

CMPT 310

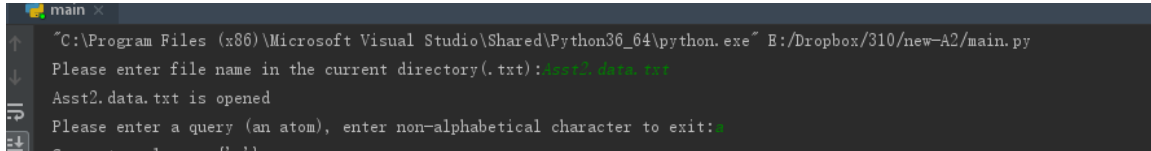
Assignment2-Documentation

Propositional Logic

Author: Sen Lin - 301250505 – sla248@sfu.ca

### **Running the program:**

Please see screen shot:



```
main
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe E:/Dropbox/310/new-A2/main.py
Please enter file name in the current directory(.txt): Asst2.data.txt
Asst2.data.txt is opened
Please enter a query (an atom), enter non-alphabetical character to exit:
```

Please put the test data with **main.py** in the same folder, and run **main.py**

Just enter the file name, without quotation mark

Please enter a single item as query.

The program automatically reruns, so you can enter another query after current query is done.

Exit program by entering any non-alphabetical character.

### **Test Data:**

Asst2.data.txt – original data knowledge base given

Asst2.loop.txt – looping data knowledge base, found in slides.

Asst2.pair.txt – pair data knowledge base

Asst2.self.txt – self containing data [p,p,q], with no other way to prove p

Asst2.self2.txt – self containing data [p,p,q], with another way to prove p

### **Approach:**

The program follows the pseudo code provided in the documentation, takes a single atom initially, and construct a goal list which recursion.

The program detects known fact and the length of the goals list as bottom recursive value.

The program uses dictionary to store proved value. If an atom is proved, and reoccurs in the goal list, program will exclude the proved goal and continue searching.

Program records the items in proving to break tie when infinite looping happens.

The program produces traceable output

### **Limitation:**

The program can successfully process  $p \leftarrow p^q$ , shown in “Asst2.self.txt” and “Asst2.self2.txt”. For Asst2.self.txt, there is no other way to prove q, hence q cannot be proved, then  $p \leftarrow p q$  cannot be proved. For Asst2.self2.txt, query p, the program can detect another way to prove p. For Asst2.self.txt, the program can detect there is no other way to prove p, hence p is false in this testcase.

Program can successfully process pair rules such as  $p \rightarrow q$  and  $q \leftarrow p$ , output as query p is false and query q is also false.

The program can handle of infinite looping logic by limiting the recursion looping count (in “Asst2.loop.txt”, query p,q,m). When some element is in proving more than a constraint, current recursion will be assumed wrong in order for previous recursion to break tie.

The draw back for this approach is the output is redundant, which is not a very concise way of solving the infinite looping problem.