Practical 3

Title: Write a C program to implement finite automata and string validation.

Hint : The purpose of this code is to implement finite automata and string validation**.**

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int searchAlphabets(char alphabets[], char toBeFound)
{
   int i;
   for (i = 0; i < strlen(alphabets); i++)
   {
      if (alphabets[i] == toBeFound)
         {
         return i;
      }
   }
}

void main()
{
   char alphabets[128], input[100];
   int states, intinalState, finalStates[100], currentState, i, j, transitionTable[100][128];
   int temp = 0, term = 0;
   printf("Jay Dalsaniya \n");
   printf("92100103336 \n");
   printf("Enter alphabets in DFA as a string : ");
   scanf("%s", &alphabets);
   printf("Enter number of states : ");
   scanf("%d", &states);
   printf("Enter Intial state : ");
   while (0 == 0)
   {
         scanf("%d", &intinalState);
         if (intinalState < states)
         {          break;          }
         printf("Please enter a valid IntialState : ");
   }
   printf("Enter Final States one by one. When completed enter -1\n");
```

```
i = 0;

while (0 == 0)
{
   scanf("%d", &temp);
   if (temp == -1 && term > 0)
   {
      break;
   }
   else if (temp == -1 && term == 0) {
      printf("Please enter atleast one FinalState\n");
   }
   else if (temp >= states){
      printf("Please enter a valid FinalState\n");
   }
   Else{
      finalStates[i] = temp;
      i++;      term++;
   }  }
printf("Entering Transition table\n");
for (i = 0; i < states; ++i) {
   for (j = 0; j < strlen(alphabets); ++j) {
      printf("Enter state when on state %d and input alphabets is %c: ", i, alphabets[j]);
      scanf("%d", &transitionTable[i][j]);
   }
}
while (0 == 0){
   printf("\nEnter String to match(Enter -1 to stop matching) : ");
   scanf("%s", &input);
   if (strcmp(input, "-1") == 0)
   {
      break;
   }
   currentState = intinalState;
   for (i = 0; i < strlen(input); i++) {
         currentState = transitionTable[currentState][searchAlphabets(alphabets, input[i])];
   }
   j = 1;
   for (i = 0; i < states; i++)
   {
      if (currentState == finalStates[i])
      {
         j = 0;
      }
   }
```

```
        if (j == 0){
           printf("Given String belongs to given DFA\n");
        }
        else {
              printf("Given String does not belong to given DFA\n");
        }
    }
}
```

Output:

```
Jay Dalsaniya
92100103336
Enter alphabets in DFA as a string : ab
Enter number of states : 4
Enter Intial state : 0
Enter Final States one by one. When completed enter -1
2-1
Entering Transition table
Enter state when on state 0 and input alphabets is a: 1
Enter state when on state 0 and input alphabets is b: 3
Enter state when on state 1 and input alphabets is a: 3
Enter state when on state 1 and input alphabets is b: 2
Enter state when on state 2 and input alphabets is a: 2
Enter state when on state 2 and input alphabets is b: 2
Enter state when on state 3 and input alphabets is a: 3
Enter state when on state 3 and input alphabets is b: 3

Enter String to match(Enter -1 to stop matching) : abab
Given String belongs to given DFA

Enter String to match(Enter -1 to stop matching) : baba
Given String does not belong to given DFA

Enter String to match(Enter -1 to stop matching) : abba
Given String belongs to given DFA

Enter String to match(Enter -1 to stop matching) : aabab
Given String does not belong to given DFA

Enter String to match(Enter -1 to stop matching) : babab
Given String does not belong to given DFA
```