=b
>>> b.next=c
>>> print(a.data)
11
>>> print(a.next.data)
52
>>> print(a.next.next.data)
18
```

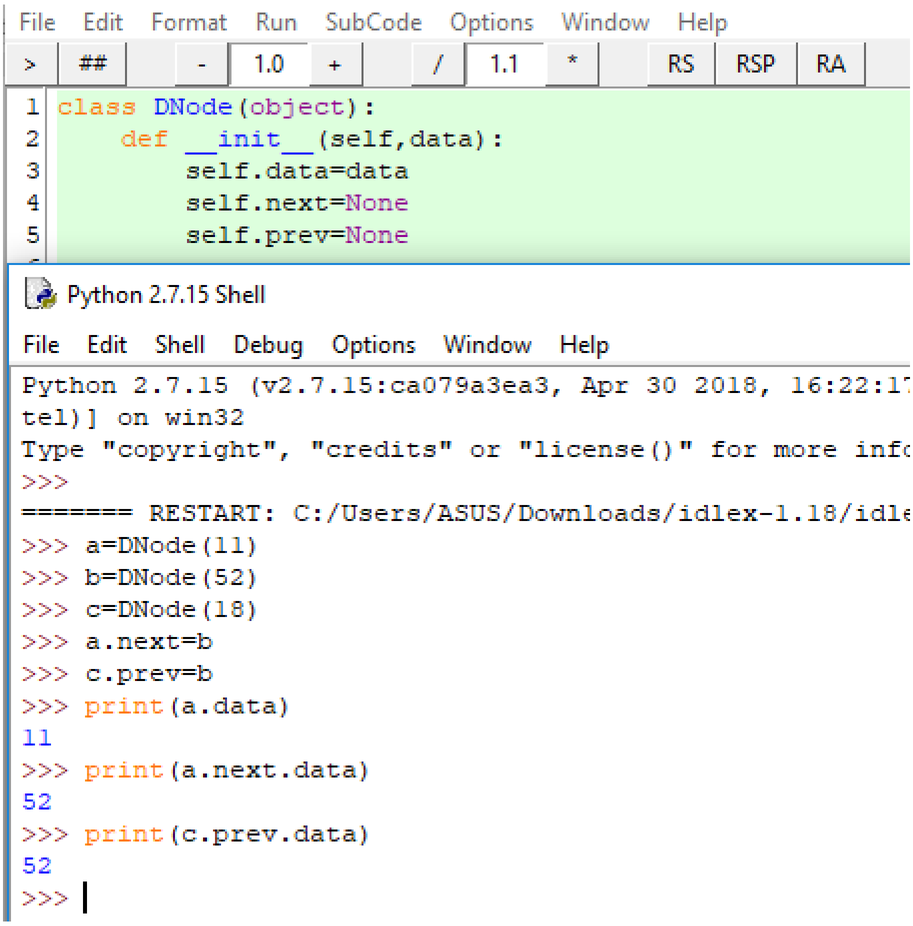


File Edit Format Run SubCode Options Window Help
> ## - 1.0 + / 1.1 * RS RSP RA

```python
class Node(object):
    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

```



```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:22:17) [MSC v.1500 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
======= RESTART: C:/Users/ASUS/Downloads/idlex-1.18/idlex-1.18/3.3.py =======
>>> a=Node(11)
>>> b=Node(52)
>>> c=Node(18)
>>> a.next=b
>>> b.next=c
>>> kunjungi(a)
11
52
18
```


File Edit Format Run SubCode Options Window Help
> ## - 1.0 + / 1.1 * RS RSP RA
Python 2.7.15 Shell
File Edit Shell Debug Options Window Help

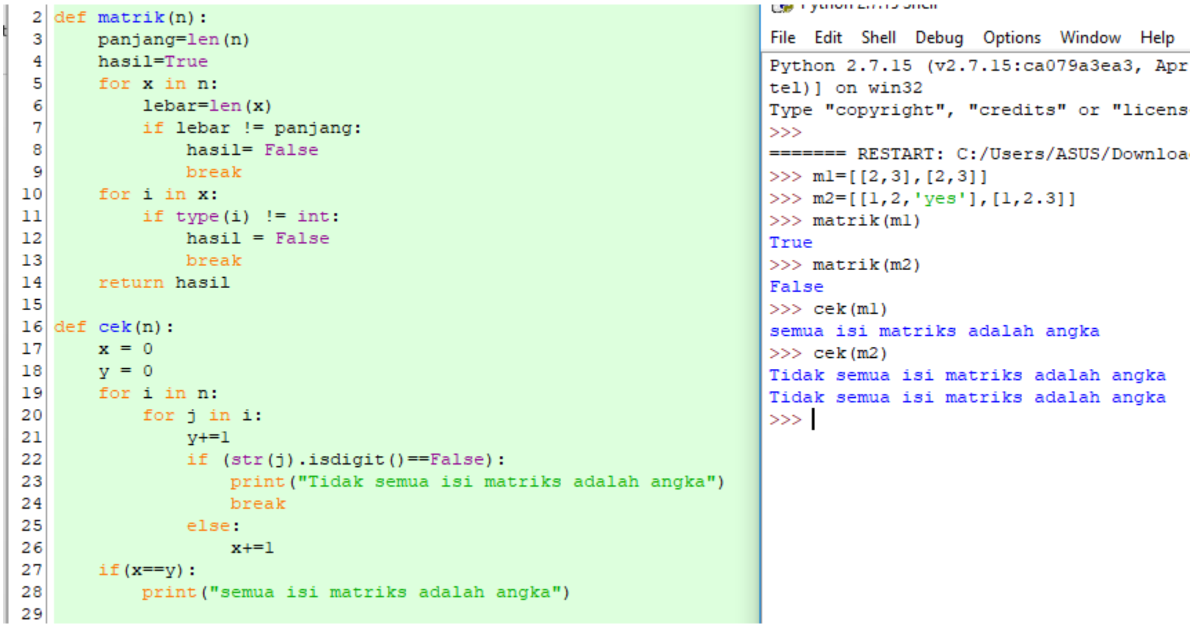
```
class DNode(object):
    def __init__(self,data):
        self.data=data
        self.next=None
        self.prev=None
```

```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:22:17
tel)] on win32
Type "copyright", "credits" or "license()" for more info
>>> 
======= RESTART: C:/Users/ASUS/Downloads/idlex-1.18/idle
>>> a=DNode(11)
>>> b=DNode(52)
>>> c=DNode(18)
>>> a.next=b
>>> c.prev=b
>>> print(a.data)
11
>>> print(a.next.data)
52
>>> print(c.prev.data)
52
>>> 
```

SOAL-SOAL UNTUK MAHASISWA

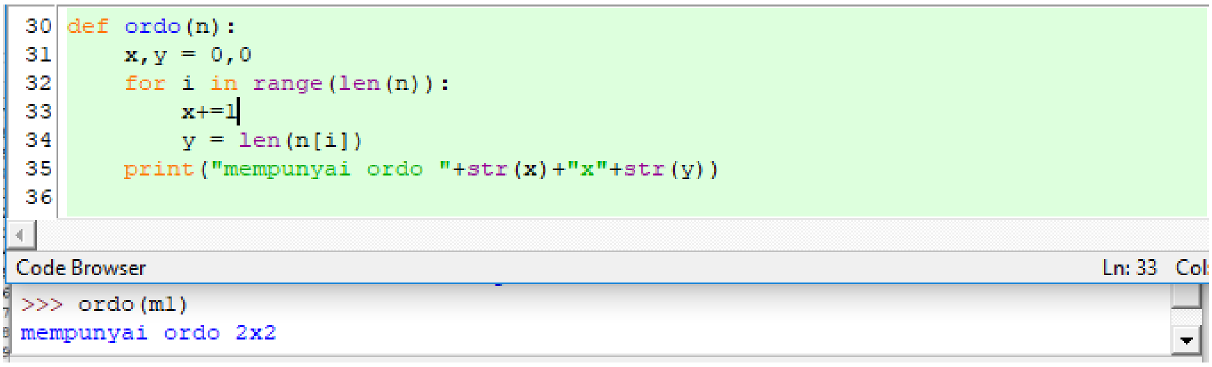
1. ARRAY DUA DIMENSI, MATRIX YANG BERISI ANGKA-ANGKA.

A. FUNGSI CEK ISI DAN UKURAN MATRIX



```
def matrik(n):
    panjang=len(n)
    hasil=True
    for x in n:
        lebar=len(x)
        if lebar != panjang:
            hasil= False
            break
    for i in x:
        if type(i) != int:
            hasil = False
            break
    return hasil

def cek(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")

```

```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr
tel)] on win32
Type "copyright", "credits" or "licens
>>> 
======= RESTART: C:/Users/ASUS/Downloa
>>> m1=[[2,3],[2,3]]
>>> m2=[[1,2,'yes'],[1,2.3]]
>>> matrik(m1)
True
>>> matrik(m2)
False
>>> cek(m1)
semua isi matriks adalah angka
>>> cek(m2)
Tidak semua isi matriks adalah angka
Tidak semua isi matriks adalah angka
>>> 
```

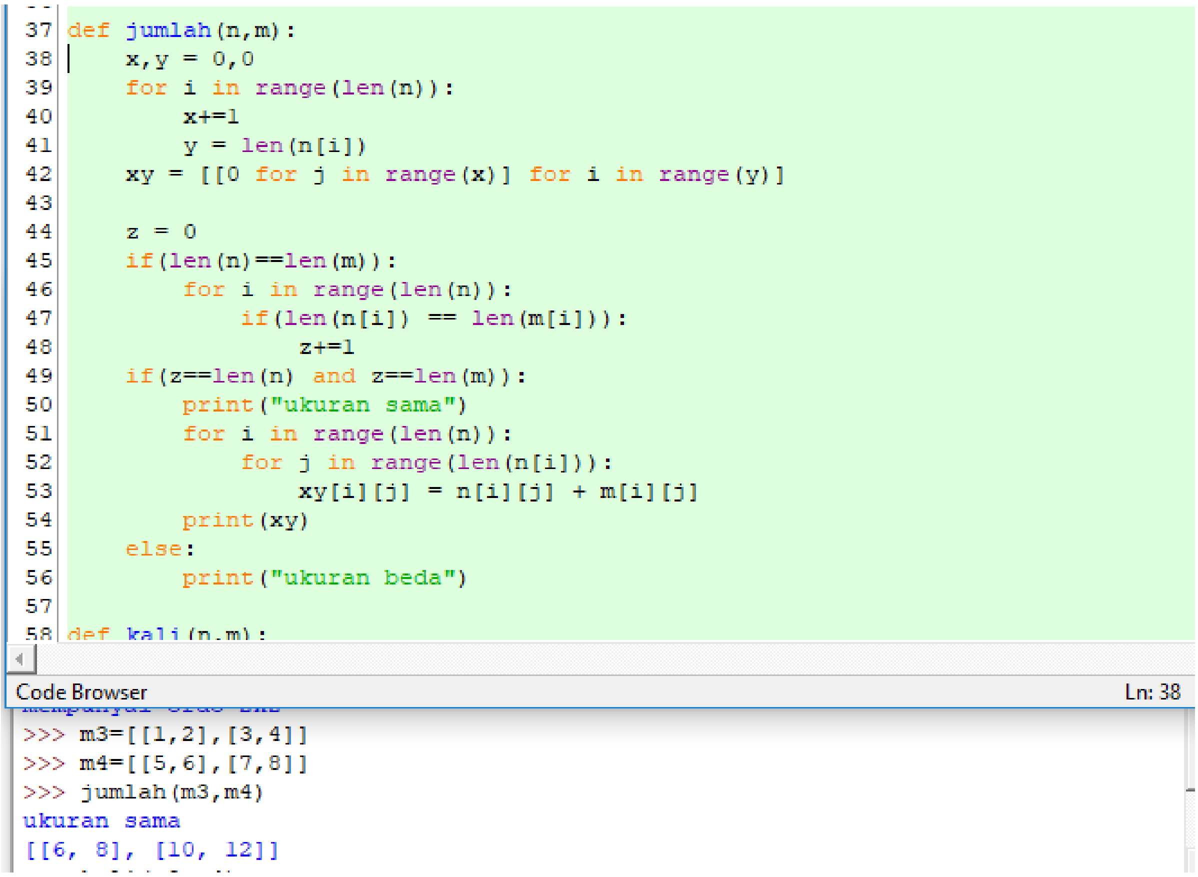


```
def ordo(n):
    x,y = 0,0
    for i in range(len(n)):
        x+=1
        y = len(n[i])
    print("mempunyai ordo "+str(x)+"x"+str(y))
```

```
>>> ordo(m1)
mempunyai ordo 2x2
```

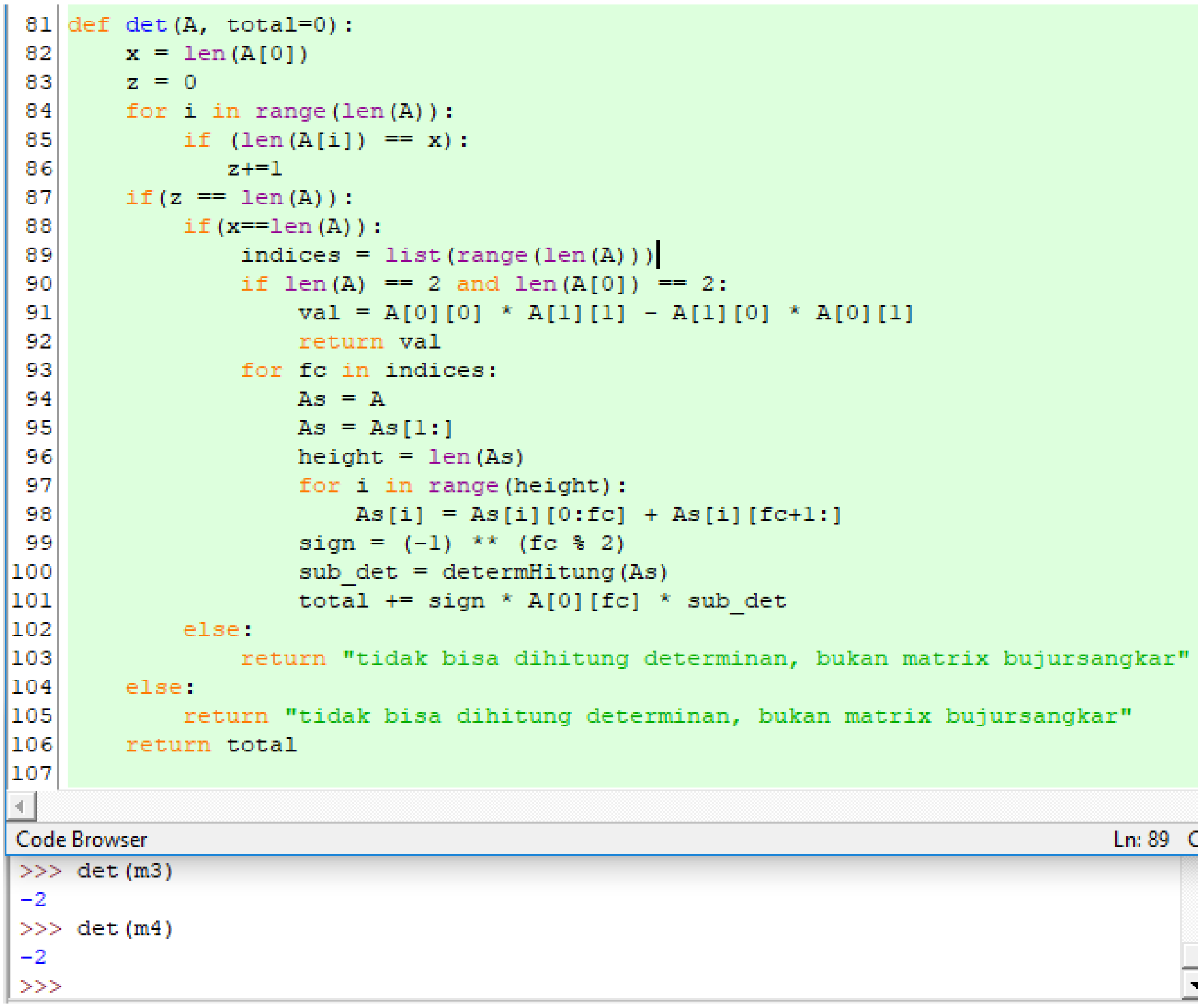
B. FUNGSI MENGAMBIL UKURAN MATRIX

C. FUNGSI MENJUMLAHKAN MATRIX



```
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")

def kali(n,m):
```

```
>>> m3=[[1,2],[3,4]]
>>> m4=[[5,6],[7,8]]
>>> jumlah(m3,m4)
ukuran sama
[[6, 8], [10, 12]]
```

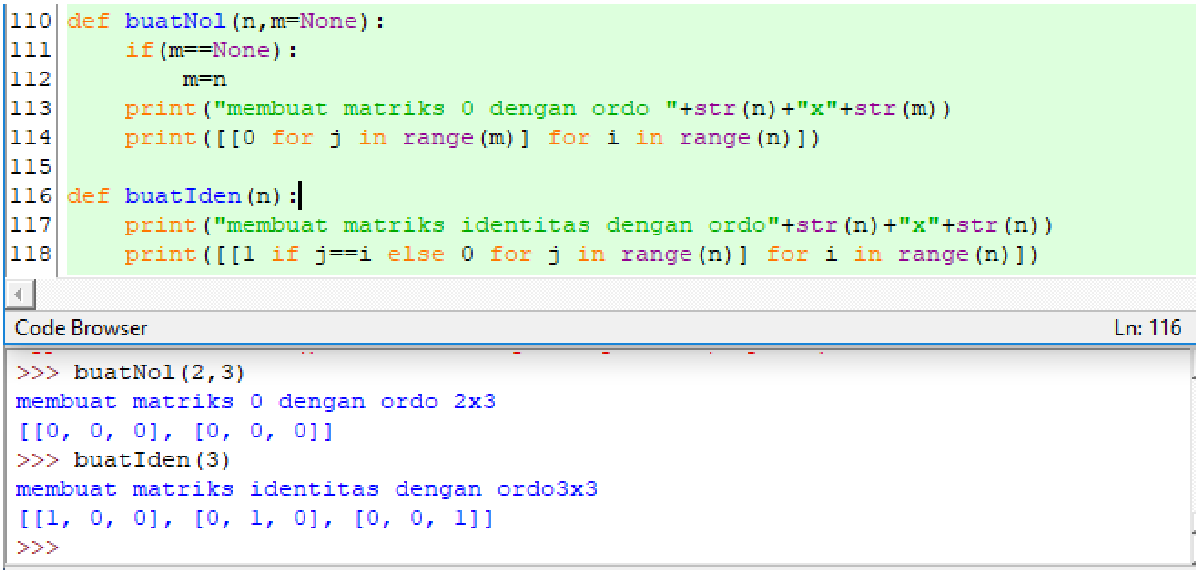


```
def det(A, total=0):
    x = len(A[0])
    z = 0
    for i in range(len(A)):
        if (len(A[i]) == x):
            z+=1
    if(z == len(A)):
        if(x==len(A)):
            indices = list(range(len(A)))
            if len(A) == 2 and len(A[0]) == 2:
                val = A[0][0] * A[1][1] - 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 = determHitung(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
```

```
>>> det(m3)
-2
>>> det(m4)
-2
>>>
```

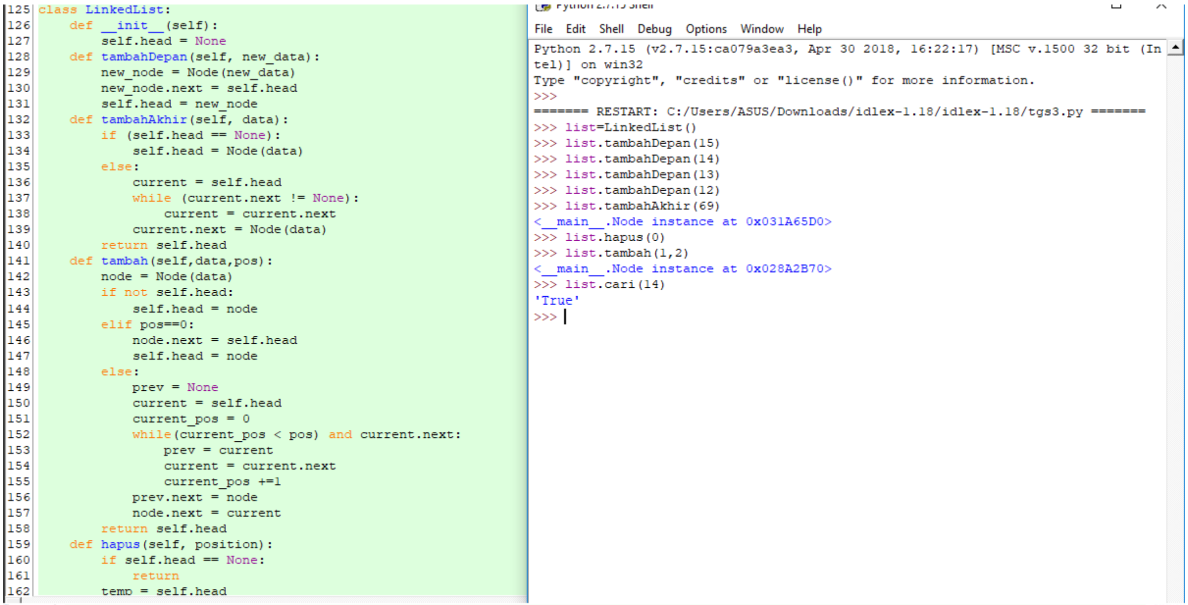
D. FUNGSI MENHITUNG DETERMINAN MATRIX

2. TERKAIT MATRIX DAN LIST COMPREHENSION



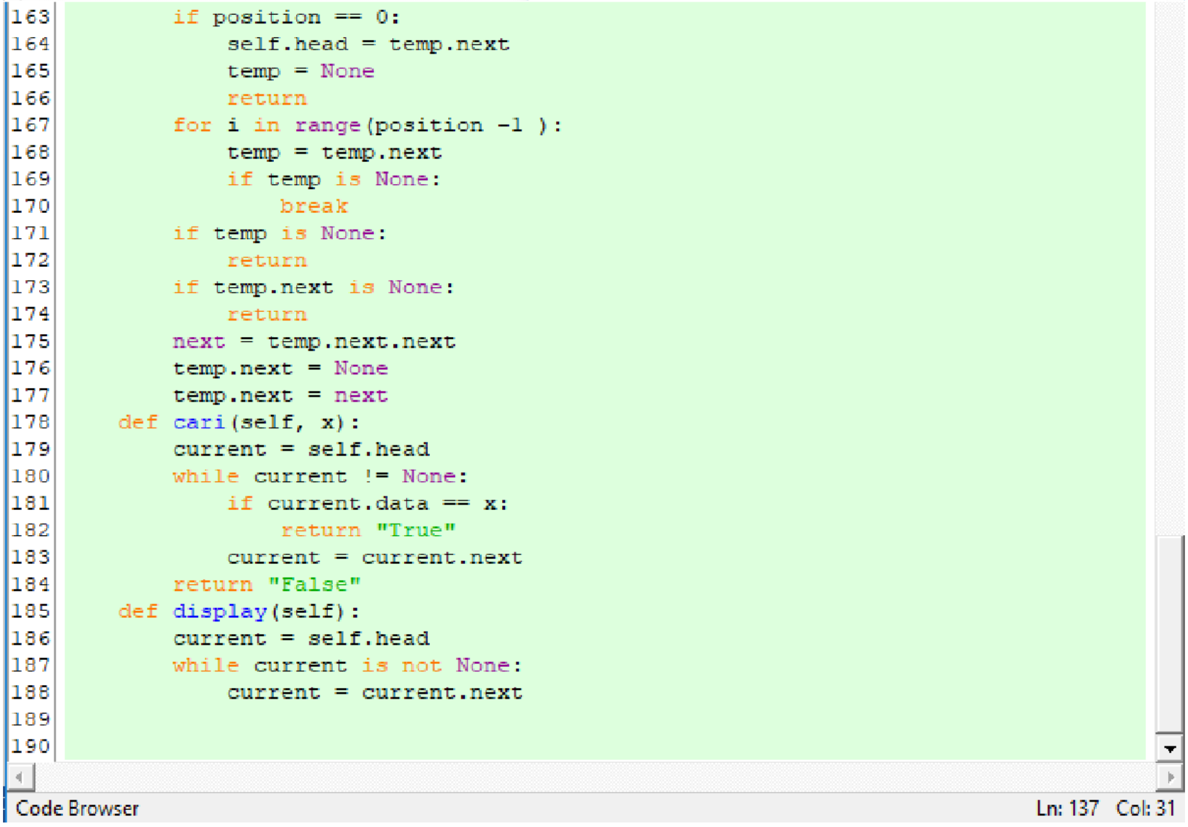
```
def buatNol(n,m=None):
    if(m==None):
        m=n
    print("membuat matriks 0 dengan ordo "+str(n)+"x"+str(m))
    print([[0 for j in range(m)] for i in range(n)])

def buatIden(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)])
```

```
>>> buatNol(2,3)
membuat matriks 0 dengan ordo 2x3
[[0, 0, 0], [0, 0, 0]]
>>> buatIden(3)
membuat matriks identitas dengan ordo3x3
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
>>>
```



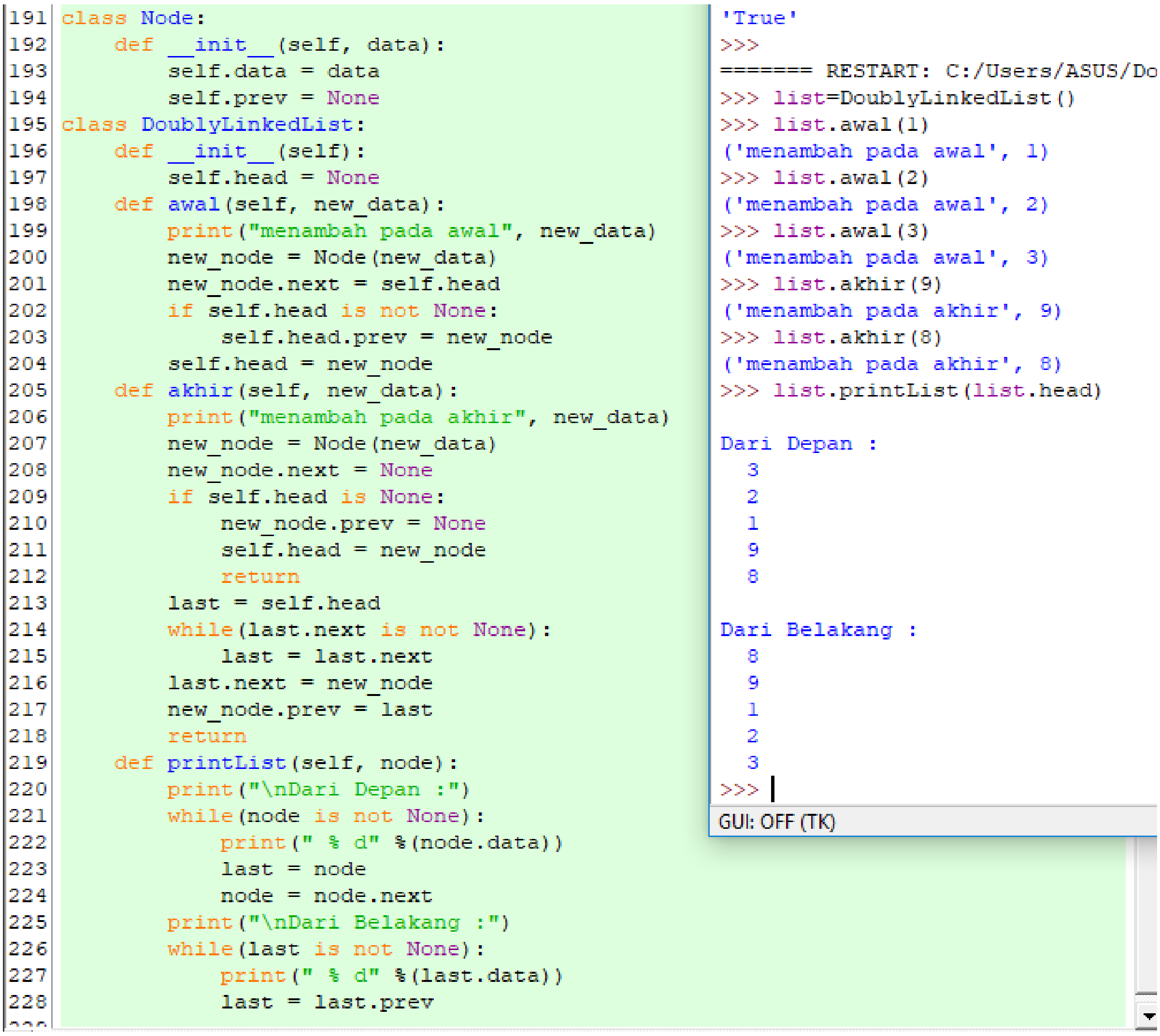
```python
class LinkedList:
    def __init__(self):
        self.head = None
    def tambahDepan(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node
    def tambahAkhir(self, data):
        if (self.head == None):
            self.head = Node(data)
        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, position):
        if self.head == None:
            return
        temp = self.head
```

```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:22:17) [MSC v.1500 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
======= RESTART: C:/Users/ASUS/Downloads/idlex-1.18/idlex-1.18/tgs3.py =======
>>> list=LinkedList()
>>> list.tambahDepan(15)
>>> list.tambahDepan(14)
>>> list.tambahDepan(13)
>>> list.tambahDepan(12)
>>> list.tambahAkhir(69)
<__main__.Node instance at 0x031A65D0>
>>> list.hapus(0)
>>> list.tambah(1,2)
<__main__.Node instance at 0x028A2B70>
>>> list.cari(14)
'True'
>>> 
```



```python
        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 cari(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:
            current = current.next
```

3. TERKAIT LINKED LIST



```python
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
```

```
'True'
>>>
======= RESTART: C:/Users/ASUS/Do
>>> list=DoublyLinkedList()
>>> list.awal(1)
('menambah pada awal', 1)
>>> list.awal(2)
('menambah pada awal', 2)
>>> list.awal(3)
('menambah pada awal', 3)
>>> list.akhir(9)
('menambah pada akhir', 9)
>>> list.akhir(8)
('menambah pada akhir', 8)
>>> list.printList(list.head)

Dari Depan :
  3
  2
  1
  9
  8

Dari Belakang :
  8
  9
  1
  2
  3
>>> 
```

4. TERKAIT DOUBLY LINKED LIST