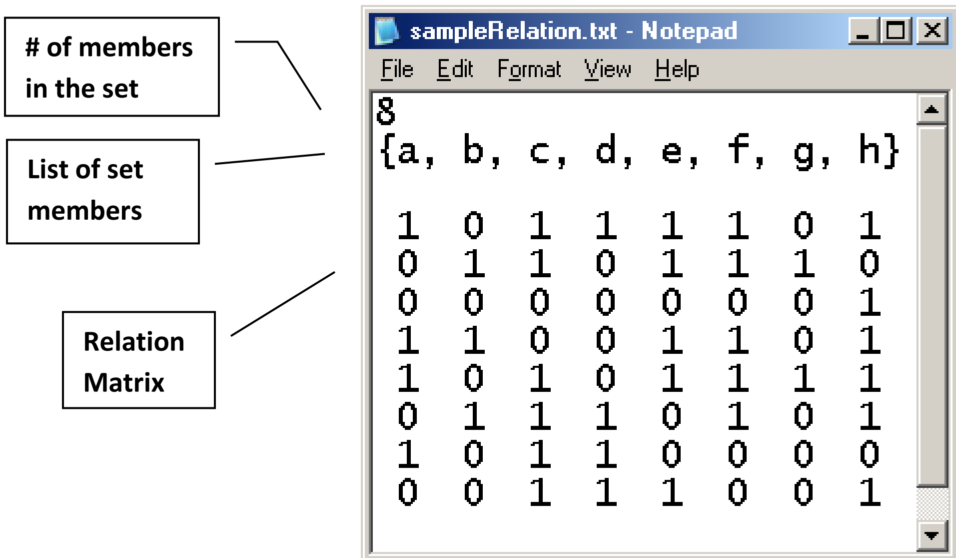
Programming Assignment

Relations Analyzer

Write a program that does the following:

(a) Takes as input a finite relation in matrix form. You could use the following notepad file as a guideline.



(b) Your program then does the following processing on this relation

BASIC FUNCTIONALITY: The program finds and states the following properties for the input relation.

(i) reflexivity (reflexive, irreflexive or none)

(ii) symmetry (symmetry, asymmetry, antisymmetry or none)

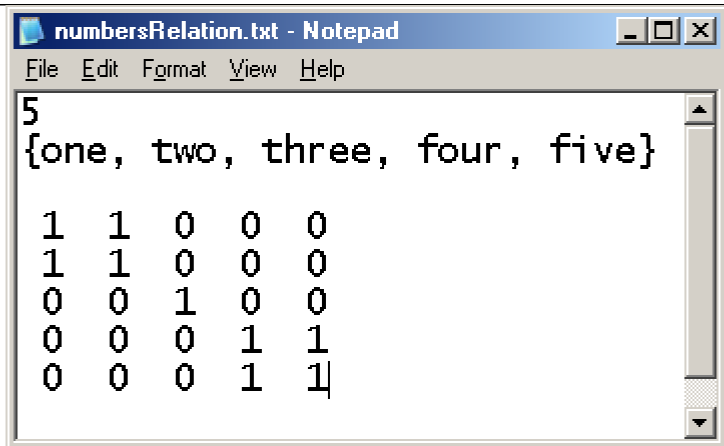
(iii) transitivity (transitive or not)

(iv) whether the input relation is an equivalence relation or a partial ordering.

A sample output could be:

```
The input relation is reflexive, antisymmetric
and transitive. Hence it is a partial ordering.
```

INTERMEDIATE FUNCTIONALITY: If the relation is an equivalence relation, the program finds and states all the equivalence classes in it. For example the following relation is an equivalence relation:



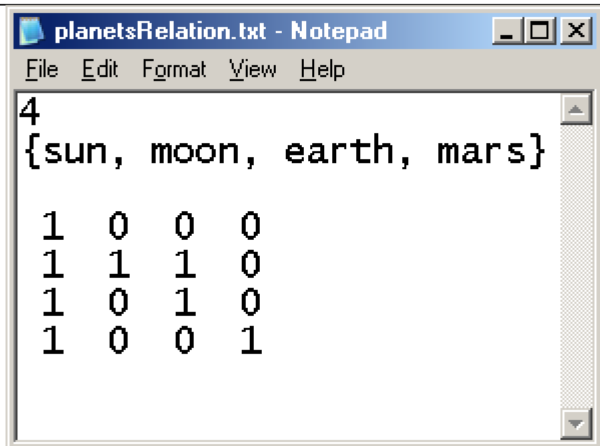

```
5
{one, two, three, four, five}

 1  1  0  0  0
 1  1  0  0  0
 0  0  1  0  0
 0  0  0  1  1
 0  0  0  1  1
```

The equivalence classes in this relation are

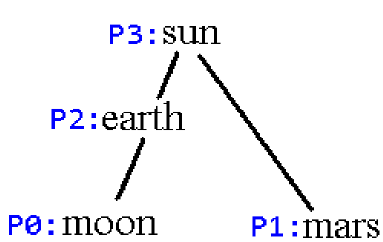
EQ1: {one, two}
EQ2: {three}
EQ3: {four, five}

If the relation is a partial ordering, the program finds and states all the elements in its Hasse Diagram. For example, the following relation is a partial ordering (Hasse Diagram also shown). Also find and print the maximal, minimal, greatest and the least elements.




```
4
{sun, moon, earth, mars}

 1  0  0  0
 1  1  1  0
 1  0  1  0
 1  0  0  1
```



The elements in the Hasse Diagram can be stated as follows:

P0: (-, {moon})
P1: (-, {mars})
P2: (P0, {earth})
P3: (P1 P2, {sun})

Each element is assigned a label P# and is represented as an ordered pair, where in an ordered pair (X, Y), X represents the preceding elements in the Hasse Diagram and Y represents the actual elements.