VIRLIANA AR
L200180017
A

MODUL 3

Collections,Arrays, And Linked Structures

Soal – Soal Untuk Mahasiswa

1. Terkait array 2 dimensi

```
a = [[2,4],
     [6,21]]
b = [[2,67],
     [2,9],
     [1,5,"b",9]]
c = [[3,5,7],[21,87,9]]
d = [[1,2,3],[21,90,3],[6,5,78]]
e = [[1,2],[43,7]]
##Memastikan isi dan ukuran matrixnya konsisten
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("Matrix konsisten")
    else:
        print("Matrix tidak konsisten")

print("NOMOR 1A")
cekkonsisten(a)
cekkonsisten(b)
cekkonsisten(c)
cekkonsisten(d)
def cekinteger(n):
    x = 0
    y = 0
    for i in n:
        for j in i:
            y += 1
            if(str(j).isdigit() == False):
                print("Tidak semua isi matrix angka")
                break
            else:
                x += 1
    if (x == y):
        print("Semua isi matrix adalah angka")
cekinteger(a)
cekinteger(b)
```

Ln: 136 Col: 0

File Edit Format Run Options Window Help

```
##Mengambil ukuran matrix
def cekordo(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))

print("NOMOR 1B")
cekordo(a)
cekordo(b)
cekordo(c)
cekordo(d)
##Menjumlahkan 2 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")

print("NOMOR 1C")
jumlah(a,b)
jumlah(a,e)
##Mengkalikan 2 matrix
def kali(n,m):
```

Ln: 136 Col: 0

```
##Mengkalikan 2 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")
print("NOMOR 1D")
kali(a,e)
kali(b,d)
```

```
##Menghitung determinan
def determinan(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 = determinanHitung(As)
                total += sign * A[0][fc] * sub_det
        else:
            return """tidak bisa dihitung determinannya,
                    karena bukan matrix bujursangkar"""
    else:
        return """tidak bisa dihitung determinannya,
                karena bukan matrix bujursangkar"""
    return total
print("NOMOR 1E")
print(determinan(a))
print(determinan(b))
```

Ln: 133 Col: 16

```
NOMOR 1A
Matrix konsisten
Matrix tidak konsisten
Matrix konsisten
Matrix konsisten
Semua isi matrix adalah angka
Tidak semua isi matrix angka
NOMOR 1B
Mempunyai ordo2x2
Mempunyai ordo3x4
Mempunyai ordo2x3
Mempunyai ordo3x3
NOMOR 1C
ukuran beda
ukuran sama
[[3, 6], [49, 28]]
NOMOR 1D
bisa dikalikan
[[174, 32], [909, 159]]
tidak memenuhi syarat
NOMOR 1E
18
tidak bisa dihitung determinannya,karena bukan matrix bujursangkar
>>>
```

2. Terkait matrix dan list comprehension

File Edit Format Run Options Window Help

```
## 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 x in range(m)] for y in range(n)])
print("NOMOR 2A")
buatNol(3)
buatNol(4,3)

## 2b
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)])

print("NOMOR 2B")
buatIdentitas(5)
buatIdentitas(3)
```

Ln: 6 Col: 51

```
NOMOR 2A
Membuat matriks 0 dengan ordo 3x3
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]
Membuat matriks 0 dengan ordo 4x3
[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0]]
NOMOR 2B
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 ordo3x3
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
>>>
```

Ln: 15 Col: 4

3. Terkait linked list

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 pushAwal(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node
    def pushAkhir(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):
```

Ln: 84 Col: 0

File Edit Format Run Options Window Help

```
    def hapus(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 -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:
            print(current.data, end = ' ')
            current = current.next

llist = LinkedList()
llist.pushAwal(12)
llist.pushAwal(13)
llist.pushAwal(14)
llist.pushAwal(15)
llist.pushAwal(3)
llist.pushAwal(17)
```

Ln: 84 Col: 0

```
llist.pushAwal(17)
llist.pushAkhir(18)
llist.hapus(0)
llist.tambah(1,4)
print(llist.cari(13))
print(llist.cari(16))
llist.display()
```

Ln: 84 Col: 0

```
True
False
3 15 14 13 1 12 18
>>>
```

Ln: 8

4. Terkait doubly linked list

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 menambahAwal(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 menambahAkhir(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.menambahAwal(2)
```

Ln: 38 Col: 0

```
llist = DoublyLinkedList()
llist.menambahAwal(2)
llist.menambahAwal(1)
llist.menambahAkhir(3)
llist.menambahAkhir(4)
llist.printList(llist.head)
```

Ln: 38 Col: 0

```
menambah pada awal 2
menambah pada awal 1
menambah pada akhir 3
menambah pada akhir 4

Dari Depan :
  1
  2
  3
  4

Dari Belakang :
  4
  3
  2
  1
>>>
```

Ln: 21 Col: 4