CentraleSupélec

MASTER DSBI
DECISION MODELING
PRACTICAL WORKS

PREFERENCES AS BINARY RELATIONS

1. For all the questions below, the binary relation is represented by a matrix given in an Excel file (.xls or .xlsx).
   - ⋆ You could implement a Python function converting this file to a .csv file.
   - ⋆ You could implement a Python function showing a graphical representation of this matrix by using appropriate libraries like networkx and matplotlib.

✓ 2. Build a Python function `CompleteCheck` testing if a binary relation is complete.

✓ 3. Build a Python function `ReflexiveCheck` testing if a binary relation is reflexive.

✓ 4. Build a Python function `AsymmetricCheck` testing if a binary relation is asymmetric.

✓ 5. Build a Python function `SymmetricCheck` testing if a binary relation is symmetric.

✓ 6. Build a Python function `AntisymmetricCheck` testing if a binary relation is antisymmetric.

✓ 7. Build a Python function `TransitiveCheck` testing if a binary relation is transitive.

✓ 8. Build a Python function `NegativetransitiveCheck` testing if a binary relation is negativetransitive.

✓ 9. Build a Python function `CompleteOrderCheck` testing if a binary relation is a complete order.

✓ 10. Build a Python function `CompletePreOrderCheck` testing if a binary relation is a complete pre-order.

✓ 11. Build a Python function `StrictRelation` returning the strict relation part of a binary relation.

✓ 12. Build a Python function `IndifferenceRelation` returning the indifference relation part of a binary relation.

13. Build a Python function `Topologicalsorting` returning a topological sorting of a binary relation.

Complete Order = complete, transitive, anti-symmetric

Complete Pre-Order = Complete check, transitive

Strict relation = creating a new matrix where asymmetry xRy and NOT yRxl later check for asymmetry

Indifference= create a new matrix where xRy and yRx so new matrix yRx and check for symmetry