Assignment : 2
[PART-B]

1- Write a program that takes in a Boolean formula and simplifies it using Boolean algebra rules such as De Morgan's laws and distributivity .For Example
Enter a propositional formula: ~(A ^ (B v C))
Simplified formula: (~A v (~B ^ ~C))

2- Write a program that takes a Boolean expression as input and outputs whether the expression is true or false, using propositional logic.

3-

3- Set: https://www.setgame.com/set is a card game that involves matching cards based on their attributes. The game requires players to use set theory to identify groups of cards that share the same attributes, such as color, shape, or number.
Create a program that can identify sets of cards based on their attributes, such as color, shape, or number. You can use loops and conditional statements to iterate through a set of cards, compare their attributes, and identify sets that meet specific criteria.

4- Implement a Prolog program that helps you keep track of your daily tasks and their completion status.

Requirements:

The program should be able to represent daily tasks as facts, with each task having a name and a completion status.

The program should be able to represent the completion status of a task using a predicate quantifier, such as "task_completed(X)" where X is the name of the task.

The program should be able to update the completion status of a task using Prolog rules, such as "complete_task(X)".

The program should be able to handle input and output in a user-friendly way, allowing the user to add new tasks, mark tasks as completed, and query the program for the completion status of tasks.

The program should be able to handle multiple tasks and should be able to update its knowledge base based on new information.

Hints:

Use the built-in Prolog predicates assert/1 and retract/1 to add and remove facts from the knowledge base.

Use Prolog rules to represent the completion status of a task, such as "task_completed(X) :- completed(X, true)."

Use the built-in Prolog predicate write/1 to display the completion status of tasks to the user.

Example Input/Output:

```
?- assert(task(work, false)).
true.

?- assert(task(study, false)).
true.

?- assert(task(exercise, false)).
true.

?- complete_task(study).
true.

?- task_completed(study).
true.

?- task_completed(work).
false.

?- complete_task(work).
true.

?- task_completed(work).
true.

?- task_completed(exercise).
false.
```