Nama : Tomy Satmoko Aji

NIM : L200180057

Kelas : C

MODUL 3

Latihan 3.1 Membuat matrix dengan ordo 2 x 2.

```
Latihan 3.1 Membuat Matrix 2 x 2
>>> A = [ [2,3], [5,7] ]
>>> A [0][1]
3
>>> A[1][1]
7
>>> A[0][0]
2
>>> A[1][0]
5
>>>
```

Latihan 3.2 Membuat matrix 3 x 3 berisi 0 semua.

```
Latihan 3.2 Membuat Matrix 3 x 3 berisi 0
>>> B = [ [0 for j in range(3)] for i in range(3)]
>>> B
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]
>>>
```

List Comprehension

- Membuat list kuadratbilangandari 0 sampai 6

```
>>> [x**2 for x in range(0, 7)]
[0, 1, 4, 9, 16, 25, 36]
```

- Membuat list yang berisi tuple pasanganbilangandankuadratnya, dari 0 sampai 6

```
>>> [(x, x**2) for x in range(7)]
[(0, 0), (1, 1), (2, 4), (3, 9), (4, 16), (5, 25), (6, 36)]
```

- Membuat list kuadratbilangangenapantara 0 sampai 15

```
>>> [x**2 for x in range(15) if x%2 == 0]
[0, 4, 16, 36, 64, 100, 144, 196]
```

- Membuat list sepanjang 5 elemen yang berisibilangan 3

```
>>> [3 for i in range(5)]
[3, 3, 3, 3, 3]
```

- Membuat list sepanjang 3 elemen yang berisi list sepanjang 3 elemenangka 0

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

- Membuat matrix identitas 3 x 3

```
>>> [[1 if j==i else 0 for j in range (3)] for i in range (3)]
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
```

- Membuat list yang berisihuruf vocal suatu string

```
>>> d = 'Yogyakarta dan Surakarta'
>>> [x for x in d if x in 'aiueoAIUEO']
['o', 'a', 'a', 'a', 'a', 'u', 'a', 'a', 'a']
```

- Membuat list bilangangenapdari 20 sampai 50

```
>>> [x for x in range(20,50) if apakahGenap(x)]
[20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48]
>>>
```

Linked Structures

- Linked List

```
class Node(object) :
    '''Sebuah simpul di linked list'''
    def __init__(self, data, next=None):
        self.data = data
        self.next = next
```

Saatdijalankan di Python Shell :

```
>>> a = Node(11)
>>> b = Node(52)
>>> c = Node(18)
>>> d = Node(7)
>>> e = Node(31)
>>> a.next = b
>>> b.next = c
>>> c.next = d
>>> d.next = e
>>> print(a.data)
11
>>> print(a.next.data)
52
>>> print(a.next.next.data)
18
>>> print(c.next.data)
7
>>> print(b.next.next.next.data)
31
```

- Mengunjungisetiapsimpuldaridepan

```
class Node(object) :
    '''Sebuah simpul di linked list'''
    def __init__(self, data, next=None):
        self.data = data
        self.next = next
    def kunjungi(self, head):
        curNode = head
        while curNode is not None:
            print(curNode.data)
            curNode = curNode.next
```

Saatdijalankan di Python Shell :

```
>>> a = Node(11)
>>> b = Node(52)
>>> c = Node(18)
>>> d = Node(7)
>>> e = Node(31)
>>> a.next = b
>>> b.next = c
>>> c.next = d
>>> d.next = e
>>> a.kunjungi(a)
11
52
18
7
31
>>> d.kunjungi(d)
7
31
```

- Advanced Linked List

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

Saatdijalankan di Python Shell :

```
>>> a = DNode(11)
>>> b = DNode(52)
>>> c = DNode(18)
>>> d = DNode(7)
>>> e = DNode(31)
>>> a.next = b
>>> b.next = c
>>> c.next = d
>>> e.prev = d
>>> c.prev = b

>>> print(a.next.data)
52
>>> print(e.prev.data)
7
```

.

3.4 Soal-soaluntukMahasiswa

1. Membuattipe data sebuah matrix yang berisiangka-angka. Membuatfungsi :
   a. Untukmemastikanbahwaisidanukuran matrix-nyakonsisten

```
def cekKonsisten(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")
```

Saat di jalankan di Python Shell, hasilnya :

```
NO 1 A
>>> a = [[1,2],[3,4]]
>>> b = [[5,6],[7,8]]
>>> c = [[12,3,"x","y"],[12,33,4]]
>>> d = [[3,4],[2,4],[1,5]]
>>> e = [[5,6,7],[7,8,9]]
>>> f = [[1,2,3],[4,5,6],[7,8,9]]
>>> cekKonsisten(a)
matriks konsisten
>>> cekKonsisten(f)
matriks konsisten
>>> cekKonsisten(d)
matriks konsisten
>>> cekKonsisten(c)
matrik tidak konsisten
>>>
```

b. Untukmengambilukuranmatrixnya

```
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")

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))
```

Saatdijalankan di Python Shell, hasilnya :

```
>>> a = [[1,2],[3,4]]
>>> b = [[5,6],[7,8]]
>>> c = [[12,3,"x","y"],[12,33,4]]
>>> d = [[3,4],[2,4],[1,5]]
>>> e = [[5,6,7],[7,8,9]]
>>> f = [[1,2,3],[4,5,6],[7,8,9]]
>>> cekInt(a)
semua isi matriks adalah angka
>>> cekInt(c)
tidak semua isi matriks adalah angka
>>> ordo(f)
mempunyai ordo 3x3
>>> ordo(c)
mempunyai ordo 2x3
>>> ordo(d)
mempunyai ordo 3x2
```

c. Untukmenjumlahkandua matrix

```
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))

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")
```

Saatdijalankan di Python Shell :

```
>>> a = [[1,2],[3,4]]
>>> b = [[5,6],[7,8]]
>>> c = [[12,3,"x","y"],[12,33,4]]
>>> d = [[3,4],[2,4],[1,5]]
>>> e = [[5,6,7],[7,8,9]]
>>> f = [[1,2,3],[4,5,6],[7,8,9]]
>>> ordo(a)
mempunyai ordo 2x2
>>> ordo(b)
mempunyai ordo 2x2
>>> jumlah(a,b)
ukuran sama
[[6, 8], [10, 12]]
>>> ordo(d)
mempunyai ordo 3x2
>>> ordo(e)
mempunyai ordo 2x3
>>> jumlah(d,e)
ukuran beda
```

d. Untukmengalikandua matrix

```
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")
```

Saatdijalankan di Python Shell :

```
>>> x = [[1,2,3],[1,2,3]]
>>> z = [[1],[2],[3]]
>>> a = [[1,2],[3,4]]
>>> d = [[3,4],[2,4],[1,5]]
>>> f = [[1,2,3],[4,5,6],[7,8,9]]
>>> b = [[5,6],[7,8]]
>>> kali(x,z)
bisa dikalikan
[[14], [14]]
>>> kali(a,f)
tidak memenuhi syarat
>>> kali(a,b)
bisa dikalikan
[[19, 22], [43, 50]]
```

e. Untukmenghitungdeterminansebuah matrix bujursangkar

```
def determHitung(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
```

Saat di jalankan di Python Shell, hasilnya :

```
>>> z = [[3,1],[2,5]]
>>> x = [[1,2,1],[3,3,1],[2,1,2]]
>>> v = [[1,-2,0,0],[3,2,-3,1],[4,0,5,1],[2,3,-1,4]]
>>> r = [[10,23,45,12,13],[1,2,3,4,5],[1,2,3,4,6],[4,2,3,4,8],[1,4,5,6,10]]
>>> d = [[3,4],[2,4],[1,5]]
>>> f = [[1,2,3],[4,5,6],[7,8,9]]
>>> e = [[5,6,7],[7,8,9]]
>>> print(determHitung(d))
tidak bisa dihitung determinan, bukan matrix bujursangkar
>>> print(determHitung(x))
-6
>>> print(determHitunng(z))
Traceback (most recent call last):
  File "<pyshell#59>", line 1, in <module>
    print(determHitunng(z))
NameError: name 'determHitunng' is not defined
>>> print(determHitung(z))
13
>>> print(determHitung(r))
330
```

2. Membuatfungsi :
    a. Untukmembangkitkan matrix berisi 0 semua, dengandiberikanukurannya. Pemanggilan :`buatNol(m,n)`dan`buatNol(m)`. pemanggilandengancaraterakhirakanmemberikan matrix bujursangkarukuran`m x m`.

```
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)])
```

Saatdijalankan di Python Shell, hasilnya :

```
>>> buatNol(2,3)
membuat matriks 0 dengan ordo 2x3
[[0, 0, 0], [0, 0, 0]]
>>> buatNol(4)
membuat matriks 0 dengan ordo 4x4
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
>>>
```

    b. Untukmembangkitkan matrix identitas, dengandiberikanukurannya. Pemanggilan :`buatIdentitas()`

```
def buatIdentitas(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)])
```

Saatdijalankan di Python Shell, hasilnya :

```
>>> buatIdentitas(3)
membuat matriks identitas dengan ordo3x3
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
>>> buatIdentitas(4)
membuat matriks identitas dengan ordo4x4
[[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]
```

3. Terkait linked list, membuatfungsiuntuk :
    a. Mencaridata yang isinyatertentu :`cari(head, yang_dicari)`

```
def cari(head, cari):
    curr = head
    while curr != None:
        if curr.data == cari:
            print("Data ditemukan!")
        else:
            print("Check data!")
        curr = curr.nextNode
```

Saat di jalankan di python shell, hasilnya :

```
>>> a = Node(21)
>>> b = Node(89)
>>> c = Node(23)
>>> a.cari(23)
Check data!
>>> c.cari(23)
Data ditemukan!
```

    b. Menambahsuatusimpul di awal :`tambahDepan(head)`

```
def tambahDepan(head):
    newNode = Node(1)
    newNode.nextNode = head
    head = newNode
    return head
```

    c. Menambahsuatusimpul di akhir :`tambahAkhir(head)`

```
def tambahAkhir(head):
    curr = head
    while curr is not None:
        if curr.nextNode == None:
            newNode = Node(25)
            curr.nextNode = newNode
            return curr
        else:
            pass
        curr = curr.nextNode
    return curr
```

    d. Menyisipkansuatusimpul di mana saja :`tambah(head, postion)`

```
def tambah(head, posisi):
    newNode = Node(8)
    newNode.nextNode = posisi.nextNode
    posisi.nextNode = newNode
    head.head = posisi
    return head
```

e. Menghapussuatusimpul di awal, di akhir, atau di mana saja :`hapus(posisi)`

```
def hapus(head, posisi):
    curr = head
    while curr != None:
        if curr.nextNode.data == posisi:
            curr.nextNode = curr.nextNode.nextNode
            return curr
        else:
            pass
        curr = curr.nextNode
    return curr
```

4. Terkait doubly linked list, membuatfungsiuntuk :
   a. Mengunjungidanmencetak data tiapsimpuldaridepandandaribelakang

```
class doubly_linked():
    def __init__(self, Data, Next=None, Prev=None):
        self.Data = Data
        self.Next = Next
        self.Prev = Prev

    def mencetak():
        curr = head
        while curr != None:
            print(curr.Data)
            if curr.Next == None:
                curr = curr
                break
            else:
                curr = curr.Next
        print("\n")
        while curr != None:
            print(curr.Data)
            curr = curr.Prev
```

   b. Menambahsuatusimpul di awal

```
def simpulAwal(head):
    newNode = doubly_linked(25)
    newNode.Next = head
    head.Prev = newNode
    head =newNode
    return head
```

c. Menambahsuatusimpul di akhir

```
def simpulAkhir(head):
    curr = head
    while curr != None:
        if curr.Next == None:
            newNode = doubly_linked(365)
            curr.Next = newNode
            newNode.Prev = curr
            return curr
        else:
            pass
        curr = curr.Next
    return curr
```