Lab - 03

ACT : Advanced Compiler Techniques

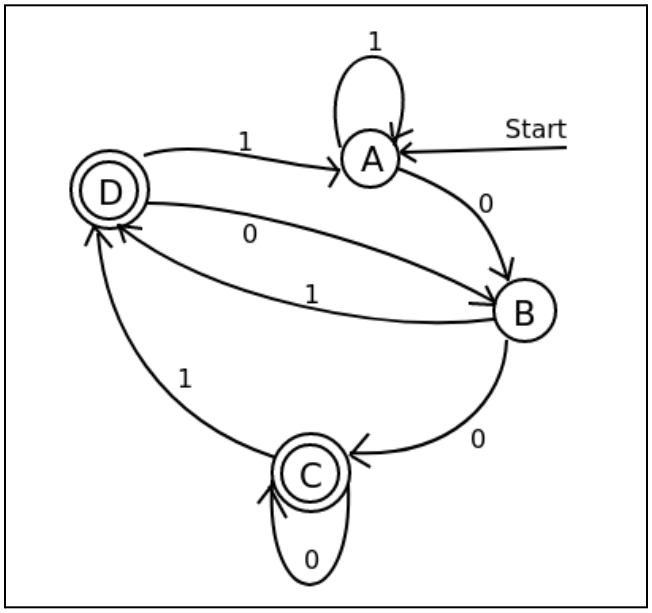
AIM : Using Finite Automata, check whether a given string is valid or invalid.

Name : Ambalia Harshit
Roll no : –
Date : 04 Sept 2023

Question 01 : check whether a given string is valid or invalid.

Input: State- Transition Table
Start State
Set of Accepting States
input String

Output: String is Valid/Invalid



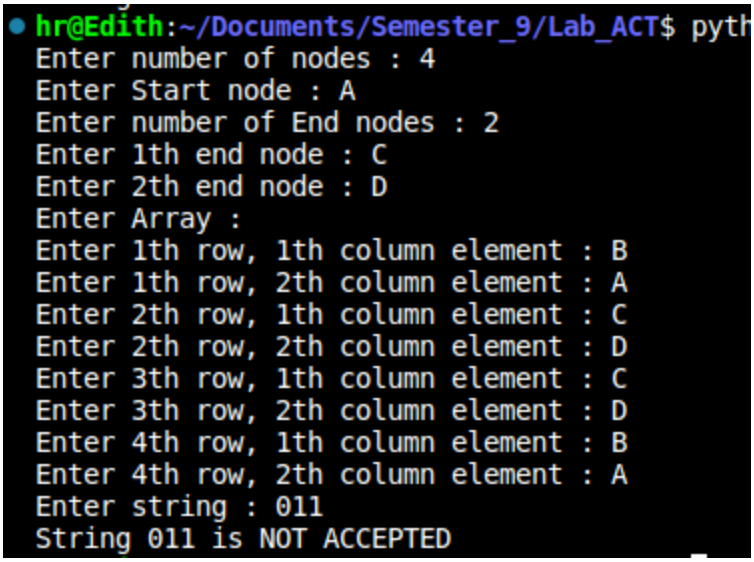
For the given automata we are validating the input/output.

```
## LAB - 03

# NAME : AMBALIA HARSHIT
# SUBJECT : ACT
# ROLL NO. : -----
# AIM : VALIDATE FINITE AUTOMETA FOR GIVEN INPUT

def create_mapping(nodes):
   names = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L',
'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
   mapping = {}
   for i in range(nodes):
       mapping.update({names[i]:i})
   return mapping

def check_autometa(strg, start, arr, end_nodes, nodes):
   mapping = create_mapping(nodes)
   current = mapping[start]
   for i in strg:
       i = int(i)
       current = mapping[arr[current][i]]

   flag = False
   for i in end_nodes:
       if(mapping[i]==current):
           flag = True
           break

   return flag


if __name__=="__main__":
   nodes = int(input("Enter number of nodes : "))

   start = input("Enter Start node : ")

   end_node_count = int(input("Enter number of End nodes : "))
   end_nodes = []
   for i in range(end_node_count):
       end_nodes.append(input(f'Enter {i+1}th end node : '))

   print("Enter Array : ")
   arr = []
   for i in range(nodes):
       a = []
```

```
        for j in range(2):
            a.append(input(f'Enter {i+1}th row, {j+1}th column element :
'))
        arr.append(a)

    strg = input("Enter string : ")

    flag = check_autometa(strg, start, arr, end_nodes, nodes)

    if flag:
        print(f'String {strg} is ACCEPTED')
    else:
        print(f'String {strg} is NOT ACCEPTED')
```

Outputs :

```
hr@Edith:~/Documents/Semester_9/Lab_ACT$ pyth
Enter number of nodes : 4
Enter Start node : A
Enter number of End nodes : 2
Enter 1th end node : C
Enter 2th end node : D
Enter Array :
Enter 1th row, 1th column element : B
Enter 1th row, 2th column element : A
Enter 2th row, 1th column element : C
Enter 2th row, 2th column element : D
Enter 3th row, 1th column element : C
Enter 3th row, 2th column element : D
Enter 4th row, 1th column element : B
Enter 4th row, 2th column element : A
Enter string : 011000
String 011000 is ACCEPTED
```



Restrictions :

- Starting node should be a single valid node.
- There should be exactly two edges for each node, using 0 and 1.
- There can be more than one ending node.
- String can only contain 0 and 1 and for every node there should be an edge using 0 and 1 only.