**Resolution reputation for propositional logic**

Assume that all propositions are represented with a single lower case alphabet. The logic operators, ‘negation’, ‘and’, ‘or’, ‘implication’ and ‘if and if only’, are denoted by ‘~’, ‘\*’, ‘+’, ‘>’ and ‘=’. The parenthesis can be used in logic expressions. Assume that there is no grammatical error. Followings show some examples of logic expressions.

|  |  |
| --- | --- |
| (a  b)(a  b) | (a>b)+~(a\*b) |
| ((a  b)(a  b))b | ((a>b)=(~a+~b))+~b |

1. Implement a program that convers a logic expression into the clausal form. For the simplicity, let’s denote the negation of propositions by upper alphabets and drop ‘or’ in the clausal form. For example,

if the input is **~(a\*b)**, your program outputs **AB**, which is **~a+~b**.

For **~(a+~b)**, the output should be **A b**, which is **~a\*b** or **~a, b** (two clauses)

For **(a>b)+~(c\*~d)**, then your program outputs **AbCd**, which is **~a+b+~c+d**.

2. Implement the resolution reputation for predication logic. First, your program negates the goal and then converts all logic expressions into the clausal form. During the inference, each step should be displayed. The following is an example of execution.

|  |
| --- |
| **Input the goal:**  q\*s  **Input logic expressions:**  ~p>r  p>s  r>q  s>~t  t  . ( <- end of logic expressions)  **Clausal forms:**  Q S pr Ps Rq ST t  **Resolving:**  Q Rq => R  S Ps => P  pr Rq => pq  …  s S => EMPTY  **The goal is TRUE** |

The final output should be one of “**The goal is TRUE**” or “**Unable to prove**”. When resolving, use the breadth first strategy shown in the followings.

|  |
| --- |
| 1. Resolve every pair of clauses in the original S. Let all the resolvants be R 2. S <- S R 3. Repeat steps 1 and 2 until empty clause comes out |