Homework 2
CSE4051 Program Verification, Fall 2025
Woosuk Lee
due: 10/27(Mon), 23:59

- You must use Python 3 for this assignment.
- Submit one python file per problem via the submission system in the course website. Make sure that your files run without errors.
- **You are NOT allowed to use any external libraries including Z3 for Python.**

Exercise 1 DIMACS format is a standard format for specifying SAT problems. A DIMACS file consists of a preamble line followed by lines representing clauses. The preamble line starts with the letter 'p', followed by the problem type (usually 'cnf' for conjunctive normal form), the number of variables, and the number of clauses.

Each clause is represented on a separate line, with positive integers representing variables and negative integers representing negated variables.

Each clause ends with a '0'. For example, the clause $x_1 \vee \neg x_2 \vee x_3$ would be represented as "1 -2 3 0".

Here is an example of a DIMACS file representing a simple SAT problem:

```
c this is a sample CNF file
p cnf 3 2
1 -2 0
-1 2 3 0
```

Any lines starting with 'c' are comments and should be ignored. This file specifies a SAT problem with 3 variables (namely, x_1, x_2, and x_3) and 2 clauses. The first clause is $x_1 \vee \neg x_2$ and the second clause is $\neg x_1 \vee x_2 \vee x_3$.

Write a python program that reads a DIMACS file whose name is given as a command-line argument and outputs "SAT" if the problem is satisfiable, and "UNSAT" otherwise. If the result is SAT, it must also display a satisfying assignment in the following format:

```
SAT
1 -2 3
```

indicating that x_1 is true, x_2 is false, and x_3 is true. If there are multiple satisfying assignments, you may output any one of them.

Submit a python file that implements a SAT solver that reads a DIMACS file and outputs the result as specified above. You may assume that the input DIMACS file is well-formed, and you may test your program on the test cases provided in the course website. The test cases include 50 SAT problems and 50 UNSAT problems where the number of variables ranges from 2 to 100 and the number of clauses ranges from 2 to 1000. If you want, you may test your program on additional test cases which can be found online (e.g., `http://www.cs.ubc.ca/~hoos/SATLIB/benchm.html`). You may use any algorithm of your choice (e.g., simple brute-force search, DPLL, CDCL), but any test case that cannot be solved within a certain time limit (e.g., 10 seconds, but this may be adjusted later) will be considered as incorrect.

Exercise 2 Write a report (in PDF) describing your implementation and preliminary evaluation on the test cases provided in the course website. The report must include the following:

- A brief description of your implementation (e.g., the algorithm you used, data structures you used, etc.)
- The time taken to solve each of the provided test cases
- The time taken to solve any additional test cases you used (if any)

The report may be written in any format of your choice, either in English or in Korean.