10.Write a C program to find ε -closure for all the states in a Non-Deterministic Finite Automata (NFA) with ε -moves.

**AIM:** To Write a C program to find ε -closure for all the states in a Non-Deterministic Finite

Automata (NFA) with ε -moves.

**ALGORITHM:**

Algorithm: Calculate ε-Closure for All States in an NFA

Input:

- n\_states: number of states in the NFA

- n\_symbols: number of input symbols (excluding ε)

- transitions: transition table with dimensions [n\_states][n\_symbols+1][n\_states], where transitions[i][j][k] represents the set of states reachable from state i with symbol j (or ε if j equals n\_symbols) leading to state k.

Output:

- epsilon\_closure: an array of sets representing ε-closure for each state

Function epsilonClosure(state):

// Function to calculate ε-closure for a given state

visited[state] = true

epsilon\_closure[state].add(state) // Include the current state in its own ε-closure

for each transition in transitions[state][n\_symbols]:

// Iterate over ε-transitions

if not visited[transition]:

epsilonClosure(transition) // Recursive call for ε-transition

Function calculateEpsilonClosure():

// Function to calculate ε-closure for all states

for each state in range(n\_states):

visited = [false] \* n\_states

epsilon\_closure[state] = empty set

epsilonClosure(state)

// Example Usage:

initialize epsilon\_closure array

calculateEpsilonClosure()

// Print ε-closure for each state

for each state in range(n\_states):

print("ε-closure(", state, "):", epsilon\_closure[state])

**PROGRAM:**

#include <stdio.h>

#include <stdbool.h>

#define MAX\_STATES 10

#define MAX\_SYMBOLS 10

int n\_states;

int n\_symbols;

int transitions[MAX\_STATES][MAX\_SYMBOLS][MAX\_STATES];

void epsilonClosure(int state, bool visited[]) {

visited[state] = true;

for (int i = 0; i < n\_states; ++i) {

if (transitions[state][n\_symbols][i] && !visited[i]) {

epsilonClosure(i, visited);

}

}

}

void printEpsilonClosure(int state) {

bool visited[MAX\_STATES] = {false};

epsilonClosure(state, visited);

printf("e-closure(%d): {", state);

for (int i = 0; i < n\_states; ++i) {

if (visited[i]) {

printf("%d ", i);

}

}

printf("}\n");

}

int main() {

printf("Enter the number of states: ");

scanf("%d", &n\_states);

printf("Enter the number of input symbols (excluding e): ");

scanf("%d", &n\_symbols);

printf("Enter transitions for each state and input symbol (use -1 for no transition):\n");

for (int i = 0; i < n\_states; ++i) {

for (int j = 0; j < n\_symbols; ++j) {

printf("Transition for state %d and symbol %d: ", i, j);

scanf("%d", &transitions[i][j][0]);

}

printf("e-transition for state %d: ", i);

scanf("%d", &transitions[i][n\_symbols][0]);

}

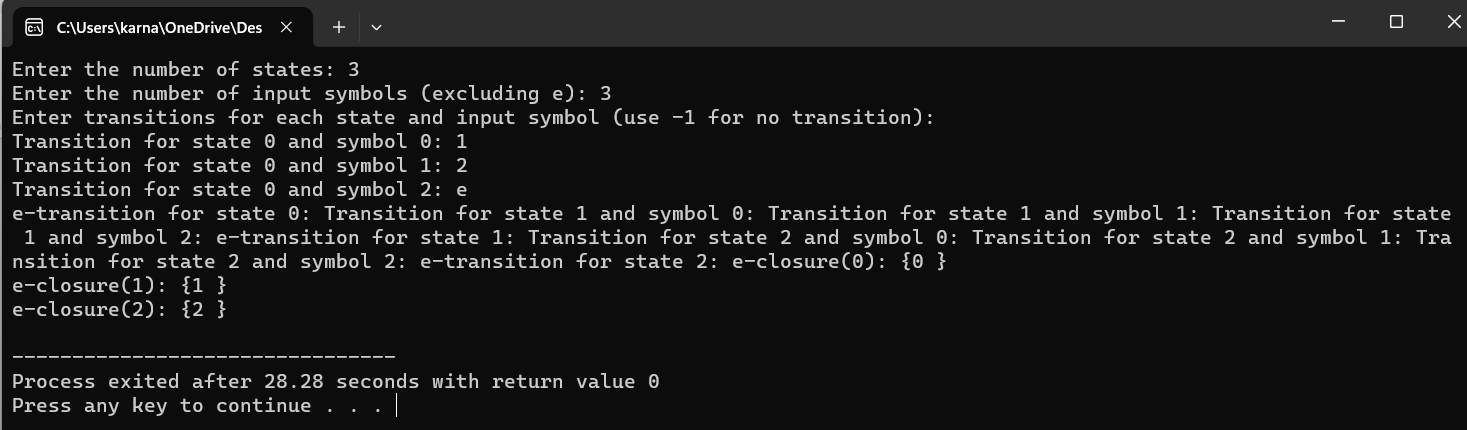
for (int i = 0; i < n\_states; ++i) {

printEpsilonClosure(i);

}

return 0;

}

**OUTPUT:** ****