PRACTICUM REPORT
ALGORITHM AND DATA STRUCTURES
MODUL 3 : COLLECTIONS, ARRAYS, AND LINKED STRUCTURES

Disusun Oleh :

ONIC AGUSTINO

L200234275

X

INFORMATICS ENGINEERING

FACULTY OF COMMUNICATION AND INFORMATICS

UNIVERSITAS MUHAMMADIYAH SURAKARTA

YEARS 2024/2025

1.11 Questions

1. Regarding two-dimensional arrays, we will create a matrix data type that contains numbers. For that, create functions

   • to ensure that the contents and size of the matrix are consistent (because each member of the outer list may have a different size, and may even be a different type!),

   • to take the matrix size,

   • to add two matrices (make sure the sizes match),

   • to multiply two matrices (make sure the sizes match),

   • to calculate the determinant of a square matrix.

Module_3 > Question > 1.py > add_matrices

```python
def is_matrix_consistent(matrix):
    """
    Memastikan bahwa isi dan ukuran matriks konsisten.
    """
    if not matrix or not isinstance(matrix, list):
        return False
    row_length = len(matrix[0])
    for row in matrix:
        if not isinstance(row, list) or len(row) != row_length:
            return False
    return True

def matrix_size(matrix):
    """
    Mengambil ukuran matriks.
    """
    if not is_matrix_consistent(matrix):
        raise ValueError("Matriks tidak konsisten")
    return len(matrix), len(matrix[0])

def add_matrices(matrix1, matrix2):
    """
    Menambahkan dua matriks (pastikan ukurannya sama).
    """
    if not (is_matrix_consistent(matrix1) and is_matrix_consistent(matrix2)):
        raise ValueError("Salah satu atau kedua matriks tidak konsisten")

    rows1, cols1 = matrix_size(matrix1)
    rows2, cols2 = matrix_size(matrix2)

    if rows1 != rows2 or cols1 != cols2:
        raise ValueError("Ukuran matriks tidak sama")

    result = []
    for i in range(rows1):
        row = []
        for j in range(cols1):
            row.append(matrix1[i][j] + matrix2[i][j])
        result.append(row)

    return result
```

```python
def multiply_matrices(matrix1, matrix2):
    """
    Mengalikan dua matriks (pastikan ukurannya sesuai).
    """
    if not (is_matrix_consistent(matrix1) and is_matrix_consistent(matrix2)):
        raise ValueError("Salah satu atau kedua matriks tidak konsisten")

    rows1, cols1 = matrix_size(matrix1)
    rows2, cols2 = matrix_size(matrix2)

    if cols1 != rows2:
        raise ValueError("Ukuran matriks tidak sesuai untuk perkalian")

    result = [[0 for _ in range(cols2)] for _ in range(rows1)]

    for i in range(rows1):
        for j in range(cols2):
            for k in range(cols1):
                result[i][j] += matrix1[i][k] * matrix2[k][j]

    return result

def determinant(matrix):
    """
    Menghitung determinan dari matriks persegi.
    """
    if not is_matrix_consistent(matrix):
        raise ValueError("Matriks tidak konsisten")

    rows, cols = matrix_size(matrix)

    if rows != cols:
        raise ValueError("Matriks bukan persegi")

    if rows == 1:
        return matrix[0][0]

    if rows == 2:
        return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0]

    det = 0
```

```python
    for c in range(cols):
        sub_matrix = [row[:c] + row[c+1:] for row in matrix[1:]]
        det += ((-1) ** c) * matrix[0][c] * determinant(sub_matrix)

    return det

# Contoh penggunaan
matrix1 = [
    [1, 2],
    [3, 4]
]

matrix2 = [
    [5, 6],
    [7, 8]
]

print("Matrix 1:", matrix1)
print("Matrix 2:", matrix2)
print("Penjumlahan Matriks:", add_matrices(matrix1, matrix2))
print("Perkalian Matriks:", multiply_matrices(matrix1, matrix2))
print("Determinan Matriks 1:", determinant(matrix1))
```

Picture 1.1 the code 1.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_3/Question/1.py"
Matrix 1: [[1, 2], [3, 4]]
Matrix 2: [[5, 6], [7, 8]]
Penjumlahan Matriks: [[6, 8], [10, 12]]
Perkalian Matriks: [[19, 22], [43, 50]]
Determinan Matriks 1: -2
```

Picture 1.2 the output 1.py

2. Regarding the matrix and list comprehension, create (using list comprehension) functions

• to generate a matrix containing all zeros, given its size. calling: buatNol(m,n) and buatNol(m). Calling the latter method will provide a square matrix of sizes m × m.

• to generate an identity matrix, given its size. Call: buatIdentitas(m.

Module_3 > Question > 2.py > ...

```python
def buatNol(m, n=None):
    """
    Menghasilkan matriks yang berisi semua nol.
    Jika hanya satu argumen diberikan, menghasilkan matriks persegi m x m.
    """
    if n is None:
        n = m
    return [[0 for _ in range(n)] for _ in range(m)]

def buatIdentitas(m):
    """
    Menghasilkan matriks identitas berukuran m x m.
    """
    return [[1 if i == j else 0 for j in range(m)] for i in range(m)]

# Contoh penggunaan
print("Matriks Nol 3x3:")
print(buatNol(3))
print("\nMatriks Nol 2x4:")
print(buatNol(2, 4))
print("\nMatriks Identitas 3x3:")
print(buatIdentitas(3))
```

Picture 2.1 the code 2.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_3/Question/2.py"
Matriks Nol 3x3:
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]

Matriks Nol 2x4:
[[0, 0, 0, 0], [0, 0, 0, 0]]

Matriks Identitas 3x3:
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
```

Picture 2.2 the output 2.py

3. Regarding the linked list, create a function for

- search for data with certain contents: cari(head,yang dicari)
- add a node at the beginning: tambahDepan(head)
- add a node at the end: tambahAkhir(head)
- insert a node anywhere: tambah(head,posisi)
- delete a node at the beginning, at the end, or anywhere: hapus(posisi)

Module_3 > Question > 3.py > ...

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

class LinkedList:
    def __init__(self):
        self.head = None

    def cari(self, yang_dicari):
        current = self.head
        while current is not None:
            if current.data == yang_dicari:
                return True
            current = current.next
        return False

    def tambahDepan(self, data):
        new_node = Node(data)
        new_node.next = self.head
        self.head = new_node

    def tambahAkhir(self, data):
        new_node = Node(data)
        if self.head is None:
            self.head = new_node
            return
        last = self.head
        while last.next:
            last = last.next
        last.next = new_node

    def tambah(self, data, posisi):
        new_node = Node(data)
        if posisi == 0:
            new_node.next = self.head
            self.head = new_node
            return
        current = self.head
        for _ in range(posisi - 1):
            if current is None:
                raise IndexError("Posisi di luar jangkauan")
```

```
43              current = current.next
44          new_node.next = current.next
45          current.next = new_node
46
47      def hapus(self, posisi):
48          if self.head is None:
49              raise IndexError("List kosong")
50          if posisi == 0:
51              self.head = self.head.next
52              return
53          current = self.head
54          for _ in range(posisi - 1):
55              if current.next is None:
56                  raise IndexError("Posisi di luar jangkauan")
57              current = current.next
58          if current.next is None:
59              raise IndexError("Posisi di luar jangkauan")
60          current.next = current.next.next
61
62      def cetak(self):
63          current = self.head
64          while current:
65              print(current.data, end=" -> ")
66              current = current.next
67          print("None")
68
69  # Contoh penggunaan
70  ll = LinkedList()
71  ll.tambahDepan(3)
72  ll.tambahDepan(2)
73  ll.tambahDepan(1)
74  ll.cetak()
75  ll.tambahAkhir(4)
76  ll.cetak()
77  ll.tambah(1.5, 1)
78  ll.cetak()
79  ll.hapus(1)
80  ll.cetak()
81  print(ll.cari(3))
82  print(ll.cari(5))
```

Picture 3.1 the code 3.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_3/Question/3.py"
1 -> 2 -> 3 -> None
1 -> 2 -> 3 -> 4 -> None
1 -> 1.5 -> 2 -> 3 -> 4 -> None
1 -> 2 -> 3 -> 4 -> None
True
False
```

Picture 3.2 the output 3.py

4. Regarding the doubly linked list, create a function for

• visit and print data for each node from the front and from the back.

• add a node at the beginning

• add a node at the end

Module_3 > Question > 4.py > DoublyLinkedList

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

class DoublyLinkedList:
    def __init__(self):
        self.head = None
        self.tail = None

    def cetakDepan(self):
        current = self.head
        while current:
            print(current.data, end=" <-> ")
            current = current.next
        print("None")

    def cetakBelakang(self):
        current = self.tail
        while current:
            print(current.data, end=" <-> ")
            current = current.prev
        print("None")

    def tambahDepan(self, data):
        new_node = Node(data)
        if self.head is None:
            self.head = self.tail = new_node
        else:
            new_node.next = self.head
            self.head.prev = new_node
            self.head = new_node

    def tambahAkhir(self, data):
        new_node = Node(data)
        if self.tail is None:
            self.head = self.tail = new_node
        else:
            new_node.prev = self.tail
            self.tail.next = new_node
            self.tail = new_node
```

```
43
44    # Contoh penggunaan
45    dll = DoublyLinkedList()
46    dll.tambahDepan(3)
47    dll.tambahDepan(2)
48    dll.tambahDepan(1)
49    dll.cetakDepan()
50    dll.cetakBelakang()
51
52    dll.tambahAkhir(4)
53    dll.cetakDepan()
54    dll.cetakBelakang()
55
```

Picture 4.1 the code 4.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_3/Question/4.py"
1 <-> 2 <-> 3 <-> None
3 <-> 2 <-> 1 <-> None
1 <-> 2 <-> 3 <-> 4 <-> None
4 <-> 3 <-> 2 <-> 1 <-> None
```

Picture 4.2 the output 4.py