1. **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

**CODE :**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <stdbool.h>**

**#define MAX\_STATES 10**

**// Structure to represent a transition in the NFA**

**typedef struct Transition {**

**char symbol;**

**int nextState;**

**} Transition;**

**// Structure to represent a state in the NFA**

**typedef struct State {**

**int stateNumber;**

**int numTransitions;**

**Transition \*transitions;**

**bool epsilonClosure[MAX\_STATES];**

**} State;**

**// Structure to represent an NFA**

**typedef struct NFA {**

**int numStates;**

**State \*states;**

**int initialState;**

**} NFA;**

**// Calculate the epsilon closure for a given state**

**void calculateEpsilonClosure(NFA \*nfa, int state) {**

**int i, j;**

**State \*currentState = &nfa->states[state];**

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

**// Set the epsilon closure of the current state to true**

**currentState->epsilonClosure[state] = true;**

**// Iterate over the epsilon transitions from the current state**

**for (i = 0; i < currentState->numTransitions; i++) {**

**if (currentState->transitions[i].symbol == '\0') {**

**// Epsilon transition found, recursively calculate the epsilon closure for the next state**

**int nextState = currentState->transitions[i].nextState;**

**if (!visited[nextState]) {**

**calculateEpsilonClosure(nfa, nextState);**

**visited[nextState] = true;**

**}**

**// Merge the epsilon closure of the next state with the current state's epsilon closure**

**for (j = 0; j < nfa->numStates; j++) {**

**currentState->epsilonClosure[j] = currentState->epsilonClosure[j] || nfa->states[nextState].epsilonClosure[j];**

**}**

**}**

**}**

**}**

**int main() {**

**// Define the NFA**

**NFA nfa = {**

**.numStates = 3,**

**.states = (State[]) {**

**{**

**.stateNumber = 0,**

**.numTransitions = 1,**

**.transitions = (Transition[]) {**

**{ '\0', 1 }**

**}**

**},**

**{**

**.stateNumber = 1,**

**.numTransitions = 2,**

**.transitions = (Transition[]) {**

**{ '\0', 1 },**

**{ 'a', 2 }**

**}**

**},**

**{**

**.stateNumber = 2,**

**.numTransitions = 1,**

**.transitions = (Transition[]) {**

**{ '\0', 0 }**

**}**

**}**

**},**

**.initialState = 0**

**};**

**int i, j;**

**// Calculate the epsilon closure for each state in the NFA**

**for (i = 0; i < nfa.numStates; i++) {**

**calculateEpsilonClosure(&nfa, i);**

**}**

**// Print the epsilon closure for each state in the NFA**

**printf("Epsilon closure for each state:\n");**

**for (i = 0; i < nfa.numStates; i++) {**

**printf("State %d: ", i);**

**for (j = 0; j < nfa.numStates; j++) {**

**if (nfa.states[i].epsilonClosure[j]) {**

**printf("%d ", j);**

**}**

**}**

**printf("\n");**

**}**

**return 0;**

**}**