Implementation details:

Parsing:
The program starts by first parsing the file of clauses.
The clauses are stored in a list of hashmaps as shown .

```
[{'O1Y': '+'}, {'O2W': '+'}, {'O3Y': '+'}, {'L1W': '+'}, {'L2Y': '+'}, {'L3B': '+'}, {'C2B': '+', 'O2W': '-',
'C2W': '+'}, {'O3Y': '-', 'C3Y': '+', 'C3B': '+'}, {'C1B': '+', 'C1Y': '+', 'O1Y': '-'}, {'L1W': '-', 'C1W': '-'},
{'L2Y': '-', 'C2Y': '-'}, {'L3B': '-', 'C3B': '-'}, {'C1B': '+', 'C1Y': '+', 'C1W': '+'}, {'C2B': '+', 'C2Y': '+',
'C2W': '+'}, {'C3W': '+', 'C3Y': '+', 'C3B': '+'}, {'C1Y': '-', 'C2Y': '-'}, {'C1Y': '-', 'C3Y': '-'}, {'C1W': '-',
'C2W': '-'}, {'C3W': '-', 'C1W': '-'}, {'C2B': '-', 'C1B': '-'}, {'C1B': '-', 'C3B': '-'}, {'C2W': '-'}]
```

This set of clauses are provided as parameters for the resolver.
Every pair of combinations are derived in the resolver function and each pair is checked whether they can be resolved. If so their position in the clauses list are added to a priority queue based on the sum of the lenghts of both the clauses , ie heur = len(clauses[i]) + len(clauses[j])

The heap is popped if we have not yet derived the empty clause. The 2 clauses corresponding to the location popped from the heap is passed on to a reolve function. The resolve function makes a deepcopy of clause1 and it removes any resolvable symbols . Since a map is used there is no question of duplicate literals .

If the derived clause has no symbols then , the entailment is true and we can stop .
Else we check if derived clause is there in the list of clauses.If not we insert it into the hashmap of clauses and again form all possible combinations with the existing clauses and check whether they can be resolved and insert it into the priority queue based on sum of lengths of the two clauses.

The derived clause is checked for visited status in the list of clauses which is in O(N) , where N is the number of clauses.
The heuristic greatly improves the performance of the program as smaller clauses result in giving even smaller clauses and hence we are closer to the solution of empty clause.

The total number of iterations is 13 and max queue size is 45