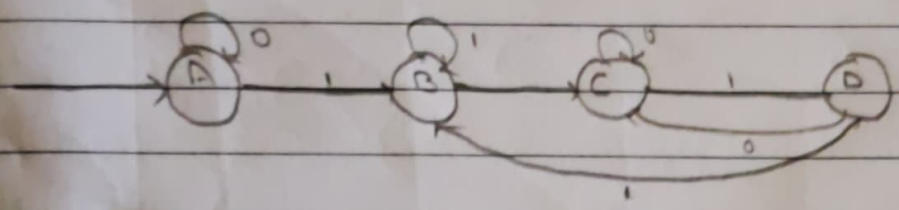
Sr. No.	Topic	Date	Sign
1.	Design a program for creating a machine that accepts string ending with 101	02/02/23	
2.	Design a program for accepts decimal number by 2.	02/02/23	
3.	Write a program for tokenization of input.	04/02/23	
4.	Design a program for accepting three consecutive one.	04/02/23	
5.	Design a program for turing machine accepts even number of is	2/3/23	
6.	Design program a creating machine which count no. 1's and 0's string	2/3/23	
7.	Design program for creating machine which accepts string have equal number of 1's and 0's.	2/3/23	

Practical 1

Aim:- Design a Program for Creating machine that accepts the string always ending with 101.

DFA:- $\Sigma = \{0, 1\}$ $Q = \{A, B, C, D\}$, $q_0 = A$ $F = D$



Transition Table:-

	0	1
A	A	B
B	C	B
C	C	D
D	C	B

Code:-
```
dfa = {
  "A": {
   "0" : "A",
   "1" : "B",
  }
  "B" : {
   "0" : "C",
   "1" : "B"
  }
  "C" : {
   "0" : "C",
   "1" : "D"
  }
```

Output:

Enter string: 0100

not accepted

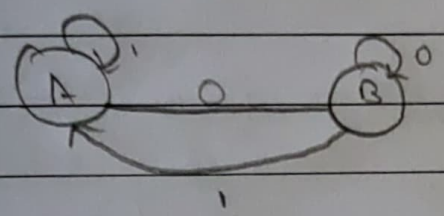
```
        "D" : {
        "0" : "C"
        "1" : "B"
        }
    }
initial state = "A"
final state = "D"
def checkString(data):
    currentstate = initial state
    for s in data:
        Current state = dfa [current state] [s]
    if current state = FinalState:
        return True
    return False
data = input ("Enter a string:")
x = checkString(data)
print (" Accepted" if x else " not accepted")
```

Practical 2

Aim:- Design a program for accepting decimal number divisible by 2.

DFA:- $\Sigma = \{0, 1\}$ $q_0 = A$ $F = B$ $Q = \{A, B\}$

The



Transition table

	0	1
A	B	A
B	B	A

```
dfa = {
    "A" : {
    "0" : "B",
    "1" : "A"
    }
    "B" : {
    "0" : "B",
    "1" : "A"
    }
}
```

Output:

Enter a string : 0101

Rejected

```
initial state : "A"
final state : "B"
def checkstring (data):
Current state : initial state
for s in data
    Current state = dfa[current state] [s]
if current state = final state:
        return True
return False
data = int("enter a string:")
x : checkstring (data)
print ("accepted" if x else "not accepted.
```

Practical no. 3

Aim:- Write a program for tokenization of given input.

Code:-

```
UserData = input("enter a string")
print (f' string entered : {userData}')
print (f" Tokenization : {userData.split()}")
```

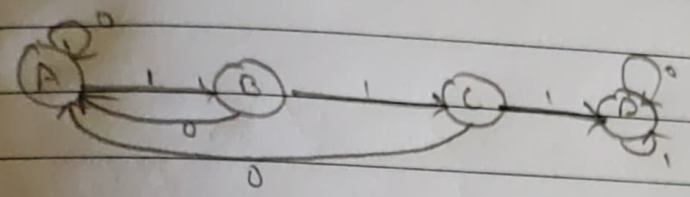
Output:-

```
Enter a string : This is a TOC practical.
String entered : This is a TOC practical
Tokenization : ["This", "is", "a", "TOC", "Practical."]
```

Practical no. 4

Aim:- Design a program for creating machine that accepts three consecutive ones.

DFA:- $\Sigma = \{0, 1\}$ $q_0 = A$ $F = D$ $Q = \{A, B, C, D\}$



Transition Table

	0	1
A	A	B
B	A	C
C	A	D
D	D	D

Code:-
```
dfa = {
    "A" {
    "0" : "A",
    "1" : "B"
    },
    "B" {
    "0" : "A",
    "1" : "C"
    },
    "C" {
    "0" : "A",
    "1" : "D",
    }
```

Output:

Enter a string: 0110101

Rejected.

```
            "D" {
            "0" : "D",
            "1" : "D"
            }
        }
initial state : "A"
Final state : "B"
def chekstring (data):
    Currentstate : initial state
    for s in data:
        current state : dta [current state] (s)
    if Current state : finalstate
        return True
    return false
data = input ("Enter a string")
x: chek string (data)
print ("accepted" if x else "rejected")
```

Practical No: 5

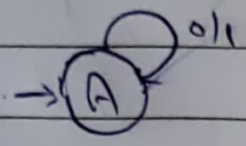
Aim:- Design a program for turing machine accepts even number of 1's.

```
states : {
    'A': {
        '0' : ('A', '0', 'R'),
        '1' : ('B', '0', 'R'),
        '_' : ('D', '_', 'L')
    },
    'B' : {
        '0' : ('B', '0', 'R'),
        '1' : ('A', '0', 'R'),
        '_' : ('C', '_', 'L')
    },
    'C': {
        '0' : ('C', '0', 'L'),
        '1' : ('C', '1', 'L'),
        '_' : ('A', '_', 'R')
    },
    'D': {
        '0' : ('D', '0', 'L'),
        '1' : ('D', '1', 'L'),
        '_' : ('E', '_', 'R'),
    },
    'E' : {
        '0' : ('D', '0', 'R'),
        '1' : ('D', '1', 'R'),
        '_' : ('F', '_', 'L')
    },
```

```
        'F': {
            '0': ('F', '0', 'L'),
            '1': ('F', '1', 'L'),
            '_': ('A', '_', 'R')
        }
    }
initial_state = "A"
final_state = {"A"}
def turing_machine (input_str):
    current_state = initial_state
    tape = list (input_str)
    i_head = 0
    while True:
        if tape[i_head] not in states[current_state]:
            return false
        new_state, write_value, move_dir = states[current_state]
        tape[i_head] = write_value
        if move_dir == 'R':
            i_head += 1
        elif move_dir == 'L':
            i_head -= 1
        current_state = new_state
print (turing_machine ('011'))
print (turing_machine ('01111'))
print (turing_machine ('01'))
```

Practical no. 6

Aim: Design program for creating a machine which count no. of 1's and 0's in given string.



	0	1
A	A	A

```
state : {
    "A" : {
        "0": "A",
        "1": "A"
    }
}
initial_state = "A"
def count_string(string: str):
    current_state = initial_state
    count_0 = 0
    count_1 = 0
    for s in string:
        if s == "0"
            count_0 += 1
        if s == "1"
            count_1 += 1
        current_state = states[current_state][s]
    return f" Number of 1's {count_1} Number of 0's is
    {count_0}"
```

```
if __name__ == "__main__":
    user_input = input("Enter string:")
    x = count_string(user_input)
    print(x)
```

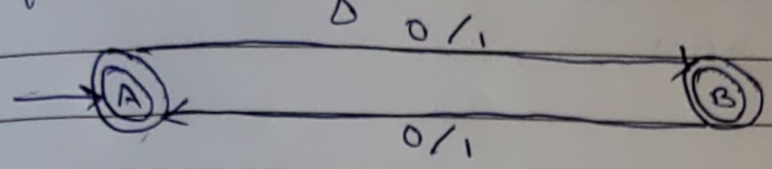
Output :-

Enter String : 0101

The number of 1's is 2 and number of 0's is 2

Practical no. 7

Aim:- Design a program for Creating a machine which accepts string have equal number of 1's and 0's.



Transistion table

	0	1
A	B	B
B	A	A

```
states = {
    "A" : {
        "0" : "B"
        "1" : "B"
    },
    "B" : {
        "0" : "A",
        "1" : "A"
    }
}
initial_state = "A"
final_state = {"A"}
```

Output

Enter string :- 0011
Accepted.

```
def check_string (string: str):
    current_state = initial_state
    count_0 = 0
    count_1 = 0
    for s in string
        if s == "0"
            count_0 += 1
        if s == "1":
            count_1 += 1
        current_state = state [current_state] [s]
    if current_state in final_state and count_0 == count_1:
        return True
    return false

if __name__ == "__main__":
    user_input = input ("Enter string:")
    x = check_string (user_input)
    print ("Accepted" if x else "Not accepted")
```