Intelligent Systems

Exercise sheet 2
Propositional Logic

This exercise sheet is about propositional logic, you'll have to implement an object model for the syntax, an interpretation for the semantics and an algorithm for reasoning. Please read all instructions before you start implementing, since the exercises depend on each other.

You may solve the exercises in groups up to the size of two people; everybody must however be able to explain all parts of the implementation. You may use the programming language of your choice, please refrain from very esoteric languages. If you are not able to solve an exercise (X) which is needed as basis for another exercise (Y), you are allowed to take exercise X from somebody else to solve exercise Y, but you will not get points for exercise X in that case.

Exercise 1 (Syntax) (10 points)[1]

Create an object model[2] that is able to represent the syntax of propositional logic. You do not need to implement a parser (the object model can be created programmatically), but the object model must be serializable[3] (you can use textual representation, i.e. "and" representing $\wedge$).

Test your implementation by representing a sentence like e.g. $A = (\neg p \rightarrow q) \leftrightarrow (p \vee q)$ and test your serialization functionality.

Exercise 2 (Semantics) (15 points)

Write a function to apply an interpretation[4] to a sentence represented in your object model. Furthermore, your implementation should be able to compute the truth value of that sentence after assigning the interpretation. Use the example sentence and interpretation from slide 18 to test your implementation.

Can you use your implementation to test whether your sentence is a tautology?

Exercise 3 (Reasoning) (15 points)

Implement propositional resolution for your object model. You do not need to implement an algorithm to transform arbitrary expressions into clausal form, you need to test it only on expressions in CNF.

Test your implementation with the example from slide 56.

---

[1] Hint: have a look at exercise 1 from exercise sheet 3, the OMs overlap.
[2] E.g. a Java class hierarchy that is capable of representing every possible sentence in propositional logic in Java objects.
[3] i.e., serialized to a textual representation on the console
[4] The interpretation could be represented by a Map<Proposition, Boolean>, i.e., when applied to a sentence it applies a truth value to all propositions in the sentence