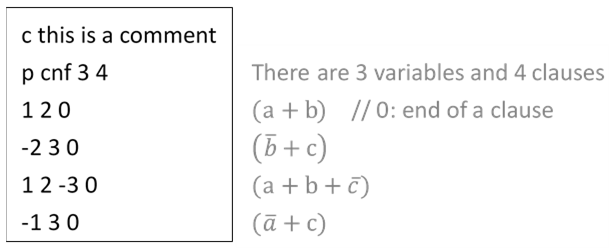
Practice/Real-Life Applications of Computational Algorithms

**Term Project: Yet Another SAT Solver (YaSat)**

1. **Goal**

In this project, you will need to implement your own SAT solver.

1. You can use the parser (parser.cpp and parser.h) we give, or write your own parser to read input which is written in CNF.



1. Then, write sat.cpp to find whether the input is satisfiable or not, and output the result in .sat file whose filename is the same as its input (.cnf).
2. If SAT, print “s SATISFIABLE” and a set of satisfying variable assignments.
3. Otherwise, print “s UNSATISFIABLE” in .sat file.
4. Notice: please use the Makefile we give to compile.
5. **Input / Output**

|  |  |
| --- | --- |
| Sample input 1 | Sample output 1 |
| p cnf 2 2  1 2 0  -1 -2 0 | s SATISFIABLE  v 1 -2 0 |
| Sample input 2 | Sample output 2 |
| p cnf 3 4  1 -2 0  1 3 0  2 -3 0  -1 0 | s UNSATISFIABLE |

1. **Command line**

./yasat [input.cnf]

1. **Hand in your project**

Please upload the following files in a zip, specifying your ID (e.g.

Student\_ID.zip) to E3 by the deadline.

(1) Source code

(2) A report that introduces your implementation

1. **Platform**

Linux

1. **Q&A**

For any question regarding this term project, please contact 黃甯琪 (blackitty321@gmail.com) and 林淯晨 (miz1205@gmail.com).