CAP 4630 - Introduction to AI (Spring 2023)
Project 3 - Due Date: Mar. 31, 2023 Friday 11:59 pm

For this team Python project, you will form your team of **up to 3**, and design and build a knowledge-based intelligent system that collects user preferences and reasons about them.

1 Requirements

1. The system should have an easy-to-use GUI, implemented using Tkinter package[1], for collecting from the user attributes and their values, hard constraints, and preferences. The system should at least allow reading in these data from files. (See section 3 for formats of these files.) It also may ask the user interactively to enter and provide these data.
    - Attributes (A) in this project are going to be **binary**.
    - Hard constraints (H) are represented as propositional formulas in the **Conjunctional Normal Form (CNF)**.
    - The system should support preferences (T) in the preference languages of **penalty logic**, **possibilistic logic**, and **qualitative choice logic**. Formulas involved in the preference theories (i.e., the φ's and the ψ's) are of **CNF** as well.
2. The system should support the following reasoning tasks:
    - Existence of feasible objects: decide whether there are feasible objects w.r.t H, that is, whether there are models of H that are truth assignments making H true.
    - Exemplification: generate, if possible, two random feasible objects, and show the preference between the two (strict preference, equivalence, or incomparison) w.r.t T.
    - Optimization: find one optimal object w.r.t T.
    - Omni-optimization: find all optimal objects w.r.t T.
3. For testing, the system should solve an instance, developed by you, that contains at least 6 hard constraints and at least 3 preference rules per each logic over at least 8 attributes. Also use this instance when demonstrating your system.

[1]See `https://realpython.com/python-gui-tkinter/` and `https://www.python-course.eu/tkinter_labels.php` for helpful references.

- In ExampleCase a small example instance is provided that has three attributes, one constraint, two rules in a penalty logic theory, two rules in possibilistic logic, and four rules in qualitative choice logic. Expected results are provided for this small instance too.

4. By **Mar. 10 noon**, each team must have ONE team member to send me an email describing who are the members of the team. Failure of this will result in deduction in the project grade.

5. By **Mar. 24 midnight**, each team must have ONE team member to email me to discuss your team's progress. In the email, you will describe what your team has done for this project and what the team plans to do, and your questions if any. This email also should describe which team member has done and will do which part of the project. Attached to the email should be a screen shot of your team's GUI or at least a design of it, in case you will not have implemented it. Once I read your emails, I will respond with my comments to help you towards completing this project. Failure of this will result in deduction in the project grade.

2 Deliverables

Each team will zip the following into [your-last-names-in-team]_Project3.zip and have ONE team member to submit ONE copy of your project to Canvas.

1. A directory called "TestCase" that contains text files of the instance your team created for testing. File names should be straightforward, similar to those in ExampleCase.
2. A directory called "src" that contains all the source codes.
3. A README file that contains instructions to build and run your system.
4. A PDF report that describes how your system works and shows the testing results using the test instance created by your team (e.g., screen shots of various steps).

3 File Formats

3.1 Attributes File

Each row in this file describes one attribute and its two values. Attributes and their values are lower-case strings. Each attribute is followed by a colon, which is followed by two values separated by a comma. Use different strings for values to avoid confusion. An example format is as follows.

```
appetizer:  soup, salad
entree:  beef, fish
drink:  beer, wine
dissert:  cake, ice-cream
...
```

3.2 Hard Constraints File

Hard constraints essentially are a long CNF formula, so in this file each row is a clause, which is a disjunction of literals that could be positive or negative. A literal is positive if it is a value of some attribute, and negative if it is the negation of a value of some attribute. Inside a clause, negation (NOT) and disjunction (OR) are all upper cases, and values are again lower cases. An example format is as follows.

```
NOT soup OR NOT ice-cream
NOT beef OR salad
...
```

3.3 Preferences File

3.3.1 Penalty Logic

In this file each row describes one penalty logic rule which is a comma-separated pair. The first part is a CNF propositional formula using values and connectives (AND, OR, and NOT), and the second part is the penalty value. An example format is as follows.

```
fish AND wine, 10
wine OR cake, 6
beer AND beer OR beef AND NOT soup, 7
...
```

3.3.2 Possibilistic Logic

In this file each row describes one possibilistic logic rule which is a comma-separated pair. The first part is a CNF propositional formula using values and connectives (AND, OR, and NOT), and the second part is the importance value. An example format is as follows.

```
fish AND wine, 0.8
wine OR cake, 0.5
beer AND beer OR beef AND NOT soup, 0.6
...
```

3.3.3 Qualitative Choice Logic

In this file each row describes one qualitative choice logic rule which is of format "ϕ_1 BT $\ldots \phi_n$ IF ψ," where BT reads better than ($>$) and IF is follwed by the condition formula ψ. Formulas here also are CNF. An example format is as follows.

```
fish BT beef IF
wine BT beer IF fish
beef AND beer BT fish AND beer BT beef AND wine IF
cake BT ice-cream IF soup
...
```

("BT" stands for "better than.")