Total pages used in this question - 5

4/4/2022, 9:00 AM
MCA
SEM-I
OBJECT ORIENTED PROGRAMMING
Course Code - 22340101
Exam Roll no - 2123475700 9

(5) →

Ans -

```
class Matrix:
    """
    It will represent the objects for
    Matrix (m x n)
    """
    obj_count = 0
    def __init__(self, row, col):
        """ Perform the initialization of
            matrix object
        """
        Matrix.obj_count += 1
        self.nRow = row
        self.nCol = col
        self.__ele = []
        for i in range(row):
            temp = []
            for j in range(col):
                temp.append(0)
            self.__ele.append(temp)
```

By default we are creating Zero Matrix.

```
@property
def nRow(self):
    return self.__nRow

@nRow.setter
def nRow(self, row):
    if row < 0:
        row = 0
    self.__nRow = row

@property
def nCol(self):
    return self.__nCol

@nCol.setter
def nCol(self, col):
    if col < 0:
        col = 0
    self.__nCol = col
```

```
def setValue(self, i, j, value):
    if i < self.nRow and j < self.nCol:
        self.__ele[i][j] = value

    else:
        print("Invalid Index")


def getValue(self, i, j):
    if i < self.nRow and j < self.nCol:
        return self.__ele[i][j]

    else:
        print("Invalid Index")
        return None


def __add__(self, other):
    if type(other) == Matrix:
        if self.nRow == other.nRow and
                self.nCol == other.nCol:

            res = []
            for i in range(self.nRow):
                temp = []
                for j in range(self.nCol):
                    temp.append(self.__ele[i][j]
                                + other.__ele[i][j])
                res.append(temp)
```

```
        return res

    else:
        print("Matrices are incompatible")

def __sub__(self, other):
    if type(other) == Matrix:
        if self.nRow == other.nRow and
                self.nCol == other.nCol:
            res = []
            for i in range(self.nRow):
                temp = []
                for j in range(self.nCol):
                    temp.append(self.__ele[i][j]
                                - other.__ele[i][j])
                res.append(temp)
            return res
        else:
            print("Matrices are incompatible")

def __mul__(self, other):
    if type(other) == Matrix:
        if self.nCol == other.nRow:
            res = []
```

```
for i in range(self.nRow):
    temp = []
    for j in range(other.nCol):
        temp.append(0)
        for k in range(other.nRow):
            temp[j] += self.__ele[i][k] *
                       other.__ele[k][j]
    res.append(temp)
return res
else:
    print("Matrix are incompatible")

def __del__(self):
    Matrix.obj_count -= 1

def __str__(self):
    res = "Matrix 0 is \n"
    res += str(self.__ele)
    return res
```