Block World Planner

The goal of the project is to define the grammar of the language you want to be able to process. Then write a parser so as to produce a parse tree of an input sentences. From the parse tree decide whether it is a statement, a query, or a command. If statement translate it into a fact to be asserted in (Prolog) knowledge base, if query translate it into (Prolog) query and execute it. If command translate it into a goal and invoke a (Prolog) planner to generate a plan of action to achieve the goal.

Once a sentence by the user has entered, the program recognizes if it is a statement, a query or a command using the Natural Language Processing (NLP). NLP is a subfield of artificial intelligence concerned with the interactions between computers and human (natural) languages. For this scope, a Python library called Natural Language Toolkit (NLTK) [http://www.ntlk.org/] is used. The library guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more.
Once the user has entered the sentence, it is split into tokens and is transformed in a part-of-speech tagged tree structure. Once created the tagged tree, the program analyzes it in order to understand what the initial sentence is.
Thanks to different functions, the program can act in different ways. In case of an assertion the program has to add facts from it to the knowledge base, while in case of a query it has to answer it according to the knowledge base. In case of a command, instead, it has to run the planner algorithm in order to reach the required goal.
These actions have been implemented with another Python library called PySWIP [https://github.com/yuce/pwswip]. PySwip is a Python - SWI-Prolog bridge enabling to query SWI-Prolog in your Python programs. It features an SWI-Prolog foreign language interface, a utility class that makes it easy querying with Prolog.
The use of this library was necessary due to the decision of use Prolog in order to implement the Planner Algorithm. When an assertion is entered by the user, the program uses PySWIP in order to add new identified facts to the SWI-Prolog knowledge base, while when a query is entered, always using PySWIP the program forwards the correct equivalent query to SWI-Prolog interpreter, so it can answer correctly according to the current knowledge base. Instead when a command is requested by the user, it is transformed into a goal to reach and the execution of the Planner is required through PySWIP in order to reach this goal.
Once the Planner has performed the actions, an output is showed in order to inform the user about the results.
The program is able to process lots of sentences, provided that they are correct in the domain of interest. The only constraint is that proper nouns (e.g. B, C) must start in upper case.

Here a list of tested sentences which can be used (terms in [] are optional):

Assertions:

- [the] [block] B is blue
- [the] [red] [block] B is on the table
- [the] [red] [block] B is on [the] [green] [block] C
- there is a [red] block B on the table
- there is a [red] block B on the [green] block C

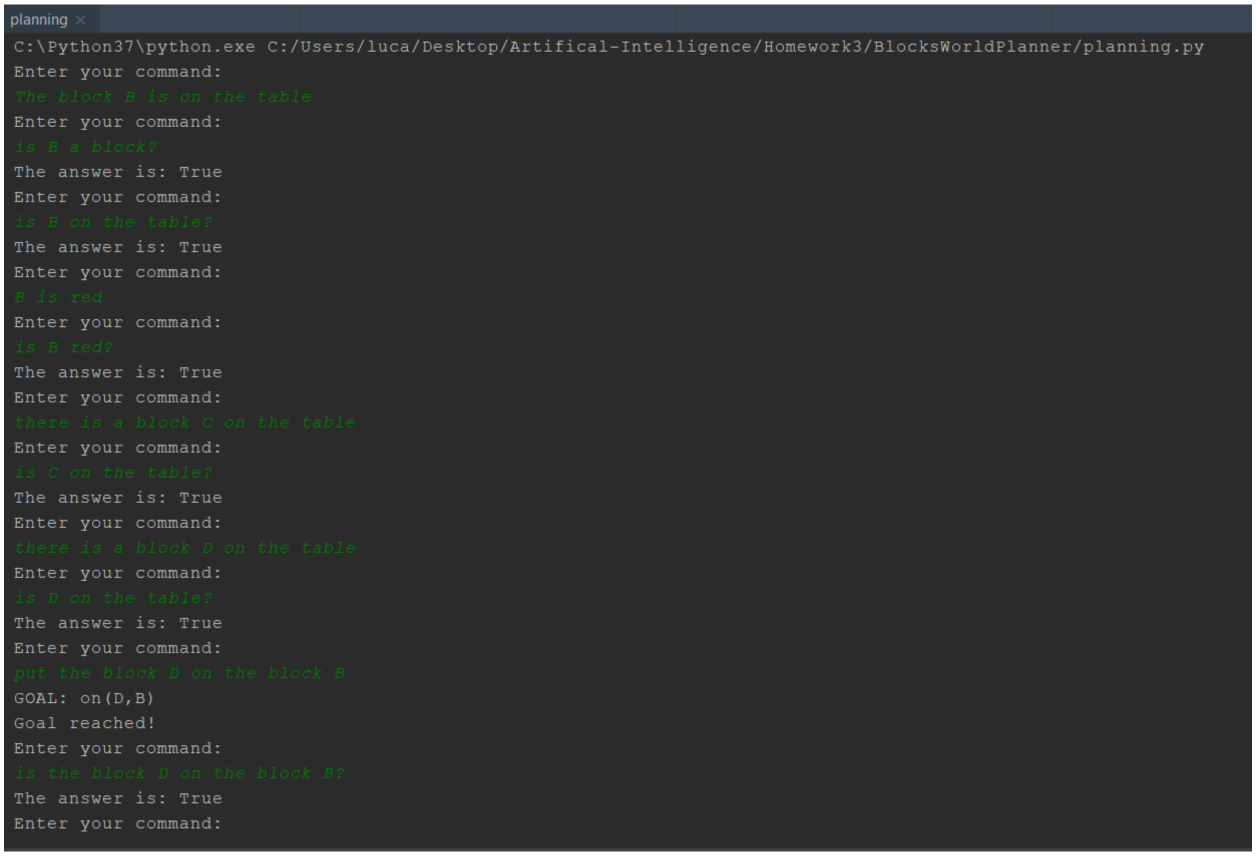
Queries:

- is B a [red] block?
- is [the] [block] B red?
- Is [the] [red] [block] B on the table?
- Is [the] [green] [block] C on [the] [red] [block] B?

Commands:

- put [the] [green] [block] C on the table
- put [the] [green] [block] C on [the] [red] [block] B
- move the [green] block C on [the] [red] [block] B
- shift the [green] block C on [the] [red] [block] B

The following picture shows the outputs of different sentences inserted by a possible user.



```
C:\Python37\python.exe C:/Users/luca/Desktop/Artifical-Intelligence/Homework3/BlocksWorldPlanner/planning.py
Enter your command:
The block B is on the table
Enter your command:
is B a block?
The answer is: True
Enter your command:
is B on the table?
The answer is: True
Enter your command:
B is red
Enter your command:
is B red?
The answer is: True
Enter your command:
there is a block C on the table
Enter your command:
is C on the table?
The answer is: True
Enter your command:
there is a block D on the table
Enter your command:
is D on the table?
The answer is: True
Enter your command:
put the block D on the block B
GOAL: on(D,B)
Goal reached!
Enter your command:
is the block D on the block B?
The answer is: True
Enter your command:
```

Figure 1. Outputs of different sentences