Practicals\Q2\Q2.py

```
1 # 02
   # Create a class RELATION, use Matrix notation to represent a relation. Include member
 3
   # functions to check if the relation is Reflexive, Symmetric, Anti-symmetric, Transitive.
4
   # Using these functions check whether the given relation is: Equivalence or Partial Order
   # relation or None
 6
7
   a = [1,2,3]
   rel = [[1,1],[2,2],[3,3]]
 8
9
10
11
    class RELATION():
12
13
        def __init__(self,rel):
14
            self.rel = rel
            self.adjMat = self.makeAdjMat(rel)
15
16
        def makeAdjMat(self,rel):
17
18
            relation = self.rel
19
            matrix = []
            for i in range(len(a)):
20
21
                row = []
                for j in range(len(a)):
22
23
                    row.append(∅)
                matrix.append(row)
24
            for i in relation:
25
                row = a.index(i[0])
26
27
                col = a.index(i[1])
28
                matrix[row][col] = 1
29
            return matrix
30
31
32
33
        def isReflexive(self):
34
            for i in range(len(self.adjMat)):
35
                for j in range(len(self.adjMat)):
                    if i == j and self.adjMat[i][j] != 1:
36
37
                        return False
            return True
38
39
        def isSymmetric(self):
40
            for i in range(len(self.adjMat)):
41
                for j in range(len(self.adjMat)):
42
                    if self.adjMat[i][j] == 1 and self.adjMat[j][i] != 1:
43
44
                        return False
45
            return True
46
        #This function is just for demo purpose. It has very high time complexity hence it is
47
    needed to be changed
48
        def isTransitive(self):
            for i in range(len(rel.adjMat)):
49
50
                for j in range(len(rel.adjMat)):
                    for k in range(len(rel.adjMat)):
51
                         if rel.adjMat[i][j] == 1 and rel.adjMat[j][k] == 1 :
52
53
                             if rel.adjMat[i][k] != 1:
                                 return False
54
55
            return True
```

```
56
       def isEquivalence(self):
57
           if self.isReflexive() and self.isSymmetric() and self.isTransitive():
58
59
                return True
60
           return False
61
62
63
64
   rel = RELATION([[1,1],[2,2],[3,3],[2,1],[1,2],[2,3]])
65 print(rel.isReflexive())
66 print(rel.isSymmetric())
   print(rel.isTransitive())
67
68 print(rel.isEquivalence())
69
70
71
72
73
```