Project 1:
This project consists of three files:
1)sdctree.py
2)input.txt
3)project_1.py

"*sdctree.py*" is a library which consists of all the functions which are used for constructing and displaying the **OBDD**,**ROBDD** and **'ite'** expression. *"input.txt"* is used to give the input cover to the program. *"project_1.py"* is the top module in which the input cover is read from input.txt and the functions in sdctree.py are used as per requirement.
The file *"project_1.py"* is supposed to be run using *python3.0* and higher and requires "graphviz" package

Format in which input is to be given:
input.txt:
a'b
bc'
ac\
Each minterm in the cube is written on a new line.The last line must end with \ and marks the end of file i.e no character must follow this.

When project3.py is run,the user is asked the splitting order which needs to be entered as shown below. (It is expected that the number of variables in splitting order must match the number of variables used in "input.txt")
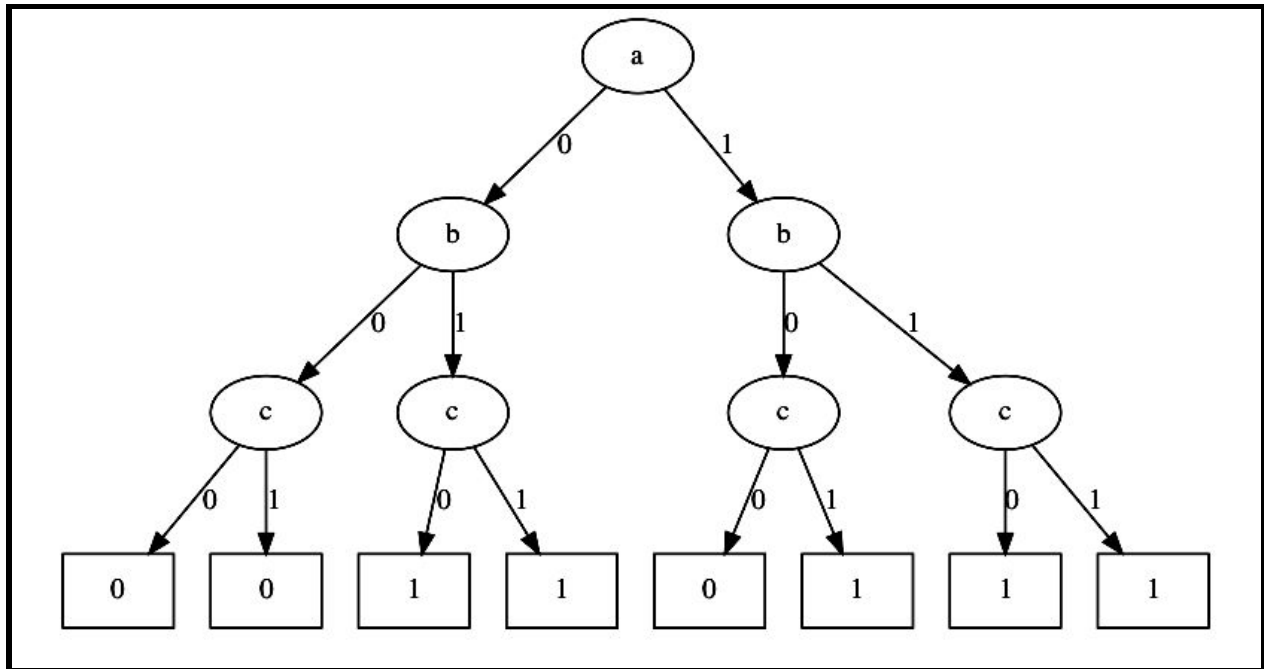
Sample output:
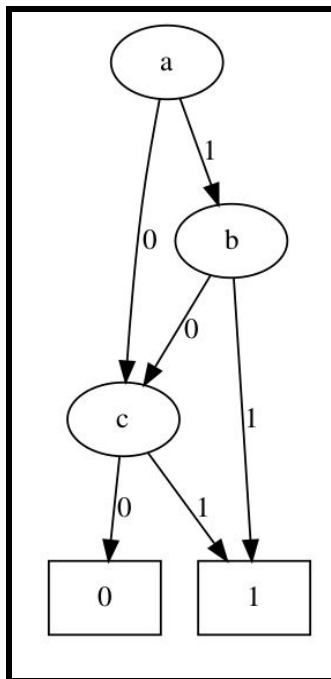
Enter the splitting order :
abc

ITE:

ite (a,ite (b,1,ite (c,1,0,))ite (b,1,0,))

Ordered Complete Binary Decision Diagram

Reduced Ordered Binary Decision Diagram

Project 2:
Project2 consists of two files:
1)table.txt
2)project_2.py

"project_2.py" is the top module which is used to find the minimum cover to the given expression. Input to the program is given in the form of a matrix in *"table.txt"*.

The rows of the matrix represent the cubes and the columns represent minterms.The element corresponding to a particular row and column combination will be **"1"** if the minterm is covered by that cube,else it will be **"0".**
Format in which input is to be given:

table.txt:
00001
00110
10100
11100\

The matrix is typed in the above fashion.The last line must end with \ and marks the end of file i.e no character must follow this.

Sample output:
The minimum cover is given by : [0, 0, 1, 0, 1]

It means that *3rd* and *5th* minterms are enough to represent the entire function.