Experiment VI

**Aim**: Write a program to convert NFA to DFA.

Algorithm

1. Start
2. Read the NFA transition table as input
3. To find the ε - closure of each state, traverse through all the states.
4. Add the current state into the ε - closure of the state.
5. For each other state, if there exists a link from the current state to a new state and the alphabet is ε, add the new state into the ε - closure of the current state.
6. Repeat step 5 recursively until the condition returns false.
7. Display the DFA transition table.
8. Stop

Output

States of NFA: A, B, C, D, E, F,

Given symbols for NFA: 0, 1, eps

NFA STATE TRANSITION TABLE

States: |0 |1 eps

---------+-------+-------+---

A |FC |- |BF

B |- |C |-

C |- |- |D

D |E |A |-

E |A |- |BF

F |- |- |-

e-Closure (A): ABF

e-Closure (B): B

e-Closure (C): CD

e-Closure (D): D

e-Closure (E): BEF

e-Closure (F): F

States of DFA: ABF, CDF, CD, BEF,

Given symbols for DFA: 0, 1,

DFA TRANSITION STATE TABLE

States: |0 |1

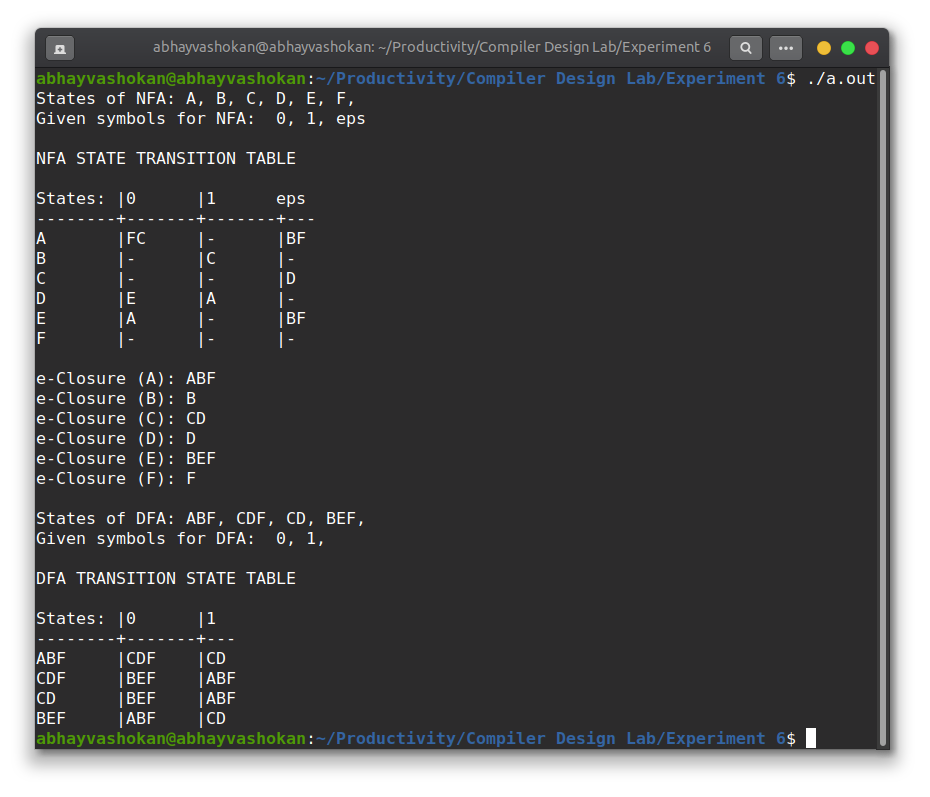
-----------+-------+---

ABF |CDF |CD

CDF |BEF |ABF

CD |BEF |ABF

BEF |ABF |CD

Screenshot

Readme

1. Compile and run the program using the command

**gcc 2Abhay-P6.l && ./a.out**

2. The DFA state transition table for the entered ε - NFA is displayed.

**Result**: Successfully implemented a program to convert an NFA to DFA.