CM 2602

ARTIFICIAL INTELLIGENCE

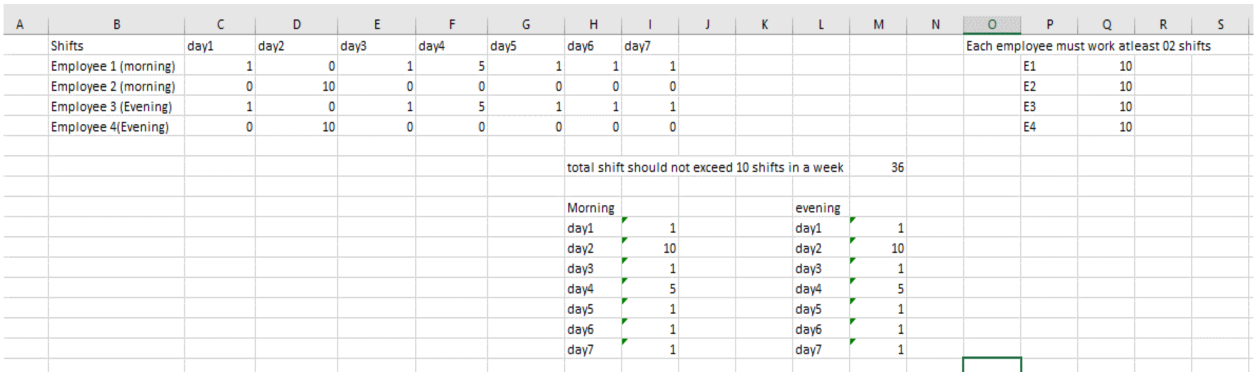
COURSE WORK

REPORT

SAMRITHAA JEEVARATNAM

20221530

QUESTION 01



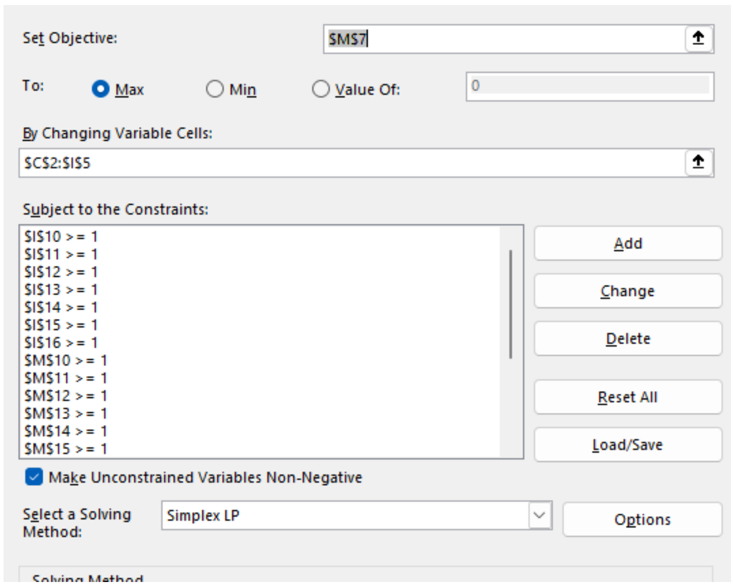
Shifts	day1	day2	day3	day4	day5	day6	day7
Employee 1 (morning)	1	0	1	5	1	1	1
Employee 2 (morning)	0	10	0	0	0	0	0
Employee 3 (Evening)	1	0	1	5	1	1	1
Employee 4(Evening)	0	10	0	0	0	0	0

Each employee must work atleast 02 shifts

E1	10
E2	10
E3	10
E4	10

total shift should not exceed 10 shifts in a week 36

Morning		evening	
day1	1	day1	1
day2	10	day2	10
day3	1	day3	1
day4	5	day4	5
day5	1	day5	1
day6	1	day6	1
day7	1	day7	1



Set Objective: $M$7

To: [✓ Max] Min Value Of: 0

By Changing Variable Cells: $C$2:$I$5

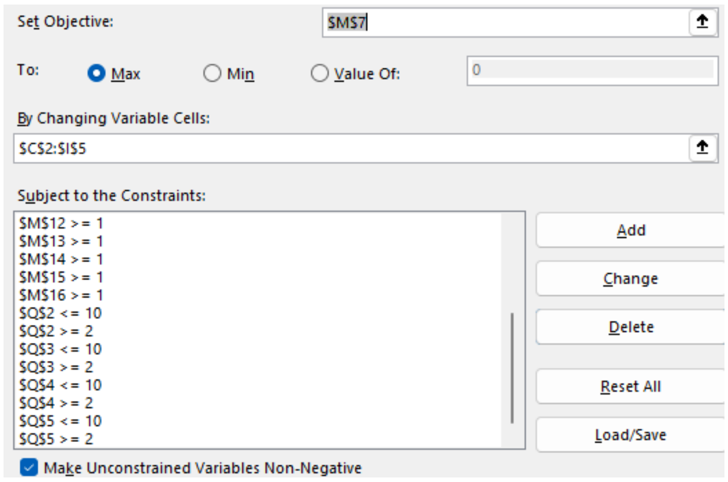
Subject to the Constraints:

$I$10 >= 1
$I$11 >= 1
$I$12 >= 1
$I$13 >= 1
$I$14 >= 1
$I$15 >= 1
$I$16 >= 1
$M$10 >= 1
$M$11 >= 1
$M$12 >= 1
$M$13 >= 1
$M$14 >= 1
$M$15 >= 1

[✓] Make Unconstrained Variables Non-Negative

Select a Solving Method: Simplex LP

Solving Method



Set Objective: $M$7

To: [✓ Max] Min Value Of: 0

By Changing Variable Cells: $C$2:$I$5

Subject to the Constraints:

$M$12 >= 1
$M$13 >= 1
$M$14 >= 1
$M$15 >= 1
$M$16 >= 1
$Q$2 <= 10
$Q$2 >= 2
$Q$3 <= 10
$Q$3 >= 2
$Q$4 <= 10
$Q$4 >= 2
$Q$5 <= 10
$Q$5 >= 2

[✓] Make Unconstrained Variables Non-Negative

An overview of the employee shift assignment report using the MS Excel solver

This Question looked into how to most effectively schedule a small team of four workers' shifts over the course of a week using MS Excel Solver. The main goal was to respect employee preferences and maintain a balance between work distribution and specific constraints.

Important limitations included:

Work requirements: Each employee must put in a minimum of two shifts per week.

Limit on total shifts: A maximum of ten shifts per week may be assigned.

Preferences of employees: two like working in the morning and the other two in the evening.

Shift coverage: A minimum of one employee must be assigned to each day's morning and evening shifts.

Employee assignments to shifts were represented by variables (1 for assigned, 0 for not assigned). Finding a solution that maximizes employee preference fulfilment while satisfying all constraints was the goal.

Every employee completed at least two shifts, with ten shifts allotted. Morning and evening shift preferences were recognized. Additionally, at least one worker was assigned to each shift.

Morning and evening shift preferences were identified. Additionally, at least one worker was assigned to each shift.

To sum up, solver helped to schedule the workforce while taking employee preferences into account and working under a variety of constraints.

QUESTION 02

a) Define the scope you attempt to cover from your knowledge model/ontology and propose relevant competency questions (at least 05 aspects). List all the sources you referred for the information gathering in the references section.

Design Parameters: Could you list the important design parameters and describe their function in maximising the functionality and performance of mechanical systems?

Compliance Standards: What role do compliance standards play in guaranteeing the dependability, safety, and quality of mechanical engineering designs?

Simulation: provide give situations of effective uses where simulation has made a big difference in mechanical system optimization.
Semantic Search: How can semantic search improve the mechanical engineering field's discovery of knowledge process?

Data Integration: In the context of knowledge discovery, how can data integration techniques be optimized to handle large and diverse datasets?

Knowledge Innovation: Could you give examples of how knowledge innovation has changed the industry fundamentally?

Quality Parameters: Could you elaborate on particular quality parameters that are frequently used when evaluating mechanical systems?

Industrial Compliance: Which important industrial compliance guidelines affect mechanical engineering quality control procedures?

Inspection Procedures: Could you give instances of how effective inspection procedures have been used to raise the quality of products?

Fundamental Concepts: In what ways do these basic ideas aid in the formation of knowledgeable experts in the field?

Educational Modules: How can thoughtfully created educational modules improve the mechanical engineering students' learning process?

Professional Training: How does professional development support the ongoing advancement of mechanical engineering professionals?

Environmental Management: Could you give a few examples of strategies that incorporate the concepts of environmental management into processes used in mechanical engineering.
Green Design Practice: How can mechanical engineering projects reduce their environmental impact by implementing green design principles?

SCOPE

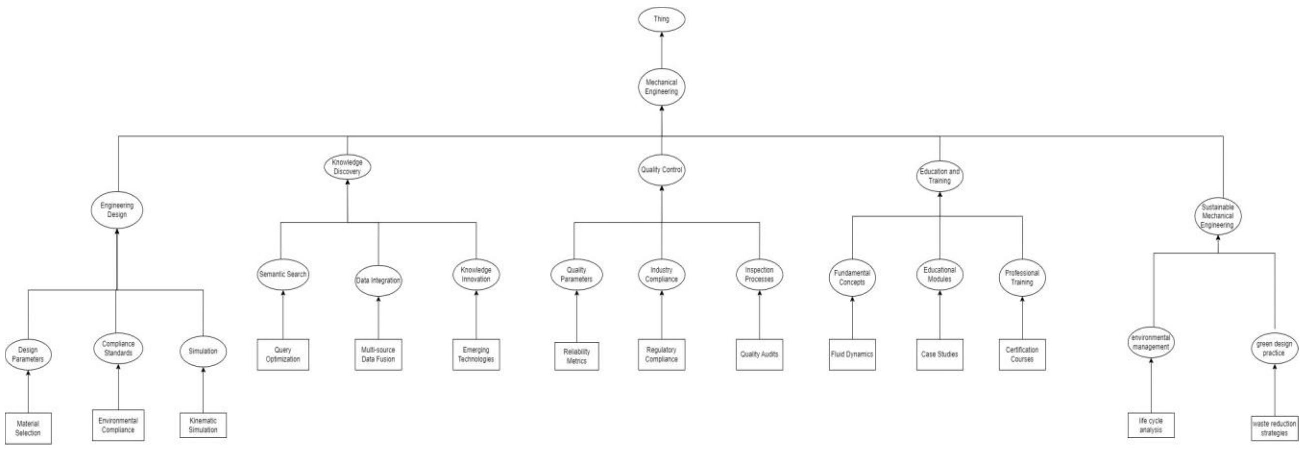
The main goal of the knowledge model/ontology that is being presented is to thoroughly describe and arrange the various aspects of mechanical engineering.

The model encompasses various subclasses, including Sustainable Mechanical Engineering, Knowledge Discovery, Engineering Design, Education and Training, and Quality Control.

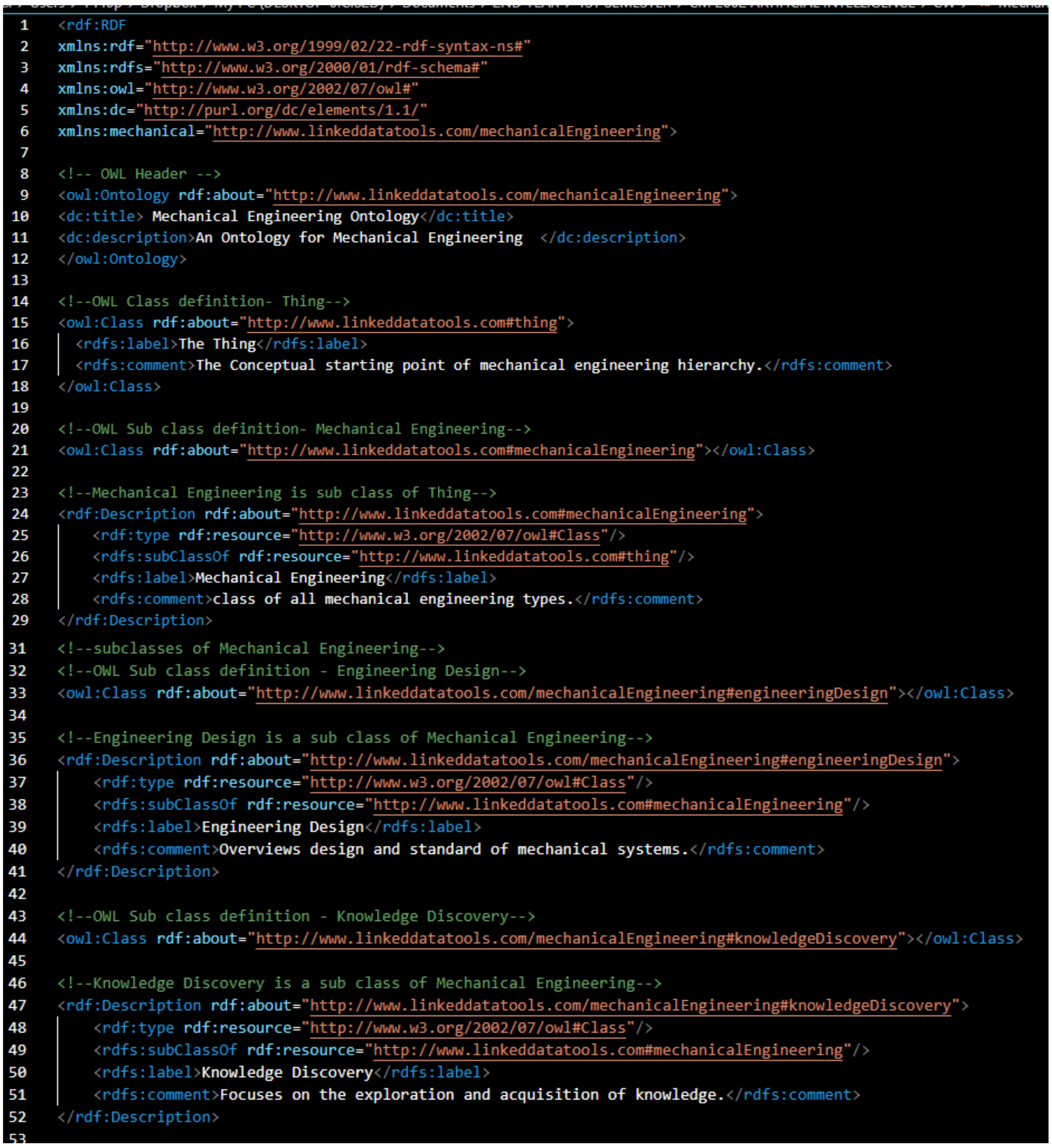
Its objective is to offer a systematic framework for understanding and handling the complex issues that are built into the field.

By exploring knowledge innovation, quality parameters, professional training, and the complexities of design parameters and compliance standards, among other topics, the ontology promotes a comprehensive understanding of mechanical engineering and helps students gain a deeper understanding of its conceptual base, real-world applications, and sustainable practices.

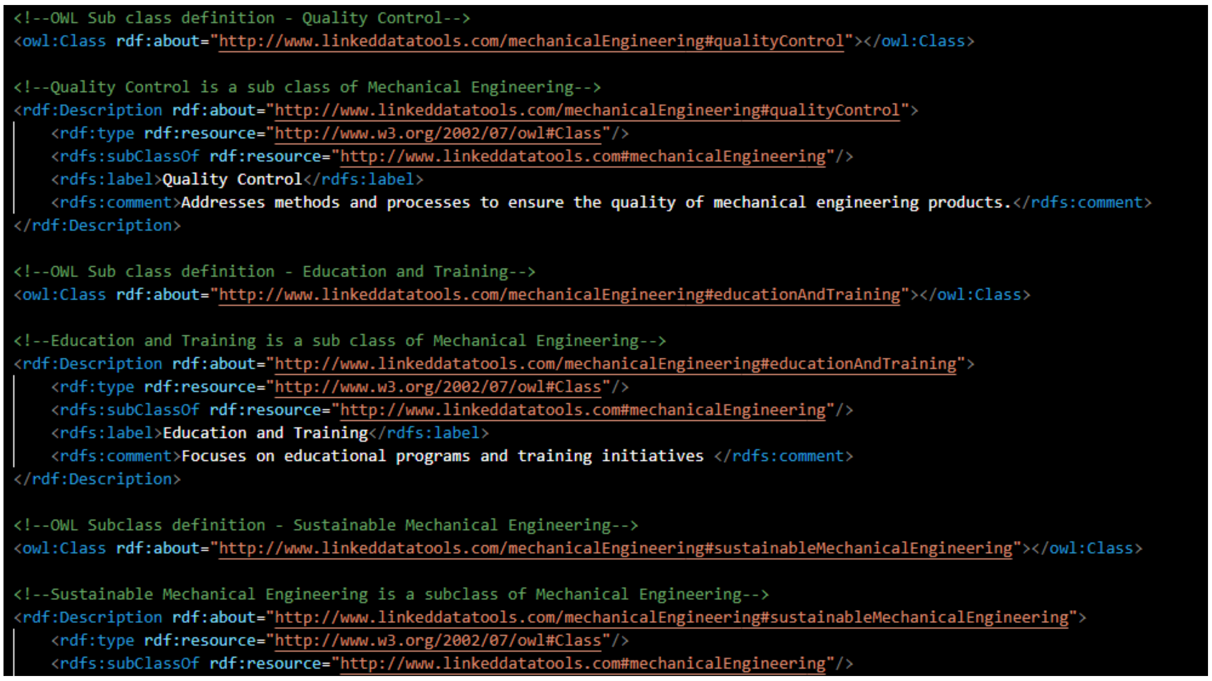
b) Concept Graph



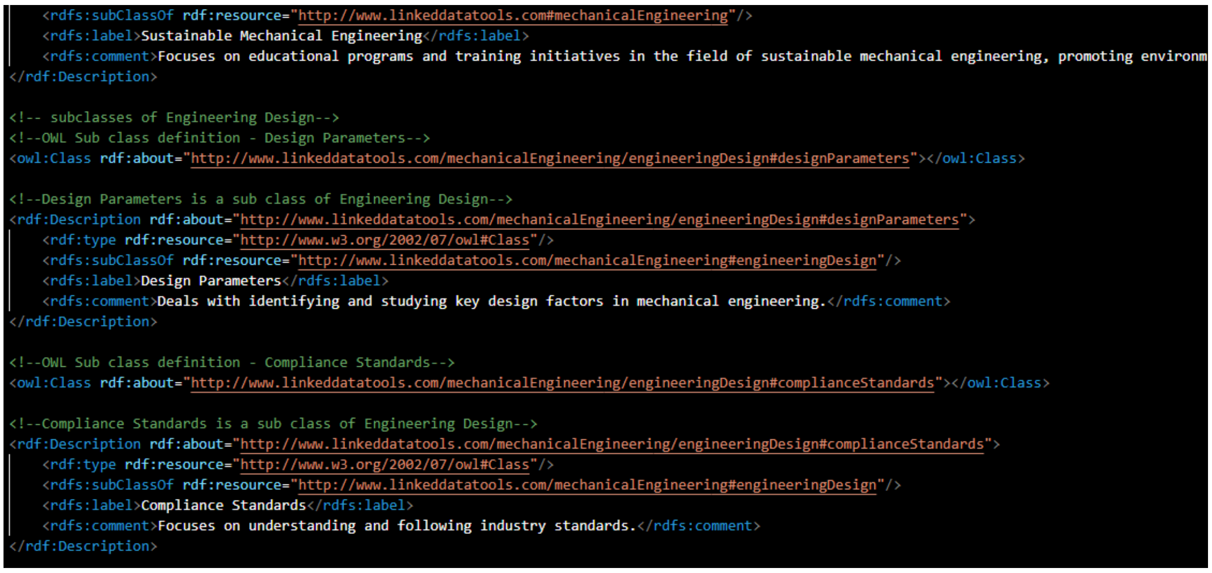
c) Design a suitable domain ontology.



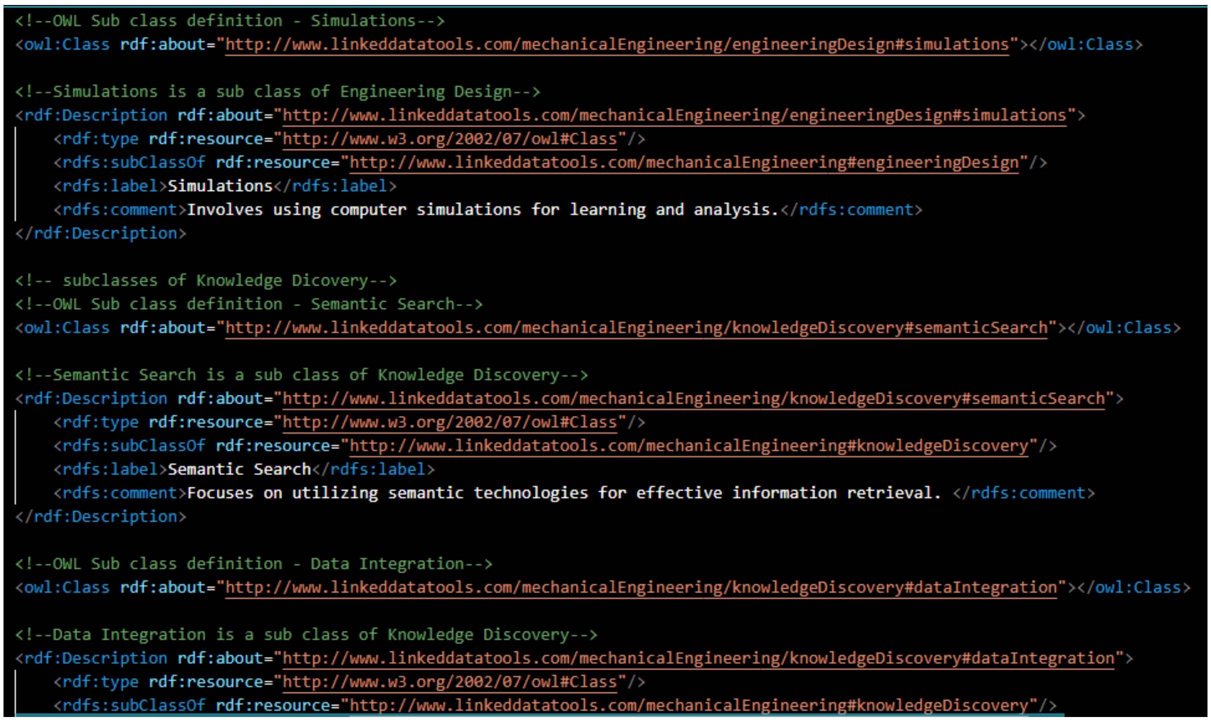
```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:mechanical="http://www.linkeddatatools.com/mechanicalEngineering">

<!-- OWL Header -->
<owl:Ontology rdf:about="http://www.linkeddatatools.com/mechanicalEngineering">
<dc:title> Mechanical Engineering Ontology</dc:title>
<dc:description>An Ontology for Mechanical Engineering  </dc:description>
</owl:Ontology>

<!--OWL Class definition- Thing-->
<owl:Class rdf:about="http://www.linkeddatatools.com#thing">
  <rdfs:label>The Thing</rdfs:label>
  <rdfs:comment>The Conceptual starting point of mechanical engineering hierarchy.</rdfs:comment>
</owl:Class>

<!--OWL Sub class definition- Mechanical Engineering-->
<owl:Class rdf:about="http://www.linkeddatatools.com#mechanicalEngineering"></owl:Class>

<!--Mechanical Engineering is sub class of Thing-->
<rdf:Description rdf:about="http://www.linkeddatatools.com#mechanicalEngineering">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#thing"/>
    <rdfs:label>Mechanical Engineering</rdfs:label>
    <rdfs:comment>class of all mechanical engineering types.</rdfs:comment>
</rdf:Description>
<!--subclasses of Mechanical Engineering-->
<!--OWL Sub class definition - Engineering Design-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign"></owl:Class>

<!--Engineering Design is a sub class of Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#mechanicalEngineering"/>
    <rdfs:label>Engineering Design</rdfs:label>
    <rdfs:comment>Overviews design and standard of mechanical systems.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Knowledge Discovery-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery"></owl:Class>

<!--Knowledge Discovery is a sub class of Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#mechanicalEngineering"/>
    <rdfs:label>Knowledge Discovery</rdfs:label>
    <rdfs:comment>Focuses on the exploration and acquisition of knowledge.</rdfs:comment>
</rdf:Description>
```



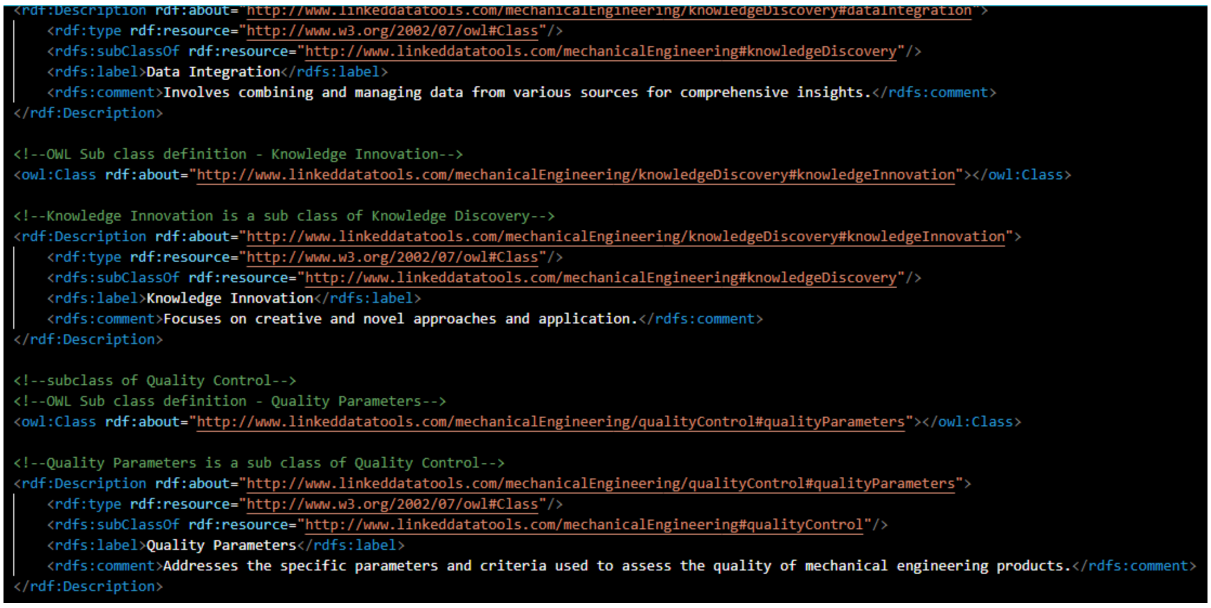
```
<!--OWL Sub class definition - Quality Control-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#qualityControl"></owl:Class>

<!--Quality Control is a sub class of Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#qualityControl">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#mechanicalEngineering"/>
    <rdfs:label>Quality Control</rdfs:label>
    <rdfs:comment>Addresses methods and processes to ensure the quality of mechanical engineering products.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Education and Training-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining"></owl:Class>

<!--Education and Training is a sub class of Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#mechanicalEngineering"/>
    <rdfs:label>Education and Training</rdfs:label>
    <rdfs:comment>Focuses on educational programs and training initiatives </rdfs:comment>
</rdf:Description>

<!--OWL Subclass definition - Sustainable Mechanical Engineering-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#sustainableMechanicalEngineering"></owl:Class>

<!--Sustainable Mechanical Engineering is a subclass of Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering#sustainableMechanicalEngineering">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#mechanicalEngineering"/>
```



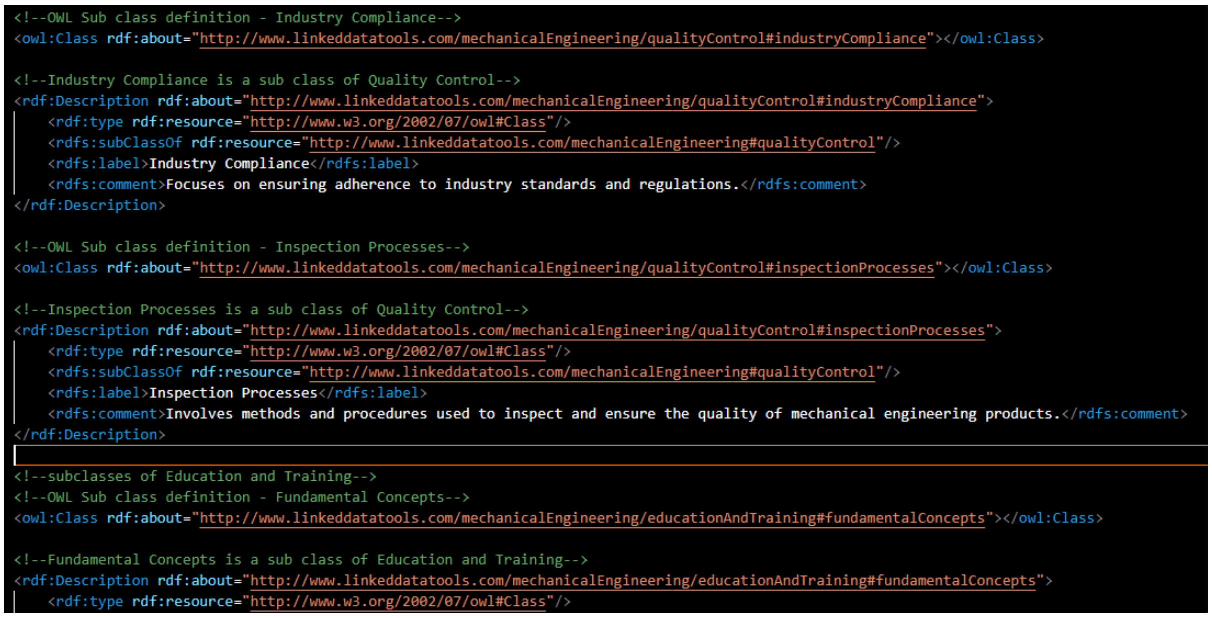
```
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com#mechanicalEngineering"/>
    <rdfs:label>Sustainable Mechanical Engineering</rdfs:label>
    <rdfs:comment>Focuses on educational programs and training initiatives in the field of sustainable mechanical engineering, promoting environm
</rdf:Description>

<!-- subclasses of Engineering Design-->
<!--OWL Sub class definition - Design Parameters-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#designParameters"></owl:Class>

<!--Design Parameters is a sub class of Engineering Design-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#designParameters">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign"/>
    <rdfs:label>Design Parameters</rdfs:label>
    <rdfs:comment>Deals with identifying and studying key design factors in mechanical engineering.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Compliance Standards-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#complianceStandards"></owl:Class>

<!--Compliance Standards is a sub class of Engineering Design-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#complianceStandards">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign"/>
    <rdfs:label>Compliance Standards</rdfs:label>
    <rdfs:comment>Focuses on understanding and following industry standards.</rdfs:comment>
</rdf:Description>
```



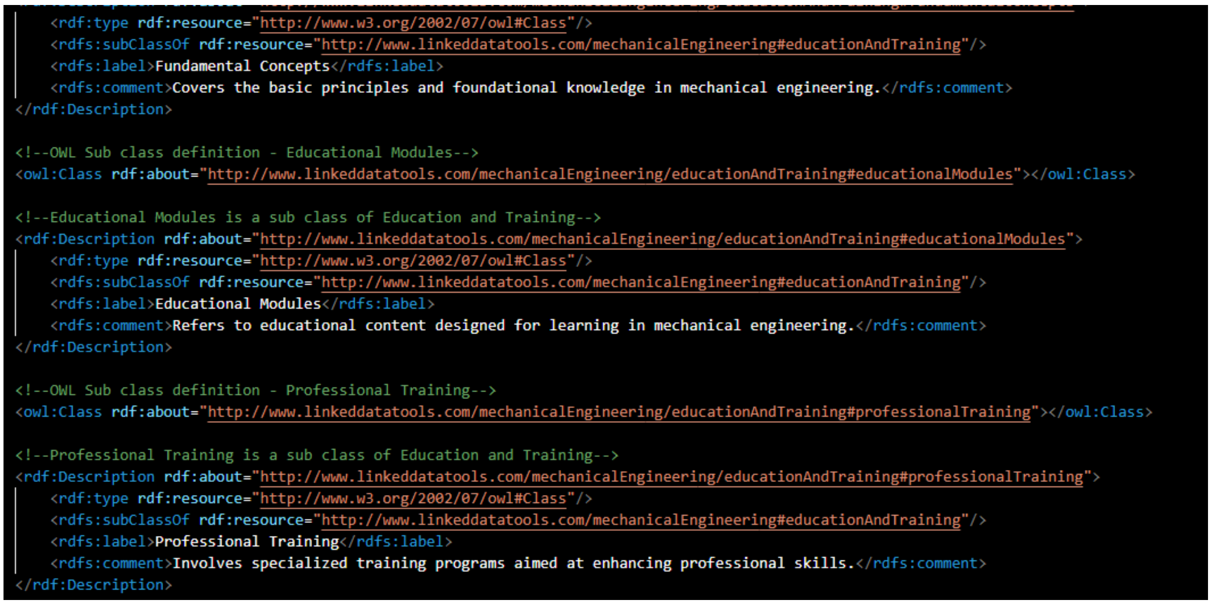
```
<!--OWL Sub class definition - Simulations-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#simulations"></owl:Class>

<!--Simulations is a sub class of Engineering Design-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#simulations">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign"/>
    <rdfs:label>Simulations</rdfs:label>
    <rdfs:comment>Involves using computer simulations for learning and analysis.</rdfs:comment>
</rdf:Description>

<!-- subclasses of Knowledge Dicovery-->
<!--OWL Sub class definition - Semantic Search-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#semanticSearch"></owl:Class>

<!--Semantic Search is a sub class of Knowledge Discovery-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#semanticSearch">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery"/>
    <rdfs:label>Semantic Search</rdfs:label>
    <rdfs:comment>Focuses on utilizing semantic technologies for effective information retrieval. </rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Data Integration-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#dataIntegration"></owl:Class>

<!--Data Integration is a sub class of Knowledge Discovery-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#dataIntegration">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery"/>
```



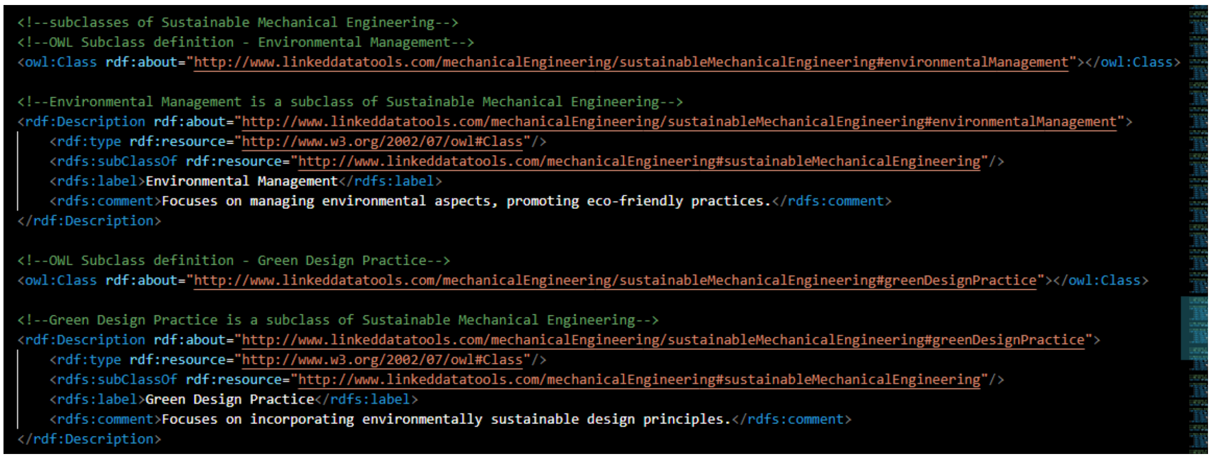
```
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#dataIntegration">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery"/>
    <rdfs:label>Data Integration</rdfs:label>
    <rdfs:comment>Involves combining and managing data from various sources for comprehensive insights.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Knowledge Innovation-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#knowledgeInnovation"></owl:Class>

<!--Knowledge Innovation is a sub class of Knowledge Discovery-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#knowledgeInnovation">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery"/>
    <rdfs:label>Knowledge Innovation</rdfs:label>
    <rdfs:comment>Focuses on creative and novel approaches and application.</rdfs:comment>
</rdf:Description>

<!--subclass of Quality Control-->
<!--OWL Sub class definition - Quality Parameters-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#qualityParameters"></owl:Class>

<!--Quality Parameters is a sub class of Quality Control-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#qualityParameters">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#qualityControl"/>
    <rdfs:label>Quality Parameters</rdfs:label>
    <rdfs:comment>Addresses the specific parameters and criteria used to assess the quality of mechanical engineering products.</rdfs:comment>
</rdf:Description>
```



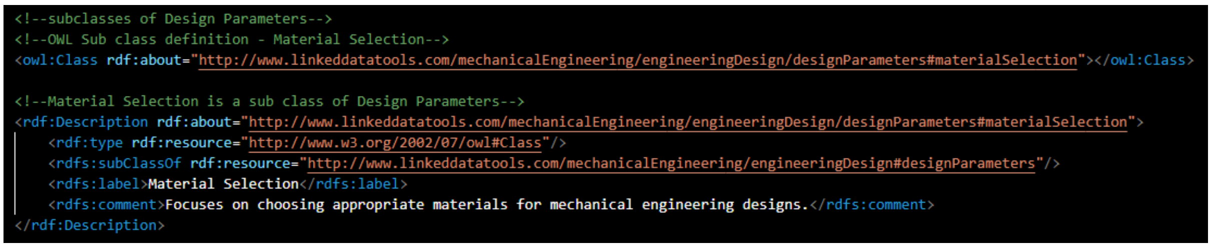
```
<!--OWL Sub class definition - Industry Compliance-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#industryCompliance"></owl:Class>

<!--Industry Compliance is a sub class of Quality Control-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#industryCompliance">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#qualityControl"/>
    <rdfs:label>Industry Compliance</rdfs:label>
    <rdfs:comment>Focuses on ensuring adherence to industry standards and regulations.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Inspection Processes-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#inspectionProcesses"></owl:Class>

<!--Inspection Processes is a sub class of Quality Control-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#inspectionProcesses">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#qualityControl"/>
    <rdfs:label>Inspection Processes</rdfs:label>
    <rdfs:comment>Involves methods and procedures used to inspect and ensure the quality of mechanical engineering products.</rdfs:comment>
</rdf:Description>

<!--subclasses of Education and Training-->
<!--OWL Sub class definition - Fundamental Concepts-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#fundamentalConcepts"></owl:Class>

<!--Fundamental Concepts is a sub class of Education and Training-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#fundamentalConcepts">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
```



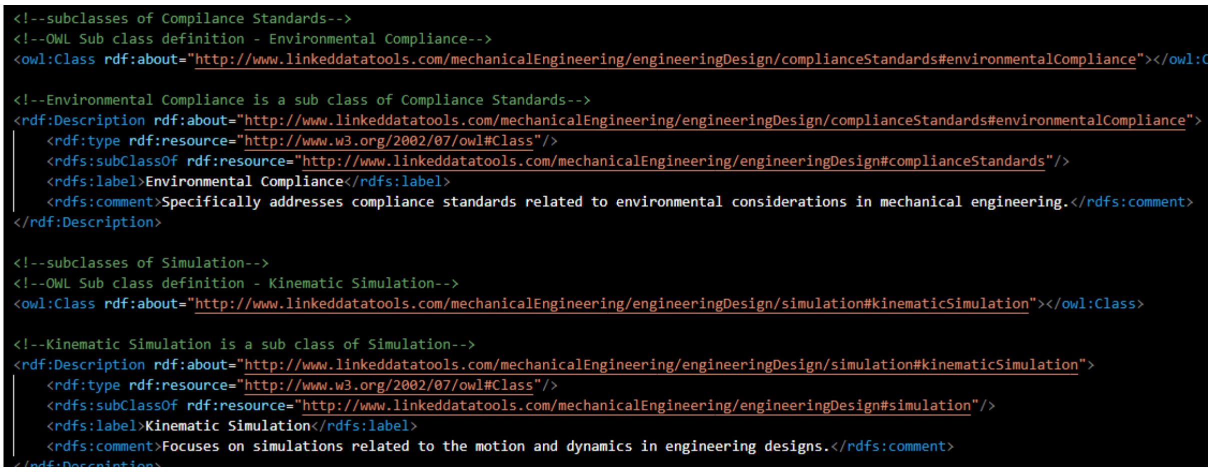
```
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining"/>
    <rdfs:label>Fundamental Concepts</rdfs:label>
    <rdfs:comment>Covers the basic principles and foundational knowledge in mechanical engineering.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Educational Modules-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#educationalModules"></owl:Class>

<!--Educational Modules is a sub class of Education and Training-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#educationalModules">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining"/>
    <rdfs:label>Educational Modules</rdfs:label>
    <rdfs:comment>Refers to educational content designed for learning in mechanical engineering.</rdfs:comment>
</rdf:Description>

<!--OWL Sub class definition - Professional Training-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#professionalTraining"></owl:Class>

<!--Professional Training is a sub class of Education and Training-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#professionalTraining">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining"/>
    <rdfs:label>Professional Training</rdfs:label>
    <rdfs:comment>Involves specialized training programs aimed at enhancing professional skills.</rdfs:comment>
</rdf:Description>
```



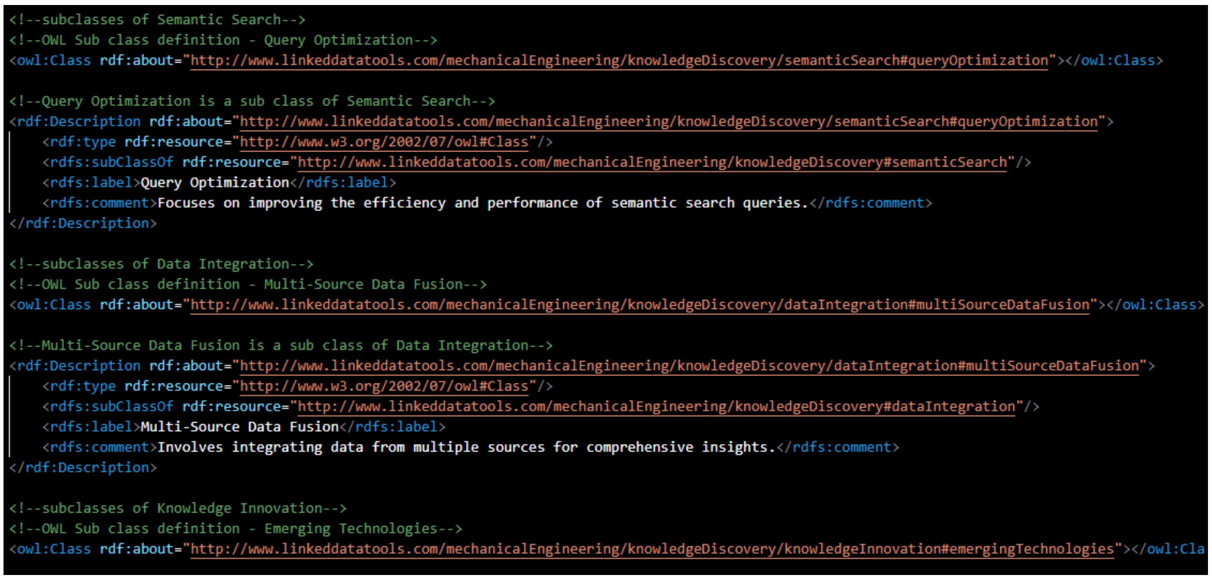
```
<!--subclasses of Sustainable Mechanical Engineering-->
<!--OWL Subclass definition - Environmental Management-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#environmentalManagement"></owl:Class>

<!--Environmental Management is a subclass of Sustainable Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#environmentalManagement">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#sustainableMechanicalEngineering"/>
    <rdfs:label>Environmental Management</rdfs:label>
    <rdfs:comment>Focuses on managing environmental aspects, promoting eco-friendly practices.</rdfs:comment>
</rdf:Description>

<!--OWL Subclass definition - Green Design Practice-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#greenDesignPractice"></owl:Class>

<!--Green Design Practice is a subclass of Sustainable Mechanical Engineering-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#greenDesignPractice">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#sustainableMechanicalEngineering"/>
    <rdfs:label>Green Design Practice</rdfs:label>
    <rdfs:comment>Focuses on incorporating environmentally sustainable design principles.</rdfs:comment>
</rdf:Description>
```



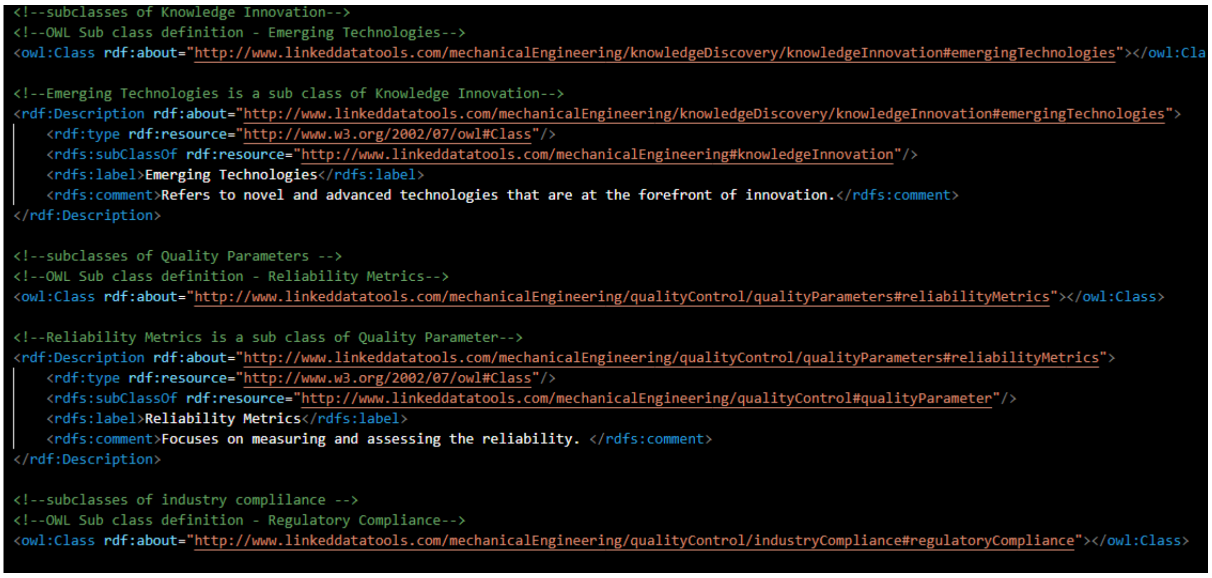
```
<!--subclasses of Design Parameters-->
<!--OWL Sub class definition - Material Selection-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign/designParameters#materialSelection"></owl:Class>

<!--Material Selection is a sub class of Design Parameters-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign/designParameters#materialSelection">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#designParameters"/>
    <rdfs:label>Material Selection</rdfs:label>
    <rdfs:comment>Focuses on choosing appropriate materials for mechanical engineering designs.</rdfs:comment>
</rdf:Description>
```



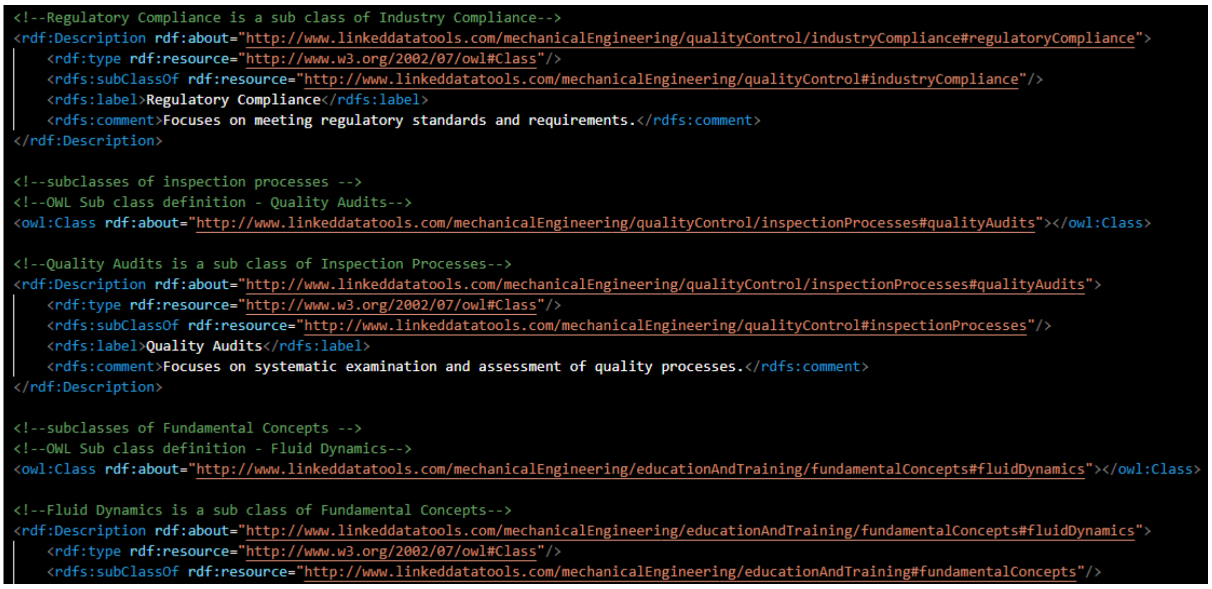
```
<!--subclasses of Compliance Standards-->
<!--OWL Sub class definition - Environmental Compliance-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign/complianceStandards#environmentalCompliance"></owl:C

<!--Environmental Compliance is a sub class of Compliance Standards-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign/complianceStandards#environmentalCompliance">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#complianceStandards"/>
    <rdfs:label>Environmental Compliance</rdfs:label>
    <rdfs:comment>Specifically addresses compliance standards related to environmental considerations in mechanical engineering.</rdfs:comment>
</rdf:Description>

<!--subclasses of Simulation-->
<!--OWL Sub class definition - Kinematic Simulation-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign/simulation#kinematicSimulation"></owl:Class>

<!--Kinematic Simulation is a sub class of Simulation-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign/simulation#kinematicSimulation">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#simulation"/>
    <rdfs:label>Kinematic Simulation</rdfs:label>
    <rdfs:comment>Focuses on simulations related to the motion and dynamics in engineering designs.</rdfs:comment>
</rdf:Description>
```



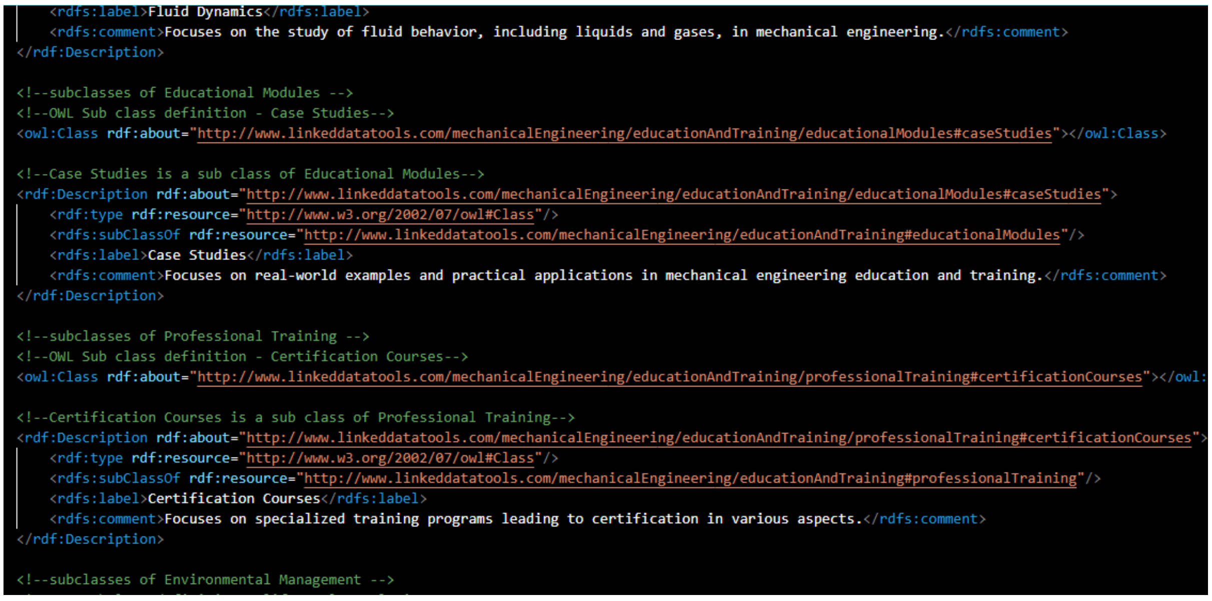
```
<!--subclasses of Semantic Search-->
<!--OWL Sub class definition - Query Optimization-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/semanticSearch#queryOptimization"></owl:Class>

<!--Query Optimization is a sub class of Semantic Search-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/semanticSearch#queryOptimization">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#semanticSearch"/>
    <rdfs:label>Query Optimization</rdfs:label>
    <rdfs:comment>Focuses on improving the efficiency and performance of semantic search queries.</rdfs:comment>
</rdf:Description>

<!--subclasses of Data Integration-->
<!--OWL Sub class definition - Multi-Source Data Fusion-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/dataIntegration#multiSourceDataFusion"></owl:Class>

<!--Multi-Source Data Fusion is a sub class of Data Integration-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/dataIntegration#multiSourceDataFusion">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery#dataIntegration"/>
    <rdfs:label>Multi-Source Data Fusion</rdfs:label>
    <rdfs:comment>Involves integrating data from multiple sources for comprehensive insights.</rdfs:comment>
</rdf:Description>

<!--subclasses of Knowledge Innovation-->
<!--OWL Sub class definition - Emerging Technologies-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/knowledgeInnovation#emergingTechnologies"></owl:Cla
```



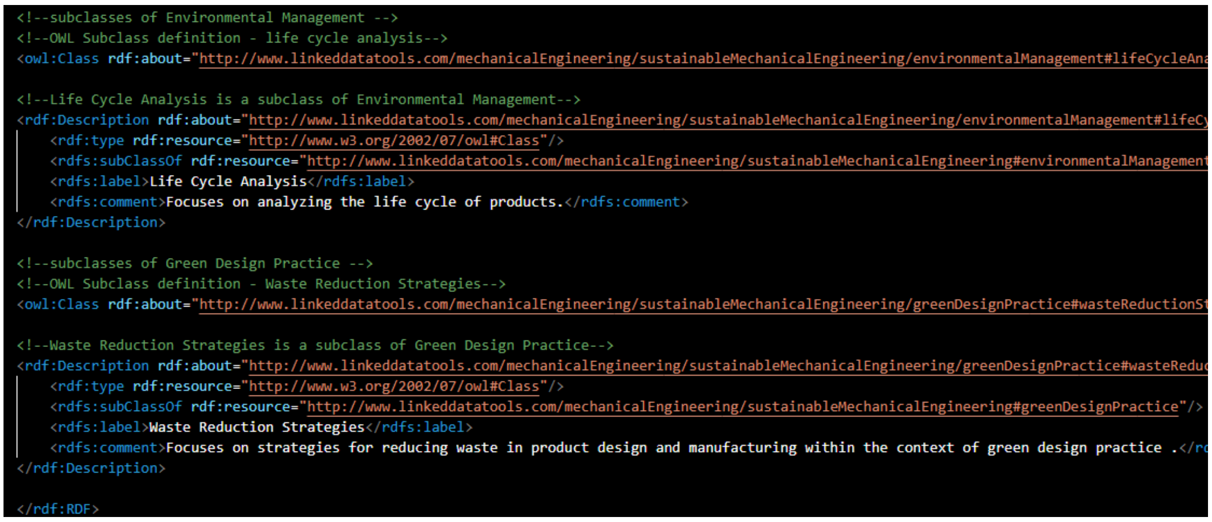
```xml
<!--subclasses of Knowledge Innovation-->
<!--OWL Sub class definition - Emerging Technologies-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/knowledgeInnovation#emergingTechnologies"></owl:Cla

<!--Emerging Technologies is a sub class of Knowledge Innovation-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/knowledgeDiscovery/knowledgeInnovation#emergingTechnologies">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering#knowledgeInnovation"/>
    <rdfs:label>Emerging Technologies</rdfs:label>
    <rdfs:comment>Refers to novel and advanced technologies that are at the forefront of innovation.</rdfs:comment>
</rdf:Description>

<!--subclasses of Quality Parameters -->
<!--OWL Sub class definition - Reliability Metrics-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/qualityParameters#reliabilityMetrics"></owl:Class>

<!--Reliability Metrics is a sub class of Quality Parameter-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/qualityParameters#reliabilityMetrics">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#qualityParameter"/>
    <rdfs:label>Reliability Metrics</rdfs:label>
    <rdfs:comment>Focuses on measuring and assessing the reliability. </rdfs:comment>
</rdf:Description>

<!--subclasses of industry complilance -->
<!--OWL Sub class definition - Regulatory Compliance-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/industryCompliance#regulatoryCompliance"></owl:Class>
```



```xml
<!--Regulatory Compliance is a sub class of Industry Compliance-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/industryCompliance#regulatoryCompliance">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#industryCompliance"/>
    <rdfs:label>Regulatory Compliance</rdfs:label>
    <rdfs:comment>Focuses on meeting regulatory standards and requirements.</rdfs:comment>
</rdf:Description>

<!--subclasses of inspection processes -->
<!--OWL Sub class definition - Quality Audits-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/inspectionProcesses#qualityAudits"></owl:Class>

<!--Quality Audits is a sub class of Inspection Processes-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/inspectionProcesses#qualityAudits">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#inspectionProcesses"/>
    <rdfs:label>Quality Audits</rdfs:label>
    <rdfs:comment>Focuses on systematic examination and assessment of quality processes.</rdfs:comment>
</rdf:Description>

<!--subclasses of Fundamental Concepts -->
<!--OWL Sub class definition - Fluid Dynamics-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining/fundamentalConcepts#fluidDynamics"></owl:Class>

<!--Fluid Dynamics is a sub class of Fundamental Concepts-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining/fundamentalConcepts#fluidDynamics">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#fundamentalConcepts"/>
```



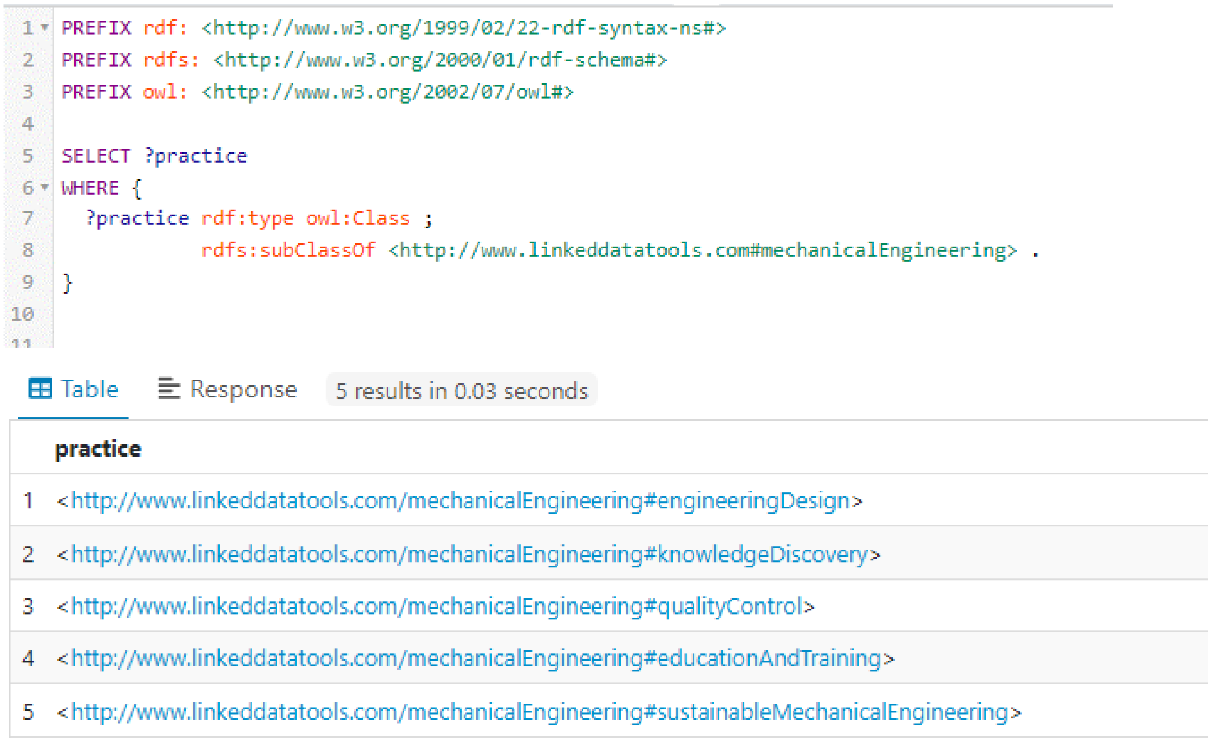
```xml
    <rdfs:label>Fluid Dynamics</rdfs:label>
    <rdfs:comment>Focuses on the study of fluid behavior, including liquids and gases, in mechanical engineering.</rdfs:comment>
</rdf:Description>

<!--subclasses of Educational Modules -->
<!--OWL Sub class definition - Case Studies-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining/educationalModules#caseStudies"></owl:Class>

<!--Case Studies is a sub class of Educational Modules-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining/educationalModules#caseStudies">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#educationalModules"/>
    <rdfs:label>Case Studies</rdfs:label>
    <rdfs:comment>Focuses on real-world examples and practical applications in mechanical engineering education and training.</rdfs:comment>
</rdf:Description>

<!--subclasses of Professional Training -->
<!--OWL Sub class definition - Certification Courses-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining/professionalTraining#certificationCourses"></owl:

<!--Certification Courses is a sub class of Professional Training-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining/professionalTraining#certificationCourses">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#professionalTraining"/>
    <rdfs:label>Certification Courses</rdfs:label>
    <rdfs:comment>Focuses on specialized training programs leading to certification in various aspects.</rdfs:comment>
</rdf:Description>

<!--subclasses of Environmental Management -->
```



```
<!--subclasses of Environmental Management -->
<!--OWL Subclass definition - life cycle analysis-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering/environmentalManagement#lifeCycleAna

<!--Life Cycle Analysis is a subclass of Environmental Management-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering/environmentalManagement#lifeCy
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#environmentalManagement
    <rdfs:label>Life Cycle Analysis</rdfs:label>
    <rdfs:comment>Focuses on analyzing the life cycle of products.</rdfs:comment>
</rdf:Description>

<!--subclasses of Green Design Practice -->
<!--OWL Subclass definition - Waste Reduction Strategies-->
<owl:Class rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering/greenDesignPractice#wasteReductionSt

<!--Waste Reduction Strategies is a subclass of Green Design Practice-->
<rdf:Description rdf:about="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering/greenDesignPractice#wasteReduc
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#greenDesignPractice"/>
    <rdfs:label>Waste Reduction Strategies</rdfs:label>
    <rdfs:comment>Focuses on strategies for reducing waste in product design and manufacturing within the context of green design practice .</r
</rdf:Description>

</rdf:RDF>
```

d) Write 05 SPARQL queries and support your answer with valid screen shots and summarized elaborations.

i) Finding the subclasses of Mechanical Engineering

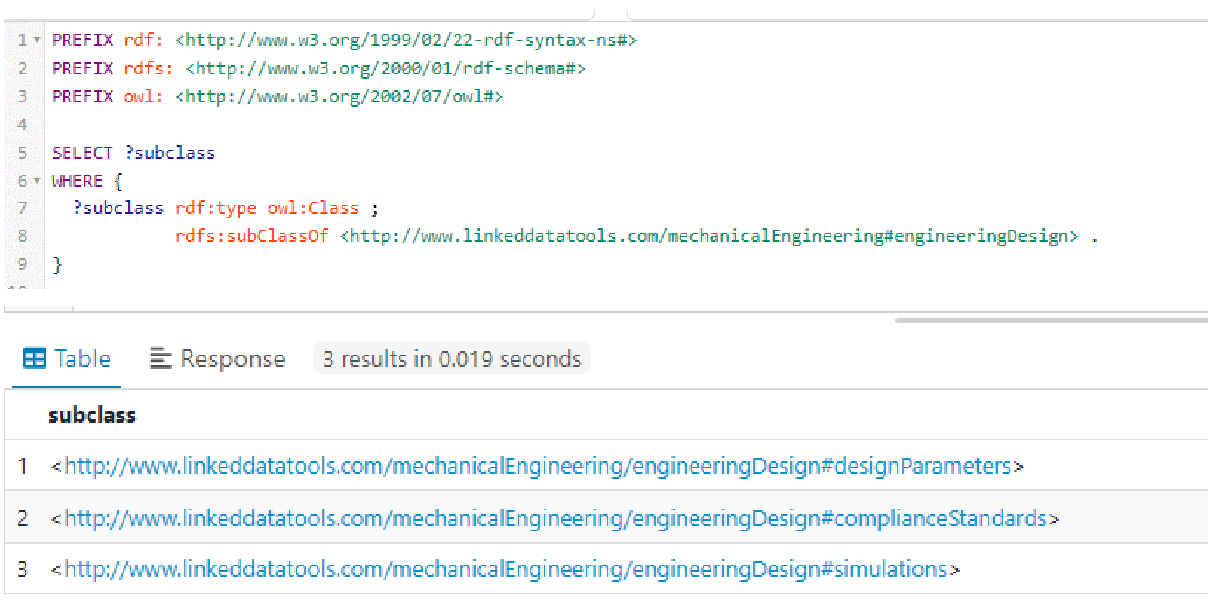


```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?practice
WHERE {
  ?practice rdf:type owl:Class ;
            rdfs:subClassOf <http://www.linkeddatatools.com#mechanicalEngineering> .
}
```

	practice
1	<http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign>
2	<http://www.linkeddatatools.com/mechanicalEngineering#knowledgeDiscovery>
3	<http://www.linkeddatatools.com/mechanicalEngineering#qualityControl>
4	<http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining>
5	<http://www.linkeddatatools.com/mechanicalEngineering#sustainableMechanicalEngineering>

This SPARQL query looks for different subclasses of mechanical engineering in a dataset, and it returns those subclasses.

ii) Retrieve all subclasses of Engineering design.

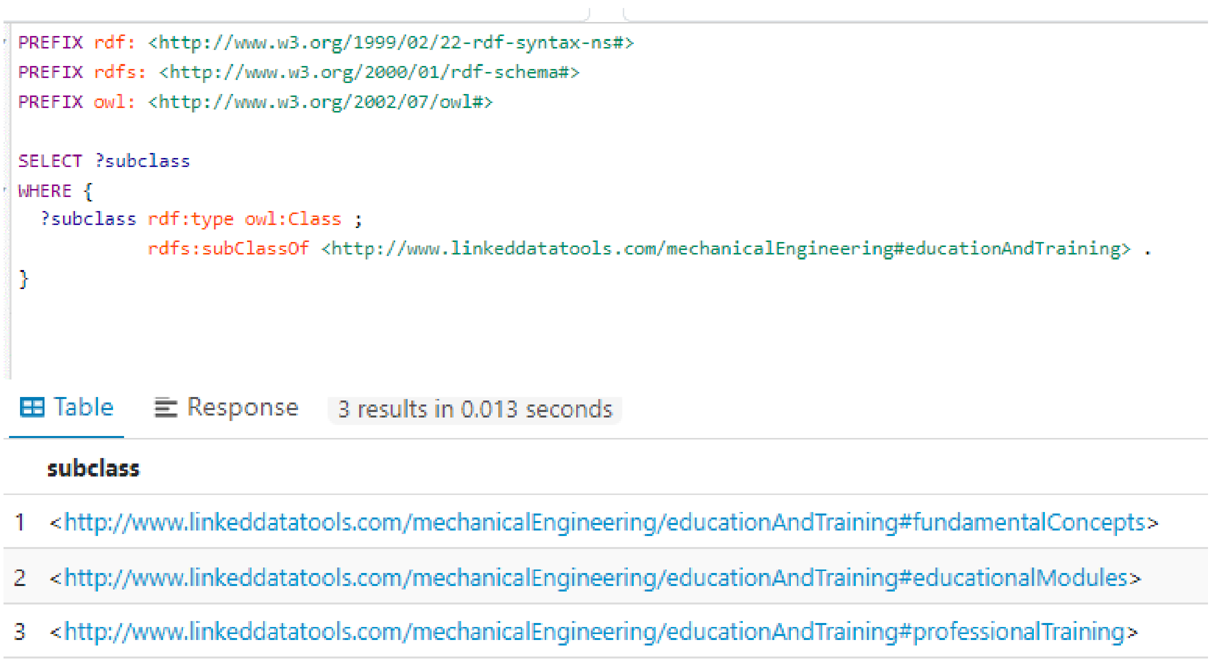


```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?subclass
WHERE {
  ?subclass rdf:type owl:Class ;
            rdfs:subClassOf <http://www.linkeddatatools.com/mechanicalEngineering#engineeringDesign> .
}
```

	subclass
1	<http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#designParameters>
2	<http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#complianceStandards>
3	<http://www.linkeddatatools.com/mechanicalEngineering/engineeringDesign#simulations>

This SPARQL query looks for different subclasses of engineering design, and it returns those subclasses.

iii) Retrieve all the subclasses of education and training

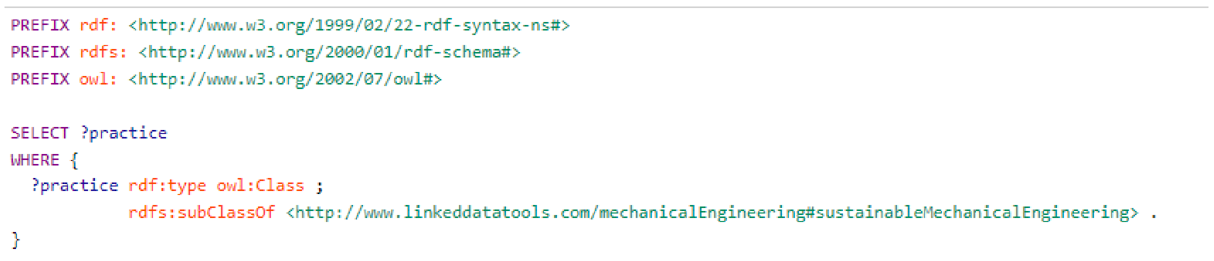


```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?subclass
WHERE {
  ?subclass rdf:type owl:Class ;
            rdfs:subClassOf <http://www.linkeddatatools.com/mechanicalEngineering#educationAndTraining> .
}
```

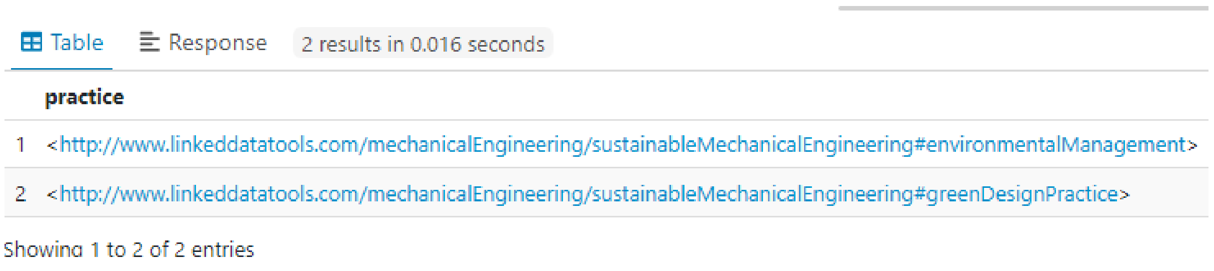
	subclass
1	<http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#fundamentalConcepts>
2	<http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#educationalModules>
3	<http://www.linkeddatatools.com/mechanicalEngineering/educationAndTraining#professionalTraining>

This SPARQL query looks for different subclasses of education and training, and it returns those subclasses.

iv) Retrieve all the subclasses of Sustainable Mechanical Engineering



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?practice
WHERE {
  ?practice rdf:type owl:Class ;
          rdfs:subClassOf <http://www.linkeddatatools.com/mechanicalEngineering#sustainableMechanicalEngineering> .
}
```



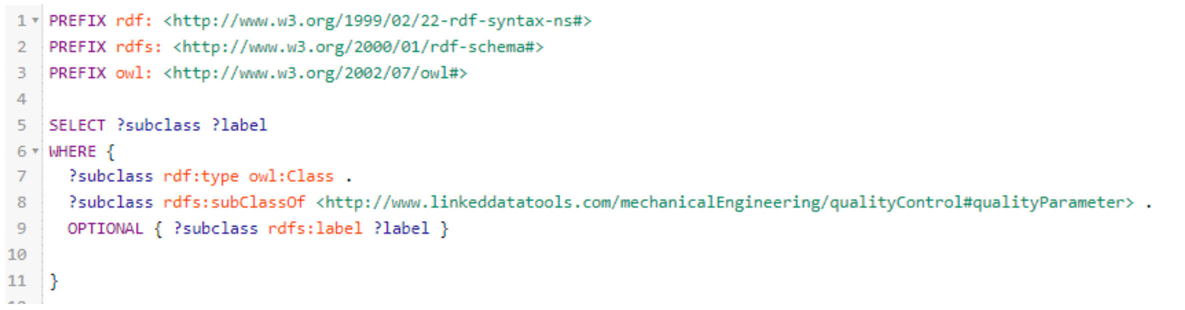
2 results in 0.016 seconds

	practice
1	<http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#environmentalManagement>
2	<http://www.linkeddatatools.com/mechanicalEngineering/sustainableMechanicalEngineering#greenDesignPractice>

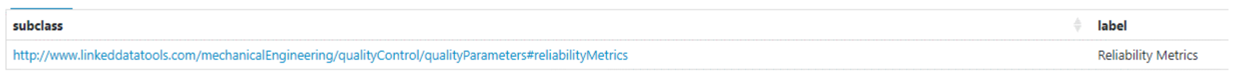
Showing 1 to 2 of 2 entries

This SPARQL query looks for different subclasses of sustainable mechanical engineering in a dataset, and it returns those subclasses

v) Retrieve all the subclass of Quality Parameters



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?subclass ?label
WHERE {
  ?subclass rdf:type owl:Class .
  ?subclass rdfs:subClassOf <http://www.linkeddatatools.com/mechanicalEngineering/qualityControl#qualityParameter> .
  OPTIONAL { ?subclass rdfs:label ?label }

}
```



subclass	label
http://www.linkeddatatools.com/mechanicalEngineering/qualityControl/qualityParameters#reliabilityMetrics	Reliability Metrics

This SPARQL query selects the subclasses of 'Quality Parameters' and, retrieves their labels too.

OVERALL SUMMARY

This is an ontology-based question which is designed to be utilized by the mechanical engineers. There are several sections that has been covered which includes identifying the scope, proposing relevant competency questions, providing an appropriate concept graph, designing a suitable domain ontology and writing SPARQL queries.

Explanations of the XMLS

"*Thing***"** class was first introduced with the URI "http://www.linkeddatatools.com#thing." This class called "*Thing*", functions as the conceptual beginning point for the mechanical engineering hierarchy. A sub class, "*Mechanical Engineering,*" is created.

Secondly, this ontology is further expanded into multiple subclasses from the main sub class "*Mechanical Engineering*". Every subclass reflects different aspects of the field. "*Knowledge Discovery*" is devoted to the investigation and acquisition of knowledge in mechanical engineering, whereas "*Engineering Design*" concentrates on the design and standardization of mechanical systems. The term "*Quality Control*" refers to the processes used to guarantee the caliber of mechanical engineering output. "*Education and Training*" focuses on initiatives for training and educational programs unique to the field. Last but not least, "*Sustainable Mechanical Engineering*" includes training and educational programs that prioritize eco-friendly methods.

This is further classified into subclasses, for instance the "*Engineering Design*", which is a subclass of the mechanical engineering representation has 3 subclasses which is "*Design Parameters*", which pinpoints important design elements, "*Compliance Standards*" highlights the importance of following industry norms and "*Simulations*" refers to the use of computer simulations for research and education.

Moreover, "*Knowledge Discovery*", a subclass of mechanical Engineering also has 03 subclasses which includes "*Semantic Search*", which is on using semantic technologies to retrieve information precisely, "*Data Integration*" is to combine various data sources to provide thorough insights and "*Knowledge Innovation*" promotes innovative methods in the industry.

"*Quality Control*", is a subclass of mechanical engineering and it also has 03 subclasses which are "*Quality Parameter*", addresses the specific parameters and criteria used to assess the quality of mechanical engineering products, "*Industry Compliance*", focuses on ensuring adherence to industry standards and regulations and "*Inspection Processes*", involves methods and procedures used to inspect and ensure the quality of mechanical engineering product.

"*Education and Training*" is a subclass of mechanical engineering. This includes 03 more subclasses which are "*Fundamental Concepts*", covers the basic principles and foundational knowledge in mechanical engineering, "*Educational Modules*", refers to educational content designed for learning in mechanical engineering and "*Professional Training*", which involves specialized training programs aimed at enhancing professional skills.

"*Sustainable Mechanical Engineering*", is a subclass of mechanical engineering and it has 02 more subclasses. This includes "*Environmental Management*", that focuses on managing environmental aspects, promoting eco-friendly practices and "*Green Design Practice*", that focuses on incorporating environmentally sustainable design principles.

Lastly, each of these subclasses have one more subclass for each of them. Few of them includes:

- Material Selection is a sub class of Design Parameters.
- Environmental Compliance is a sub class of Compliance Standards.
- Kinematic Simulation is a sub class of Simulation.
- Multi-Source Data Fusion is a sub class of Data Integration.
- Query Optimization is a sub class of Semantic Search.
- Multi-source Data fusion is a subclass of Data Integration.
- Emerging Technologies is a sub class of Knowledge Innovation.
- Reliability Metrics is a sub class of Quality Parameter.
- Regulatory Compliance is a sub class of Industry Compliance.
- Quality Audits is a sub class of Inspection Processes.
- Fluid Dynamics is a sub class of Fundamental Concepts.
- Case Studies is a subclass of Educational Modules.
- Certification Courses is a subclass of Professional Training.

- Life cycle Analysis is a subclass of Environmental Management.
- Waste Reduction Strategies is a subclass of Green Design Practice.

QUESTION 03

Priority Queue Class

```python
class PriorityQueue:
    def __init__(self):
        self.queue = []

    def push(self, item, priority):
        entry = (priority, item)
        self.queue.append(entry)
        self._bubble_up(len(self.queue) - 1)

    def pop(self):
        if not self.is_empty():
            self._swap(0, len(self.queue) - 1)
            item = self.queue.pop()[1]
            self._bubble_down(0)
            return item
        return None

    def is_empty(self):
        return len(self.queue) == 0

    def _bubble_up(self, index):
        while index > 0:
            parent_index = (index - 1) // 2
            if self.queue[parent_index][0] > self.queue[index][0]:
                self._swap(parent_index, index)
                index = parent_index
            else:
                break

    def _bubble_down(self, index):
        while True:
            left_child_index = 2 * index + 1
            right_child_index = 2 * index + 2
            smallest = index

            if left_child_index < len(self.queue) and self.queue[left_child_index][0] < self.queue[smallest][0]:
                smallest = left_child_index

            if right_child_index < len(self.queue) and self.queue[right_child_index][0] < self.queue[smallest][0]:
                smallest = right_child_index

            if smallest != index:
                self._swap(index, smallest)
                index = smallest
            else:
                break

    def _swap(self, i, j):
        self.queue[i], self.queue[j] = self.queue[j], self.queue[i]
```

The **PriorityQueue** class implements a priority queue using a min-heap data structure. Here are the details:

Initialization: The __init__ method initializes an empty list (self. queue) to store priority-item pairs.

Push Method: The push method adds an item with a given priority to the priority queue. It creates a tuple (priority, item) and appends it to the queue. The _bubble_up method is called to maintain the min-heap property.

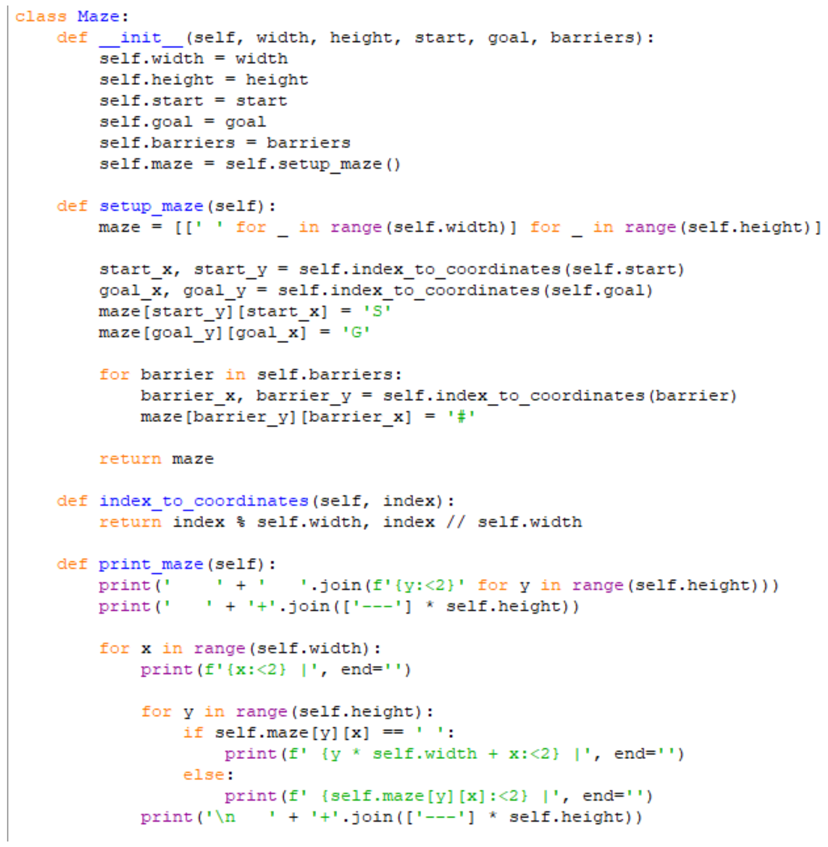
Pop Method: The pop method removes and returns the item with the highest priority. It swaps the first and last elements, pops the last element, and calls _bubble_down to maintain the min-heap property.

Is Empty Method: The is_empty method checks if the priority queue is empty.

Private Methods (_bubble_up and _bubble_down): _bubble_up and _bubble_down are helper methods to maintain the min-heap property after pushing and popping elements, respectively. _bubble_up moves the element up the heap until its priority is not greater than its parent's priority. _bubble_down moves the element down the heap until its priority is not greater than its children's priorities.

Swap Method (_swap): The _swap method swaps two elements in the priority queue.

Maze Class



```python
class Maze:
    def __init__(self, width, height, start, goal, barriers):
        self.width = width
        self.height = height
        self.start = start
        self.goal = goal
        self.barriers = barriers
        self.maze = self.setup_maze()

    def setup_maze(self):
        maze = [[' ' for _ in range(self.width)] for _ in range(self.height)]

        start_x, start_y = self.index_to_coordinates(self.start)
        goal_x, goal_y = self.index_to_coordinates(self.goal)
        maze[start_y][start_x] = 'S'
        maze[goal_y][goal_x] = 'G'

        for barrier in self.barriers:
            barrier_x, barrier_y = self.index_to_coordinates(barrier)
            maze[barrier_y][barrier_x] = '#'

        return maze

    def index_to_coordinates(self, index):
        return index % self.width, index // self.width

    def print_maze(self):
        print('    ' + '   '.join(f'{y:<2}' for y in range(self.height)))
        print('   ' + '+'.join(['---'] * self.height))

        for x in range(self.width):
            print(f'{x:<2} |', end='')

            for y in range(self.height):
                if self.maze[y][x] == ' ':
                    print(f' {y * self.width + x:<2} |', end='')
                else:
                    print(f' {self.maze[y][x]:<2} |', end='')
            print('\n   ' + '+'.join(['---'] * self.height))
```

Initialization: The __init__ method initializes the maze with width, height, start and goal points, and barriers. It calls the setup_maze method to create the maze grid.

Setup Maze Method: The setup_maze method initializes the maze grid (self. maze) with spaces. It sets the start and goal points and places barriers. The start, goal, and barriers are represented as 'S', 'G', and '#', respectively.

Index to Coordinates Method (index_to_coordinates): Converts a 1D index to 2D coordinates in the maze grid.

Print Maze Method (print_maze): Prints the maze grid with labeled coordinates, start, goal, and barriers.

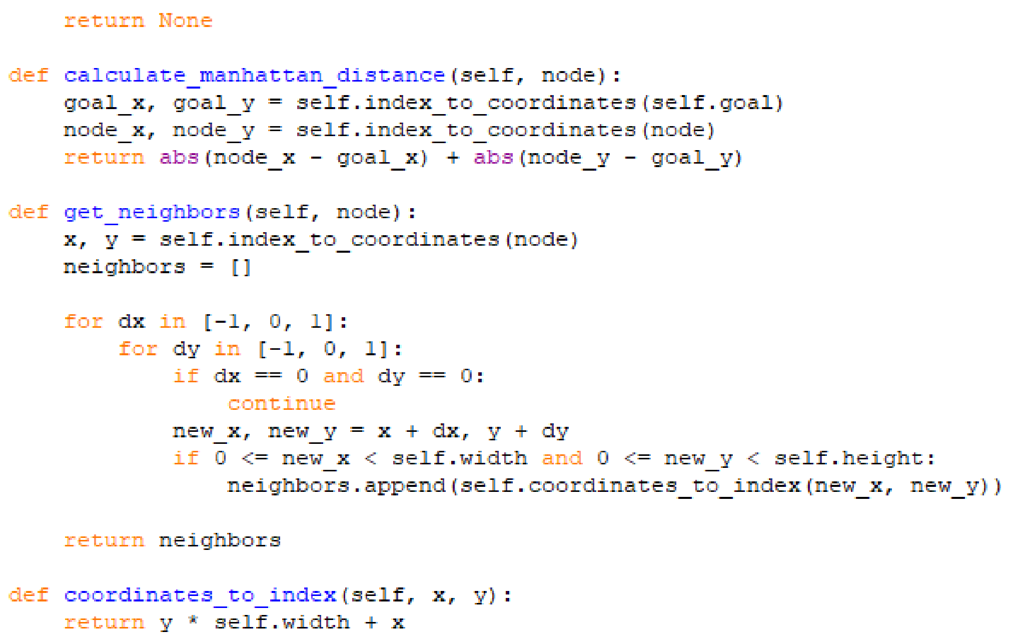
```
def dfs_analysis(self, start, goal):
    start_time = time.perf_counter()
    visited = set()
    stack = [(start, [start])]

    end_time = 0

    while stack:
        node, path = stack.pop()
        if node not in visited:
            visited.add(node)

            if node == goal:
                end_time = time.time()
                return {
                    'path': path,
                    'time': end_time - start_time,
                    'length': len(path) - 1
                }

            neighbors = self.get_neighbors(node)

            for neighbor in neighbors:
                if neighbor not in visited and neighbor not in self.barriers:
                    stack.append((neighbor, path + [neighbor]))

    return {
        'path': [],
        'time': end_time - start_time,
        'length': 0
    }

def a_star(self):
    priority_queue = PriorityQueue()
    start_time = time.perf_counter()
    start_node = self.start
    goal_node = self.goal

    priority_queue.push((start_node, [start_node]), self.calculate_manhattan_distance(start_node))
    visited = set()

    while not priority_queue.is_empty():
        current, path = priority_queue.pop()

        node = current

        if node not in visited:
            visited.add(node)

            if node == goal_node:
                end_time = time.time()
                return {
                    'path': path,
                    'time': end_time - start_time,
                    'length': len(path) - 1
                }
            neighbors = self.get_neighbors(node)

            for neighbor in neighbors:
                if neighbor not in visited and neighbor not in self.barriers:
                    new_path = path + [neighbor]
                    priority = len(new_path) + self.calculate_manhattan_distance(neighbor)
                    priority_queue.push((neighbor, new_path), priority)
```

DFS Analysis Method (dfs_analysis):

Performs Depth-First Search analysis to find a path from the start to the goal. Returns information about the path, time taken, and path length.

A Search Method (a_star*):

Implements the A* search algorithm to find an optimal path from the start to the goal. Returns information about the path, time taken, and path length.



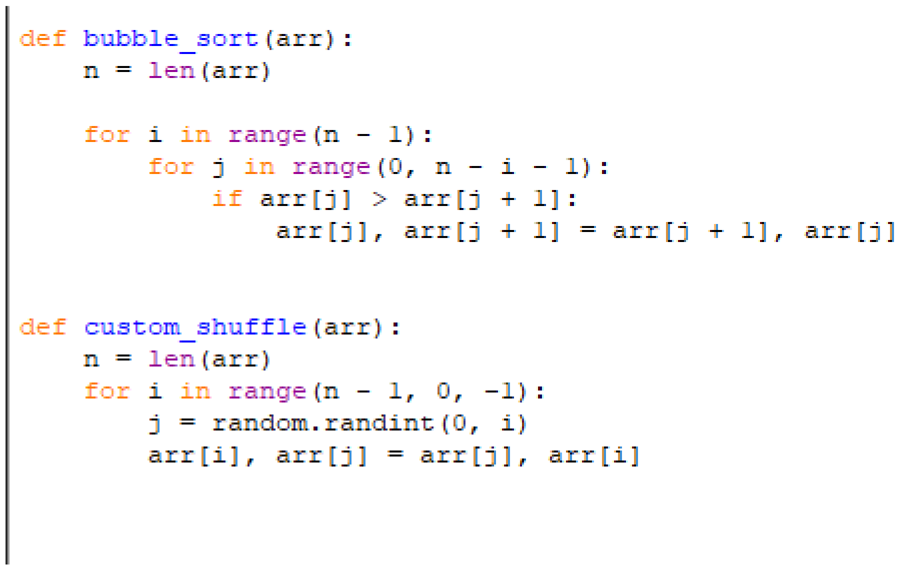
```
        return None

    def calculate_manhattan_distance(self, node):
        goal_x, goal_y = self.index_to_coordinates(self.goal)
        node_x, node_y = self.index_to_coordinates(node)
        return abs(node_x - goal_x) + abs(node_y - goal_y)

    def get_neighbors(self, node):
        x, y = self.index_to_coordinates(node)
        neighbors = []

        for dx in [-1, 0, 1]:
            for dy in [-1, 0, 1]:
                if dx == 0 and dy == 0:
                    continue
                new_x, new_y = x + dx, y + dy
                if 0 <= new_x < self.width and 0 <= new_y < self.height:
                    neighbors.append(self.coordinates_to_index(new_x, new_y))

        return neighbors

    def coordinates_to_index(self, x, y):
        return y * self.width + x
```

Heuristic Calculation Method (calculate_manhattan_distance): Calculates the Manhattan distance heuristic between a given node and the goal.

Get Neighbours Method (get_neighbors):

returns the neighbouring nodes of a given node.

Coordinates to Index Method (coordinates_to_index): Converts 2D coordinates to a 1D index.



```python
def bubble_sort(arr):
    n = len(arr)

    for i in range(n - 1):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]


def custom_shuffle(arr):
    n = len(arr)
    for i in range(n - 1, 0, -1):
        j = random.randint(0, i)
        arr[i], arr[j] = arr[j], arr[i]
```

Function of Bubble Sort:

Accepts a list called arr as input. The input list is sorted in-place, ascending order. This program executes the Bubble Sort algorithm, which compares and swaps neighboring elements repeatedly until the entire list is sorted.

Custom Shuffle Function:

Input: Takes a list arr as input.

Output: Shuffles the input list randomly in-place.

Description: Implements the Fisher-Yates Shuffle algorithm, where it iterates through the list and swaps each element with a randomly selected element that comes before it, ensuring a random permutation of the list.

Main Function

```python
def main():
    width, height = 6, 6
    num_mazes = 3
    dfs_times = []
    dfs_lengths = []
    a_star_times = []
    a_star_lengths = []

    for maze_num in range(1, num_mazes + 1):
        print(f"\nMaze {maze_num}:")

        start_node = random.randint(0, 11)
        goal_node = random.randint(24, 35)
        all_nodes = list(set(range(36)) - {start_node, goal_node})
        random.shuffle(all_nodes)
        barrier_nodes = all_nodes[:4]

        maze = Maze(width=width, height=height, start=start_node, goal=goal_node, barriers=barrier_nodes)

        print("\nInitial Maze:")
        maze.print_maze()

        print("\nDFS Analysis:")
        dfs_result = maze.dfs_analysis(start_node, goal_node)

        if dfs_result['path']:
            print("Visited Nodes (Processing Order):", dfs_result['path'][:-1])
            print("Time to Find Goal (seconds):", dfs_result['time'])
            print("Final Path:", dfs_result['path'])
            dfs_times.append(dfs_result['time'])
            dfs_lengths.append(dfs_result['length'])
        else:
            print("Goal not reachable.")

        print("\nHeuristic Cost Calculation:")
        for node in range(width * height):
            heuristic_cost = maze.calculate_manhattan_distance(node)
            print(f"Node {node}: Heuristic Cost = {heuristic_cost}")

        print("\nA* Analysis:")
        a_star_result = maze.a_star()
```

```python
        if a_star_result:
            print("Visited Nodes (Processing Order):", a_star_result['path'][:-1])
            print("Time to Find Goal (seconds):", a_star_result['time'])
            print("Final Path:", a_star_result['path'])
            a_star_times.append(a_star_result['time'])
            a_star_lengths.append(a_star_result['length'])
        else:
            print("Goal not reachable.")

    print("\nAnalysis of Results:")
    print("DFS Times:", dfs_times)
    print("DFS Mean Time:", statistics.mean(dfs_times))
    print("DFS Variance Time:", statistics.variance(dfs_times))
    print("DFS Lengths:", dfs_lengths)
    print("DFS Mean Length:", statistics.mean(dfs_lengths))
    print("DFS Variance Length:", statistics.variance(dfs_lengths))

    print("\nA* Times:", a_star_times)
    print("A* Mean Time:", statistics.mean(a_star_times))
    print("A* Variance Time:", statistics.variance(a_star_times))
    print("A* Lengths:", a_star_lengths)
    print("A* Mean Length:", statistics.mean(a_star_lengths))
    print("A* Variance Length:", statistics.variance(a_star_lengths))

if __name__ == "__main__":
    main()
```

Maze Generation: Generates multiple mazes with random start and goal points and a few barriers. Displays the initial maze and performs DFS analysis.

- width and height define the dimensions of the maze.
- num_mazes is the number of mazes to be generated and analyzed.
- Lists (dfs_times, dfs_lengths, a_star_times, a_star_lengths) will store the results for further analysis.

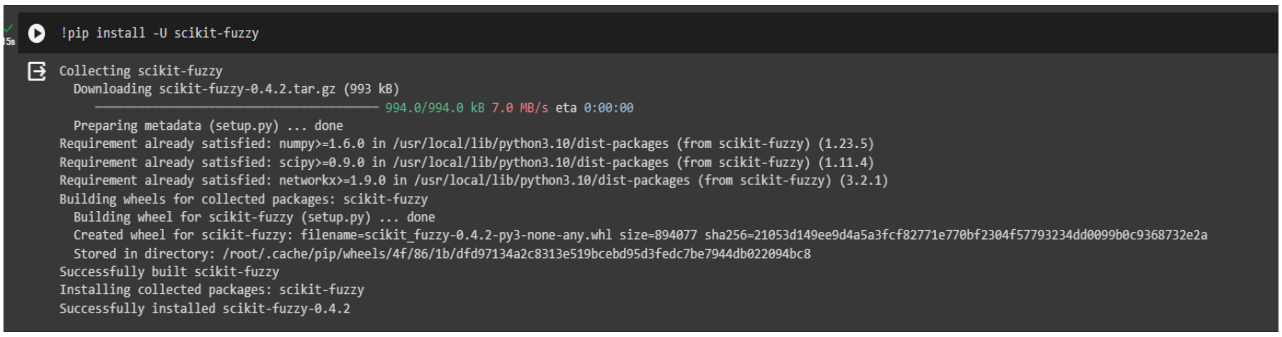
DFS Analysis: Performs Depth-First Search on each maze. Displays visited nodes, time to find the goal, and the final path.

Heuristic Cost Calculation: Calculates and displays the heuristic cost (Manhattan distance) for each node in the maze.

A Analysis*: Performs A* search on each maze. Displays visited nodes, time to find the goal, and the final path.

Results Analysis: This main function essentially generates random mazes, performs DFS and A* analysis on each maze, and then provides an overall analysis of the results, including mean times, variances, and lengths.

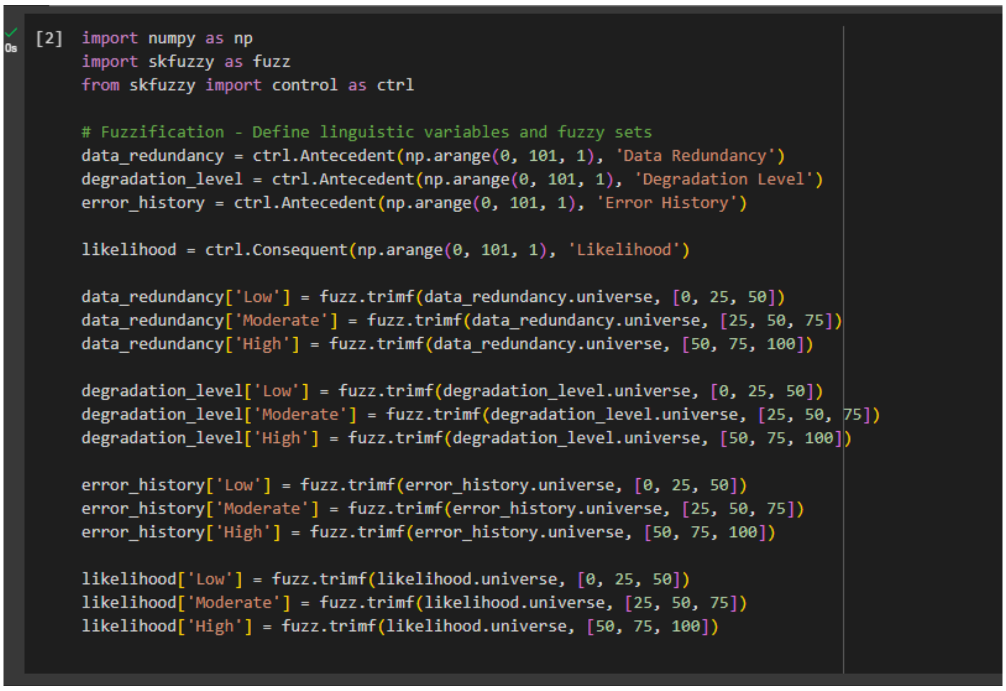
QUESTION 04



```
!pip install -U scikit-fuzzy
```

```
Collecting scikit-fuzzy
  Downloading scikit-fuzzy-0.4.2.tar.gz (993 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 994.0/994.0 kB 7.0 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from scikit-fuzzy) (1.23.5)
Requirement already satisfied: scipy>=0.9.0 in /usr/local/lib/python3.10/dist-packages (from scikit-fuzzy) (1.11.4)
Requirement already satisfied: networkx>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from scikit-fuzzy) (3.2.1)
Building wheels for collected packages: scikit-fuzzy
  Building wheel for scikit-fuzzy (setup.py) ... done
  Created wheel for scikit-fuzzy: filename=scikit_fuzzy-0.4.2-py3-none-any.whl size=894077 sha256=21053d149ee9d4a5a3fcf82771e770bf2304f57793234dd0099b0c9368732e2a
  Stored in directory: /root/.cache/pip/wheels/4f/86/1b/dfd97134a2c8313e519bcebd95d3fedc7be7944db022094bc8
Successfully built scikit-fuzzy
Installing collected packages: scikit-fuzzy
Successfully installed scikit-fuzzy-0.4.2
```

Scikit-fuzzy is a Python library for fuzzy logic operations. Fuzzy logic is a mathematical framework that deals with uncertainty and imprecision. It is particularly useful when working with systems that involve vague or subjective information



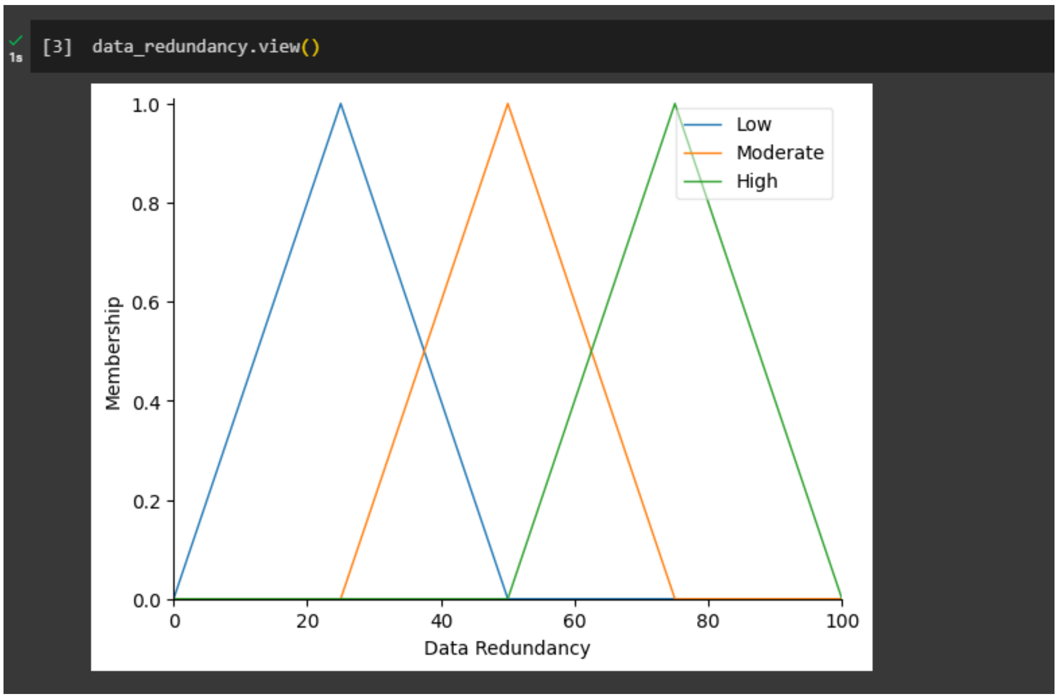
```
[2] import numpy as np
    import skfuzzy as fuzz
    from skfuzzy import control as ctrl

    # Fuzzification - Define linguistic variables and fuzzy sets
    data_redundancy = ctrl.Antecedent(np.arange(0, 101, 1), 'Data Redundancy')
    degradation_level = ctrl.Antecedent(np.arange(0, 101, 1), 'Degradation Level')
    error_history = ctrl.Antecedent(np.arange(0, 101, 1), 'Error History')

    likelihood = ctrl.Consequent(np.arange(0, 101, 1), 'Likelihood')

    data_redundancy['Low'] = fuzz.trimf(data_redundancy.universe, [0, 25, 50])
    data_redundancy['Moderate'] = fuzz.trimf(data_redundancy.universe, [25, 50, 75])
    data_redundancy['High'] = fuzz.trimf(data_redundancy.universe, [50, 75, 100])

    degradation_level['Low'] = fuzz.trimf(degradation_level.universe, [0, 25, 50])
    degradation_level['Moderate'] = fuzz.trimf(degradation_level.universe, [25, 50, 75])
    degradation_level['High'] = fuzz.trimf(degradation_level.universe, [50, 75, 100])

    error_history['Low'] = fuzz.trimf(error_history.universe, [0, 25, 50])
    error_history['Moderate'] = fuzz.trimf(error_history.universe, [25, 50, 75])
    error_history['High'] = fuzz.trimf(error_history.universe, [50, 75, 100])

    likelihood['Low'] = fuzz.trimf(likelihood.universe, [0, 25, 50])
    likelihood['Moderate'] = fuzz.trimf(likelihood.universe, [25, 50, 75])
    likelihood['High'] = fuzz.trimf(likelihood.universe, [50, 75, 100])
```

Fuzzification - Define Linguistic Variables and Fuzzy Sets

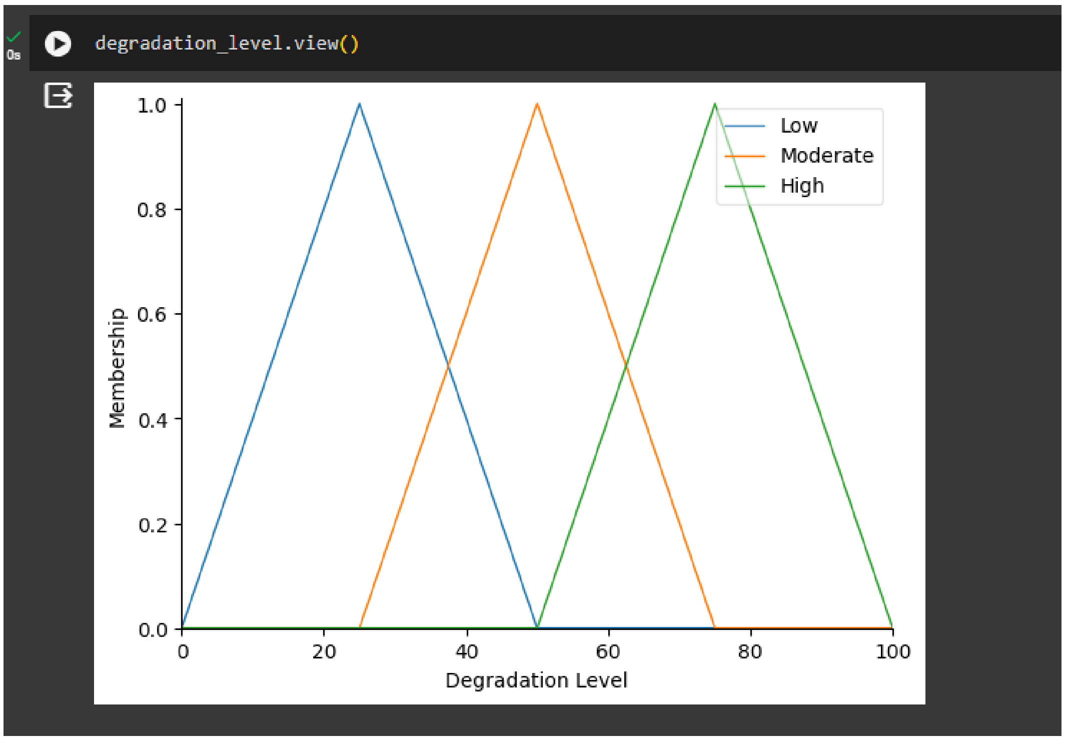
This section defines four linguistic variables: data_redundancy, degradation_level, error_history, and likelihood. The antecedent variables (data_redundancy, degradation_level, and error_history) represent input factors, while likelihood is the consequent variable representing the output.

Fuzzification - Define Fuzzy Sets

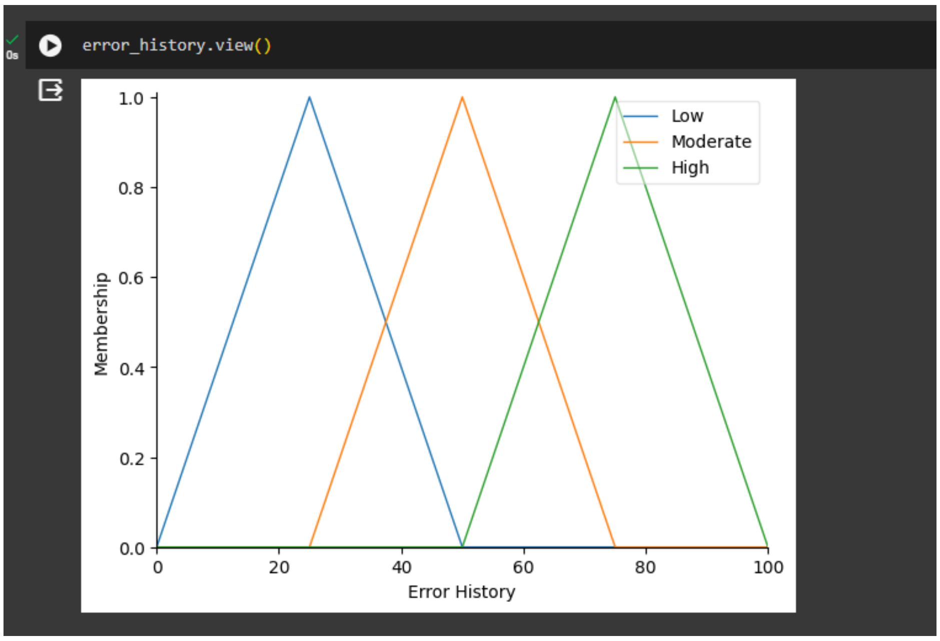
In this part, triangular fuzzy sets are defined for each linguistic variable. The trimf function is used to create triangular membership functions. These functions define the degree of membership for each linguistic term (e.g., 'Low', 'Moderate', 'High') within the universe of discourse (the range of possible values for each variable).



```
[3] data_redundancy.view()
```



```
degradation_level.view()
```



```
error_history.view()
```

The 3 Membership functions -

data_redundancy.view():

This statement generates a graphical representation (plot) of the membership functions for the linguistic variable data_redundancy. The x-axis represents the range of values for data redundancy (from 0 to 100), and the y-axis represents the membership degrees for each linguistic term ('Low,' 'Moderate,' 'High'). You'll see three triangular membership functions corresponding to the fuzzy sets 'Low,' 'Moderate,' and 'High' on the graph.

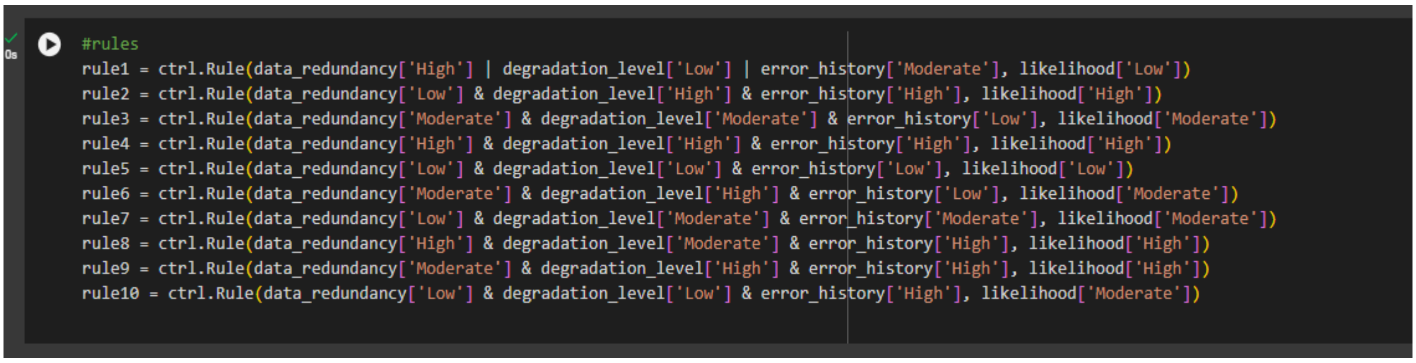
degradation_level.view():

Similar to the first statement, this generates a plot for the linguistic variable degradation_level. It shows the membership functions for 'Low,' 'Moderate,' and 'High' degradation levels across the specified input range.

error_history.view():

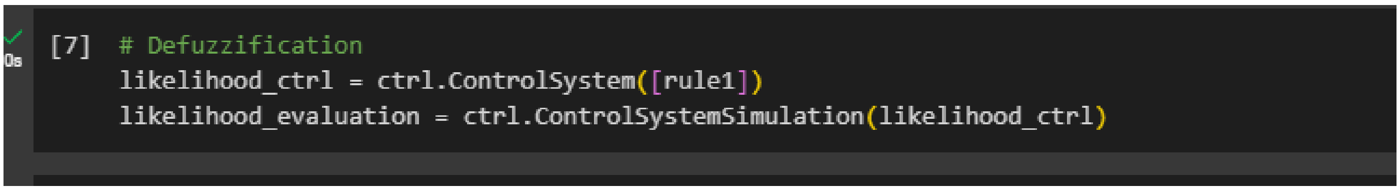
This statement creates a visualization for the linguistic variable error_history. It displays the membership functions for 'Low,' 'Moderate,' and 'High' error history values.

Fuzzy Rules



```
#rules
rule1 = ctrl.Rule(data_redundancy['High'] | degradation_level['Low'] | error_history['Moderate'], likelihood['Low'])
rule2 = ctrl.Rule(data_redundancy['Low'] & degradation_level['High'] & error_history['High'], likelihood['High'])
rule3 = ctrl.Rule(data_redundancy['Moderate'] & degradation_level['Moderate'] & error_history['Low'], likelihood['Moderate'])
rule4 = ctrl.Rule(data_redundancy['High'] & degradation_level['High'] & error_history['High'], likelihood['High'])
rule5 = ctrl.Rule(data_redundancy['Low'] & degradation_level['Low'] & error_history['Low'], likelihood['Low'])
rule6 = ctrl.Rule(data_redundancy['Moderate'] & degradation_level['High'] & error_history['Low'], likelihood['Moderate'])
rule7 = ctrl.Rule(data_redundancy['Low'] & degradation_level['Moderate'] & error_history['Moderate'], likelihood['Moderate'])
rule8 = ctrl.Rule(data_redundancy['High'] & degradation_level['Moderate'] & error_history['High'], likelihood['High'])
rule9 = ctrl.Rule(data_redundancy['Moderate'] & degradation_level['High'] & error_history['High'], likelihood['High'])
rule10 = ctrl.Rule(data_redundancy['Low'] & degradation_level['Low'] & error_history['High'], likelihood['Moderate'])
```

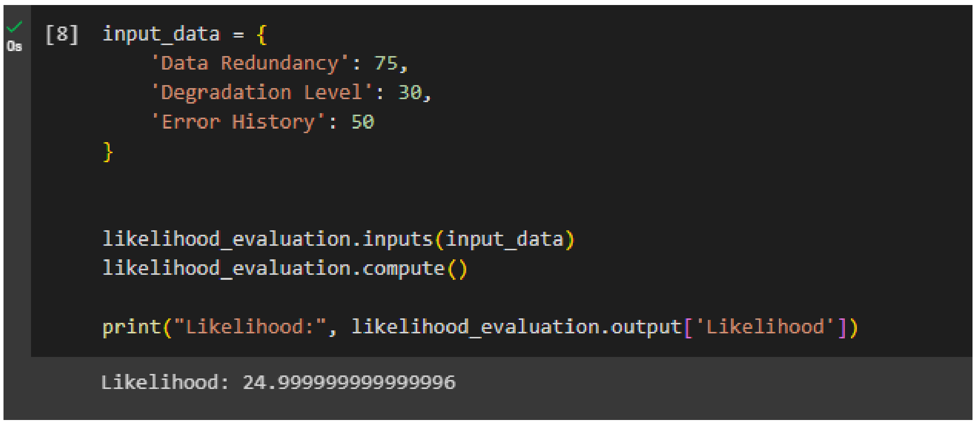
These rules define the relationships between the input linguistic variables (data_redundancy, degradation_level, error_history) and the output variable (likelihood) in a fuzzy logic system. Each rule consists of antecedents (conditions) and a consequent (result). The ctrl.Rule class from scikit-fuzzy is used to create these rules.



```
[7] # Defuzzification
    likelihood_ctrl = ctrl.ControlSystem([rule1])
    likelihood_evaluation = ctrl.ControlSystemSimulation(likelihood_ctrl)
```

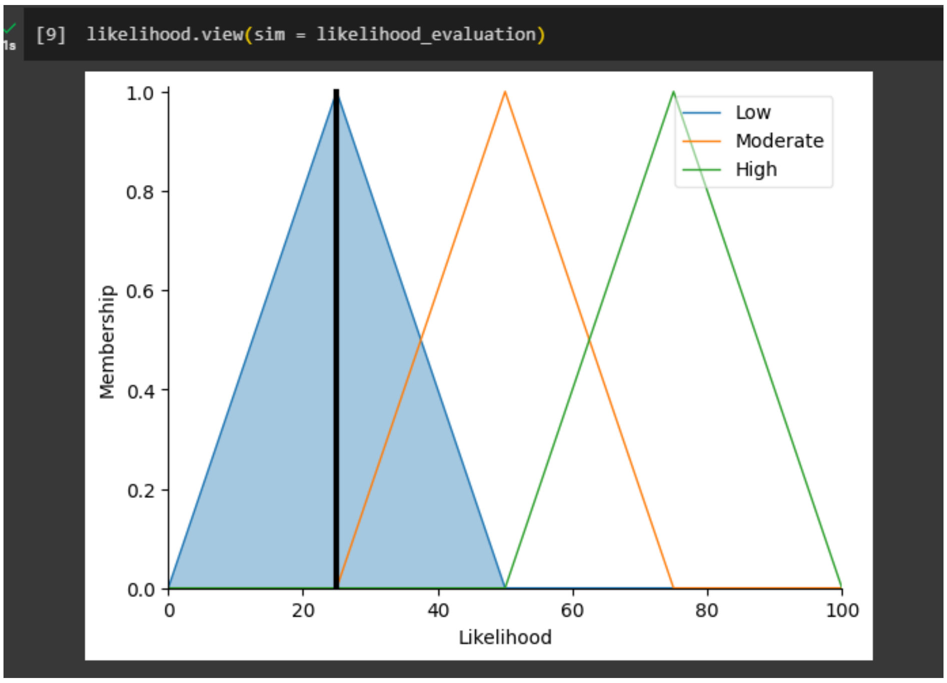
A ControlSystem is created with a list of rules. In this case, the list contains only rule1. This control system represents the fuzzy logic system that combines the defined rules for the input variables (data_redundancy, degradation_level, error_history) and the output variable (likelihood).

A ControlSystemSimulation object is created based on the initialized control system (likelihood_ctrl). This simulation object is used to evaluate the fuzzy system with specific input values.



```
[8] input_data = {
        'Data Redundancy': 75,
        'Degradation Level': 30,
        'Error History': 50
    }


    likelihood_evaluation.inputs(input_data)
    likelihood_evaluation.compute()

    print("Likelihood:", likelihood_evaluation.output['Likelihood'])

    Likelihood: 24.999999999999996
```

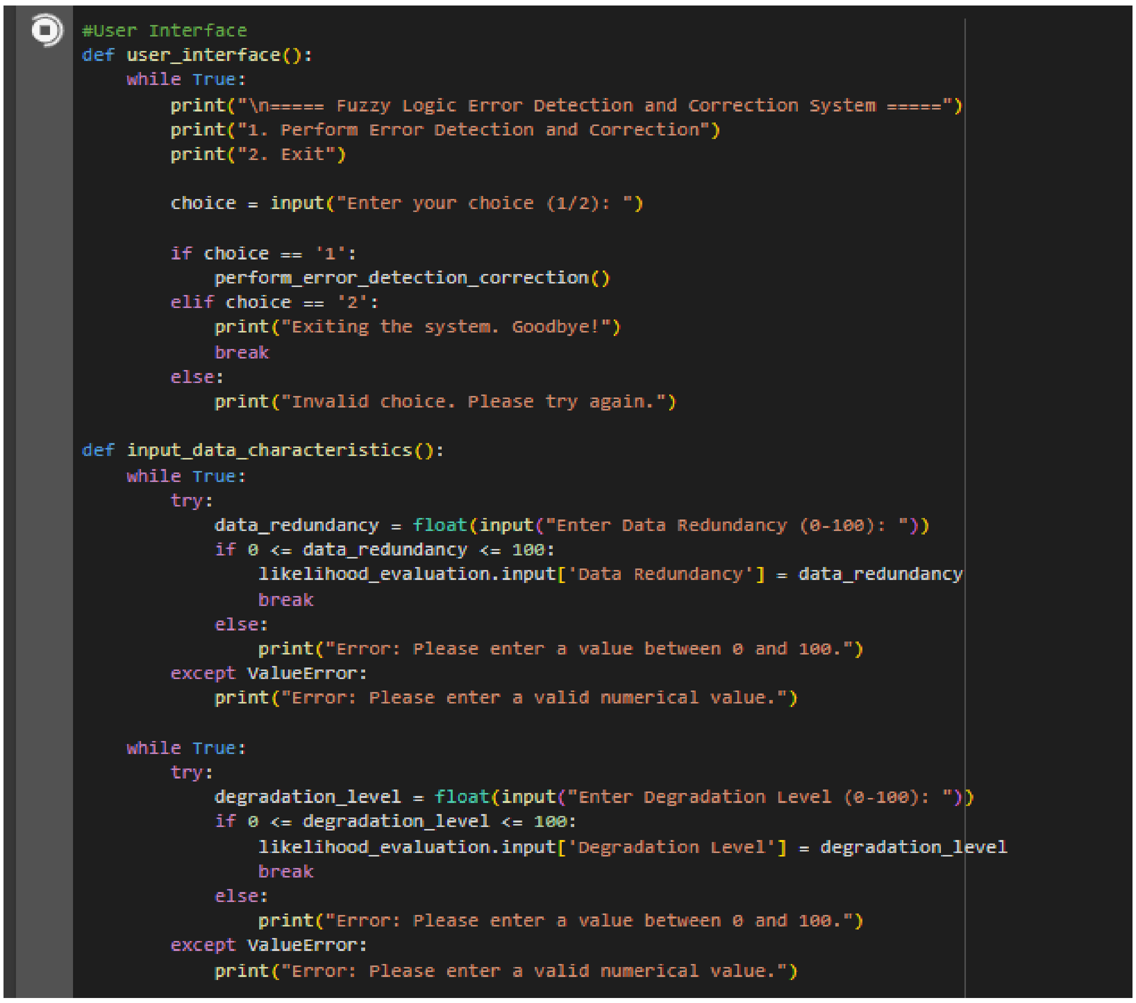
In this you are setting input values for the antecedent variables ('Data Redundancy', 'Degradation Level', 'Error History') using the input_data dictionary. Then, you use the inputs() method to update the input values in the **likelihood_evaluation** simulation object. Finally, you call the compute() method to evaluate the fuzzy system and obtain the defuzzified output for the consequent variable ('Likelihood'). The result is then printed



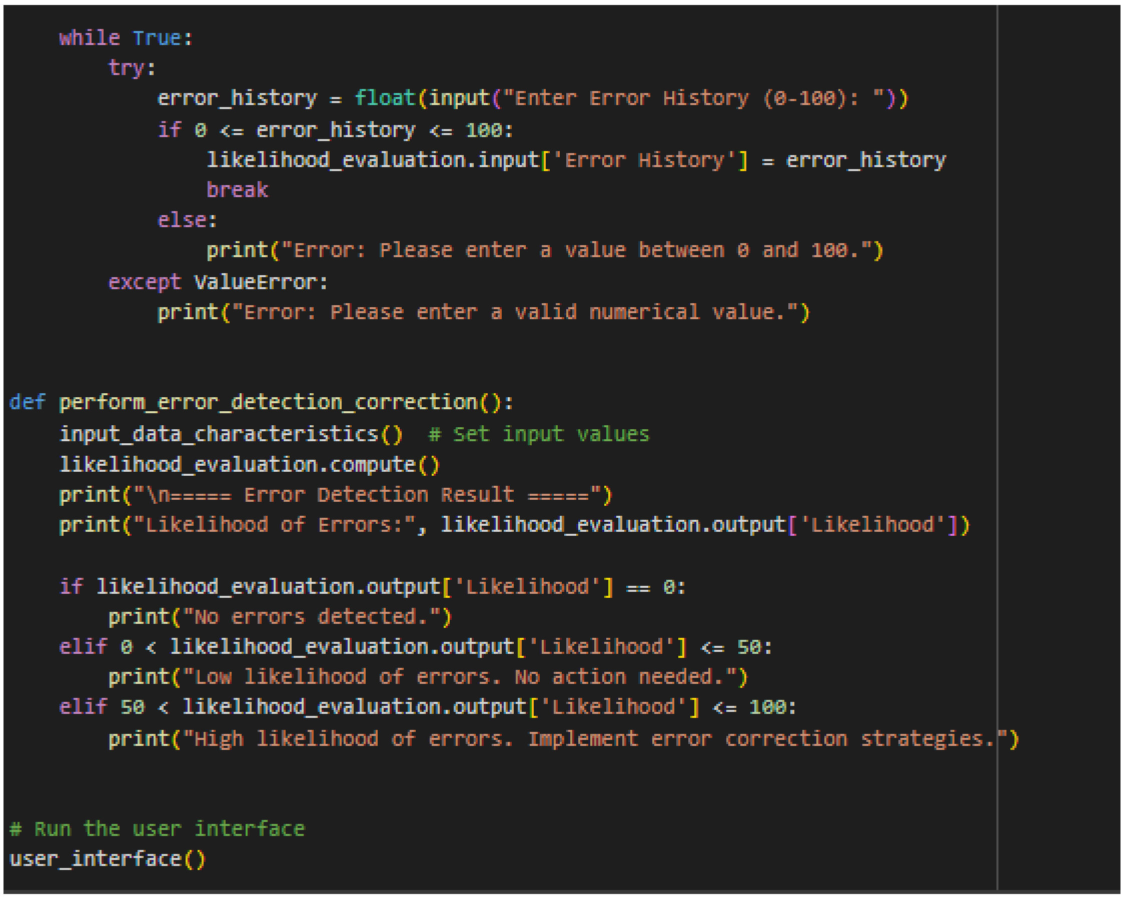
```
[9]  likelihood.view(sim = likelihood_evaluation)
```

.

The **likelihood.view(sim=likelihood_evaluation**) command is used to visualize the membership functions of the linguistic terms in the output variable ('Likelihood') after the fuzzy system has been evaluated with specific input values. This visualization helps you understand how the membership values for each linguistic term vary based on the input data.



```python
#User Interface
def user_interface():
    while True:
        print("\n===== Fuzzy Logic Error Detection and Correction System =====")
        print("1. Perform Error Detection and Correction")
        print("2. Exit")

        choice = input("Enter your choice (1/2): ")

        if choice == '1':
            perform_error_detection_correction()
        elif choice == '2':
            print("Exiting the system. Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")

def input_data_characteristics():
    while True:
        try:
            data_redundancy = float(input("Enter Data Redundancy (0-100): "))
            if 0 <= data_redundancy <= 100:
                likelihood_evaluation.input['Data Redundancy'] = data_redundancy
                break
            else:
                print("Error: Please enter a value between 0 and 100.")
        except ValueError:
            print("Error: Please enter a valid numerical value.")

    while True:
        try:
            degradation_level = float(input("Enter Degradation Level (0-100): "))
            if 0 <= degradation_level <= 100:
                likelihood_evaluation.input['Degradation Level'] = degradation_level
                break
            else:
                print("Error: Please enter a value between 0 and 100.")
        except ValueError:
            print("Error: Please enter a valid numerical value.")
```



```python
    while True:
        try:
            error_history = float(input("Enter Error History (0-100): "))
            if 0 <= error_history <= 100:
                likelihood_evaluation.input['Error History'] = error_history
                break
            else:
                print("Error: Please enter a value between 0 and 100.")
        except ValueError:
            print("Error: Please enter a valid numerical value.")


def perform_error_detection_correction():
    input_data_characteristics()  # Set input values
    likelihood_evaluation.compute()
    print("\n===== Error Detection Result =====")
    print("Likelihood of Errors:", likelihood_evaluation.output['Likelihood'])

    if likelihood_evaluation.output['Likelihood'] == 0:
        print("No errors detected.")
    elif 0 < likelihood_evaluation.output['Likelihood'] <= 50:
        print("Low likelihood of errors. No action needed.")
    elif 50 < likelihood_evaluation.output['Likelihood'] <= 100:
        print("High likelihood of errors. Implement error correction strategies.")


# Run the user interface
user_interface()
```

Function (user_interface):

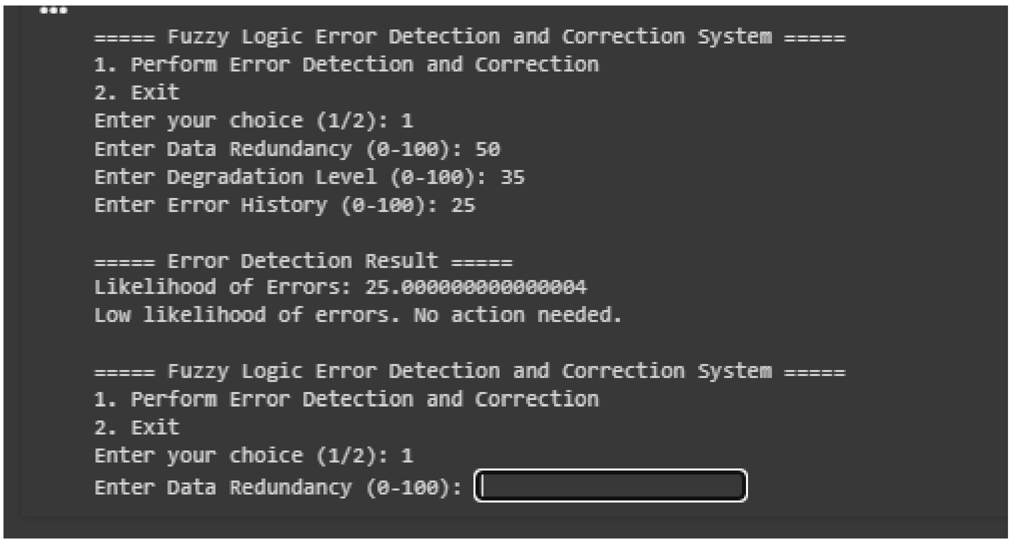
This function represents the main user interface loop. It displays a menu with three options: input data characteristics, perform error detection and correction, or exit the system. It takes user input and calls the appropriate functions based on the selected choice.

Input Data Characteristics Function (input_data_characteristics):

This function prompts the user to input data characteristics, specifically data redundancy, degradation level, and error history. It validates the input to ensure it is a numerical value within the specified range (0-100). It sets the input values in the likelihood_evaluation simulation object.

Perform Error Detection and Correction Function (perform_error_detection_correction):

This function calls input_data_characteristics() to set input values. It computes the fuzzy system using likelihood_evaluation.compute(). It prints the likelihood of errors based on the fuzzy system's output. It makes decisions on whether to implement error correction strategies based on the likelihood.



```
===== Fuzzy Logic Error Detection and Correction System =====
1. Perform Error Detection and Correction
2. Exit
Enter your choice (1/2): 1
Enter Data Redundancy (0-100): 50
Enter Degradation Level (0-100): 35
Enter Error History (0-100): 25

===== Error Detection Result =====
Likelihood of Errors: 25.000000000000004
Low likelihood of errors. No action needed.

===== Fuzzy Logic Error Detection and Correction System =====
1. Perform Error Detection and Correction
2. Exit
Enter your choice (1/2): 1
Enter Data Redundancy (0-100):
```

REFERENCES

1) Li, Z., Yang, M.C. and Ramani, K. (2008). A methodology for engineering ontology acquisition and validation. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 23(1), pp.37–51. Available at : https://doi.org/10.1017/s0890060409000092.[Accessed 5 Dec. 2023]

2) Hagedorn, T., Bone, M., Kruse, B., Grosse, I. and Blackburn, M. (2020). Knowledge Representation with Ontologies and Semantic Web Technologies to Promote Augmented and Artificial Intelligence in Systems Engineering. *INSIGHT*, 23(1), pp.15–20. Available at : https://doi.org/10.1002/inst.12279 .[Accessed 5 Dec. 2023]

3) Jiang, J.J. and Conrath, D.W. (1997). *Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy*. [online] arXiv.org. Avaliable at: https://doi.org/10.48550/arXiv.cmp-lg/9709008 .[Accessed 6 Dec. 2023]

4) Uschold, M. and Gruninger, M. (2004). Ontologies and semantics for seamless connectivity. *ACM SIGMOD Record*, 33(4), pp.58–64. Available at :https://doi.org/10.1145/1041410.1041420.[Accessed 5 Dec. 2023]

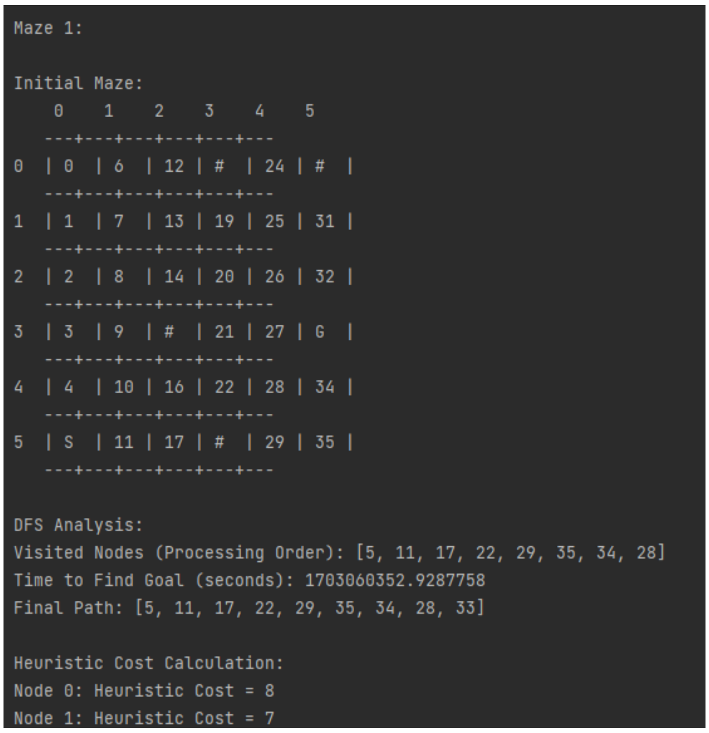
5) ieeexplore.ieee.org. (n.d.). *Ontology-Driven Learning of Bayesian Network for Causal Inference and Quality Assurance in Additive Manufacturing | IEEE Journals & Magazine | IEEE Xplore*. [online] Available at: https://ieeexplore.ieee.org/document/9457187 [Accessed 17 Dec. 2023].

6) Asme.org. (2023). Available at: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-abstract/IDETC-CIE2000/35142/17/1093570 [Accessed 18 Dec. 2023].

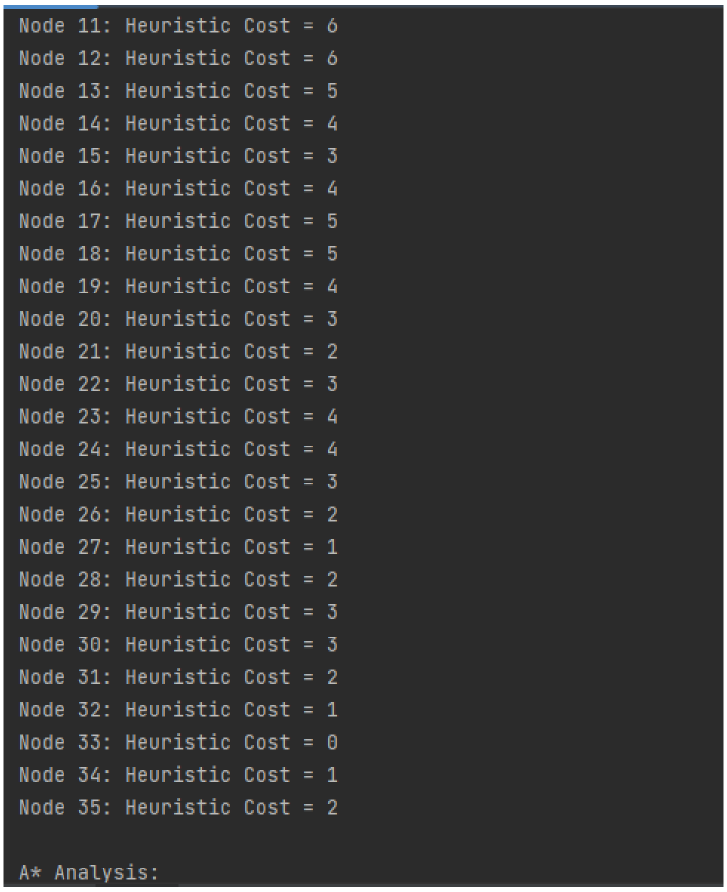
7) Kitamura, Y. and Mizoguchi, R. (2004). Ontology-based systematization of functional knowledge. *Journal of Engineering Design*, 15(4), pp.327–351. Available at : https://doi.org/10.1080/09544820410001697163[Accessed 15 Dec. 2023]

8) Linked Data for Professional Education. (n.d.). *Ontology Development 101: A Guide to Creating Your First Ontology*. [online] Available at: https://shorturl.at/lmGP7 [Accessed 20 Dec. 2023]

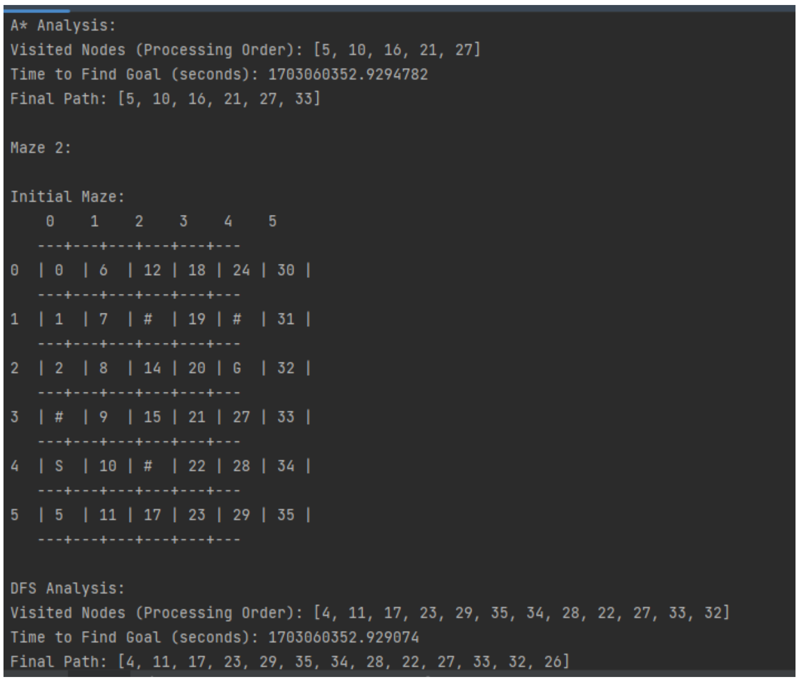
APPENDICES



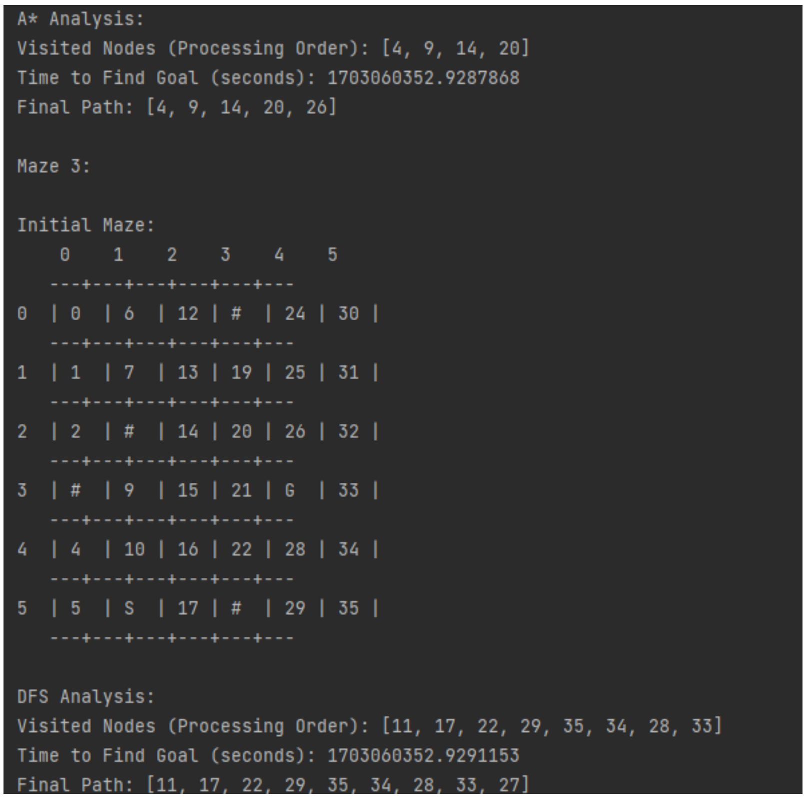
```
Maze 1:

Initial Maze:
    0    1    2    3    4    5
   ---+---+---+---+---+---
0  | 0  | 6  | 12 | #  | 24 | #  |
   ---+---+---+---+---+---
1  | 1  | 7  | 13 | 19 | 25 | 31 |
   ---+---+---+---+---+---
2  | 2  | 8  | 14 | 20 | 26 | 32 |
   ---+---+---+---+---+---
3  | 3  | 9  | #  | 21 | 27 | G  |
   ---+---+---+---+---+---
4  | 4  | 10 | 16 | 22 | 28 | 34 |
   ---+---+---+---+---+---
5  | S  | 11 | 17 | #  | 29 | 35 |
   ---+---+---+---+---+---

DFS Analysis:
Visited Nodes (Processing Order): [5, 11, 17, 22, 29, 35, 34, 28]
Time to Find Goal (seconds): 1703060352.9287758
Final Path: [5, 11, 17, 22, 29, 35, 34, 28, 33]

Heuristic Cost Calculation:
Node 0: Heuristic Cost = 8
Node 1: Heuristic Cost = 7
```



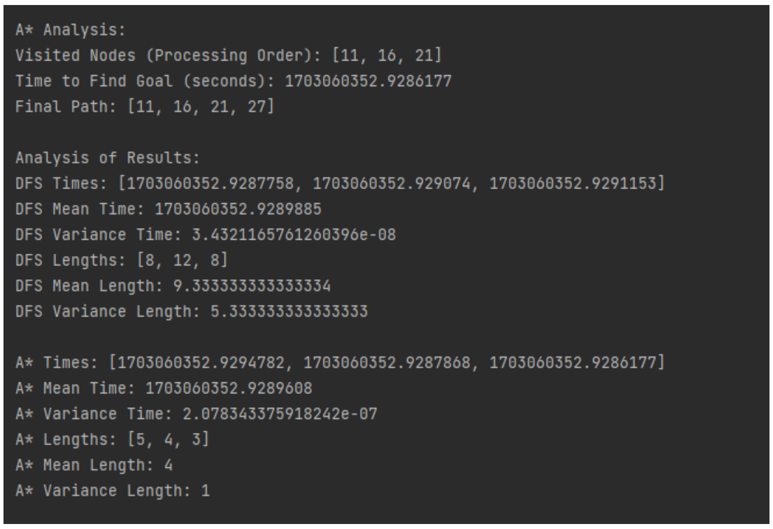
```
Node 11: Heuristic Cost = 6
Node 12: Heuristic Cost = 6
Node 13: Heuristic Cost = 5
Node 14: Heuristic Cost = 4
Node 15: Heuristic Cost = 3
Node 16: Heuristic Cost = 4
Node 17: Heuristic Cost = 5
Node 18: Heuristic Cost = 5
Node 19: Heuristic Cost = 4
Node 20: Heuristic Cost = 3
Node 21: Heuristic Cost = 2
Node 22: Heuristic Cost = 3
Node 23: Heuristic Cost = 4
Node 24: Heuristic Cost = 4
Node 25: Heuristic Cost = 3
Node 26: Heuristic Cost = 2
Node 27: Heuristic Cost = 1
Node 28: Heuristic Cost = 2
Node 29: Heuristic Cost = 3
Node 30: Heuristic Cost = 3
Node 31: Heuristic Cost = 2
Node 32: Heuristic Cost = 1
Node 33: Heuristic Cost = 0
Node 34: Heuristic Cost = 1
Node 35: Heuristic Cost = 2

A* Analysis:
```



```
A* Analysis:
Visited Nodes (Processing Order): [5, 10, 16, 21, 27]
Time to Find Goal (seconds): 1703060352.9294782
Final Path: [5, 10, 16, 21, 27, 33]

Maze 2:

Initial Maze:
    0    1    2    3    4    5
   ---+---+---+---+---+---
0  | 0  | 6  | 12 | 18 | 24 | 30 |
   ---+---+---+---+---+---
1  | 1  | 7  | #  | 19 | #  | 31 |
   ---+---+---+---+---+---
2  | 2  | 8  | 14 | 20 | G  | 32 |
   ---+---+---+---+---+---
3  | #  | 9  | 15 | 21 | 27 | 33 |
   ---+---+---+---+---+---
4  | S  | 10 | #  | 22 | 28 | 34 |
   ---+---+---+---+---+---
5  | 5  | 11 | 17 | 23 | 29 | 35 |
   ---+---+---+---+---+---

DFS Analysis:
Visited Nodes (Processing Order): [4, 11, 17, 23, 29, 35, 34, 28, 22, 27, 33, 32]
Time to Find Goal (seconds): 1703060352.929074
Final Path: [4, 11, 17, 23, 29, 35, 34, 28, 22, 27, 33, 32, 26]
```



```
A* Analysis:
Visited Nodes (Processing Order): [4, 9, 14, 20]
Time to Find Goal (seconds): 1703060352.9287868
Final Path: [4, 9, 14, 20, 26]

Maze 3:

Initial Maze:
    0    1    2    3    4    5
   ---+---+---+---+---+---
0  | 0  | 6  | 12 | #  | 24 | 30 |
   ---+---+---+---+---+---
1  | 1  | 7  | 13 | 19 | 25 | 31 |
   ---+---+---+---+---+---
2  | 2  | #  | 14 | 20 | 26 | 32 |
   ---+---+---+---+---+---
3  | #  | 9  | 15 | 21 | G  | 33 |
   ---+---+---+---+---+---
4  | 4  | 10 | 16 | 22 | 28 | 34 |
   ---+---+---+---+---+---
5  | 5  | S  | 17 | #  | 29 | 35 |
   ---+---+---+---+---+---

DFS Analysis:
Visited Nodes (Processing Order): [11, 17, 22, 29, 35, 34, 28, 33]
Time to Find Goal (seconds): 1703060352.9291153
Final Path: [11, 17, 22, 29, 35, 34, 28, 33, 27]
```



```
A* Analysis:
Visited Nodes (Processing Order): [11, 16, 21]
Time to Find Goal (seconds): 1703060352.9286177
Final Path: [11, 16, 21, 27]

Analysis of Results:
DFS Times: [1703060352.9287758, 1703060352.929074, 1703060352.9291153]
DFS Mean Time: 1703060352.9289885
DFS Variance Time: 3.4321165761260396e-08
DFS Lengths: [8, 12, 8]
DFS Mean Length: 9.333333333333334
DFS Variance Length: 5.333333333333333

A* Times: [1703060352.9294782, 1703060352.9287868, 1703060352.9286177]
A* Mean Time: 1703060352.9289608
A* Variance Time: 2.078343375918242e-07
A* Lengths: [5, 4, 3]
A* Mean Length: 4
A* Variance Length: 1
```