11.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: FindEpsilonClosure

Input:

- nfa: Non-Deterministic Finite Automaton with ε-moves represented as a transition matrix

- numStates: Number of states in the NFA

Output:

- epsilonClosure: Array of sets representing ε-closure for each state

Procedure:

1. Initialize an empty array epsilonClosure of sets, one for each state.

2. For each state q in the NFA:

a. Initialize an empty stack.

b. Create a set closure containing only the current state q.

c. Push q onto the stack.

d. While the stack is not empty:

i. Pop a state s from the stack.

ii. For each state t reachable from s via ε-moves:

- If t is not already in the closure set:

\* Add t to the closure set.

\* Push t onto the stack.

e. Store the closure set in the epsilonClosure array for state q.

3. Return the epsilonClosure array.

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

#define MAX\_STATES 10

#define MAX\_ALPHABET 10

int epsilonClosure[MAX\_STATES][MAX\_STATES];

void addStateToClosure(int state, int closure[], int n) {

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

if (closure[i] == state) {

return;

}

}

closure[n] = state;

}

void epsilonClosureUtil(int nfa[MAX\_STATES][MAX\_ALPHABET], int state, int closure[], int \*closureSize) {

addStateToClosure(state, closure, \*closureSize);

for (int i = 0; i < nfa[state][0]; i++) {

int nextState = nfa[state][i + 1];

epsilonClosureUtil(nfa, nextState, closure, closureSize);

}

}

void findEpsilonClosure(int nfa[MAX\_STATES][MAX\_ALPHABET], int numStates) {

for (int i = 0; i < numStates; i++) {

int closure[MAX\_STATES];

int closureSize = 0;

epsilonClosureUtil(nfa, i, closure, &closureSize);

printf("e-closure(q%d) = {", i + 1);

for (int j = 0; j < closureSize; j++) {

printf("q%d", closure[j] + 1);

if (j < closureSize - 1) {

printf(", ");

}

}

printf("}\n");

}

}

int main() {

int numStates, numAlphabet;

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

scanf("%d", &numStates);

printf("Enter the number of symbols in the alphabet (excluding epsilon): ");

scanf("%d", &numAlphabet);

int nfa[MAX\_STATES][MAX\_ALPHABET];

for (int i = 0; i < numStates; i++) {

printf("Enter the number of transitions for state q%d (including epsilon): ", i + 1);

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

printf("Enter transitions for state q%d (use 0 for epsilon): ", i + 1);

for (int j = 1; j <= nfa[i][0]; j++) {

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

}

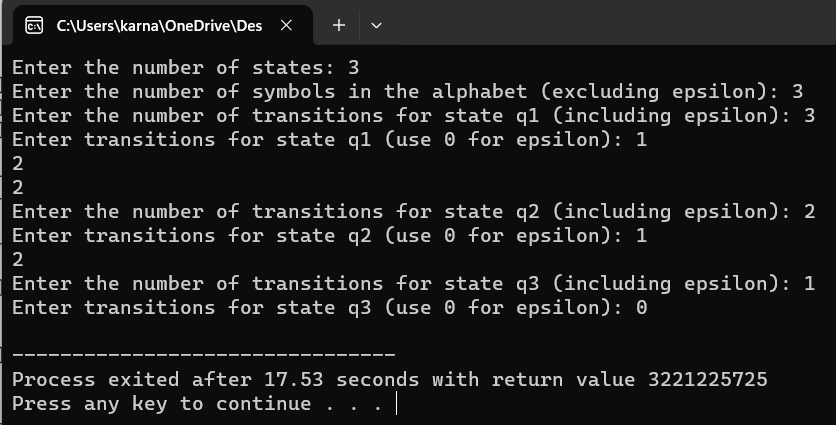
}

findEpsilonClosure(nfa, numStates);

return 0;

}

**OUTPUT:**

****