Prac 1

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

}

initalState="A"
finalSate="D"

def checkString(data):
    currentState=initalState
    for s in data:
        currentState=dfa[currentState][s]
    if currentState==finalSate:
        return True
    return False

data = input("Enter a string: ")
x= checkString(data)
print("Accepted" if x else "Not accepted")
```

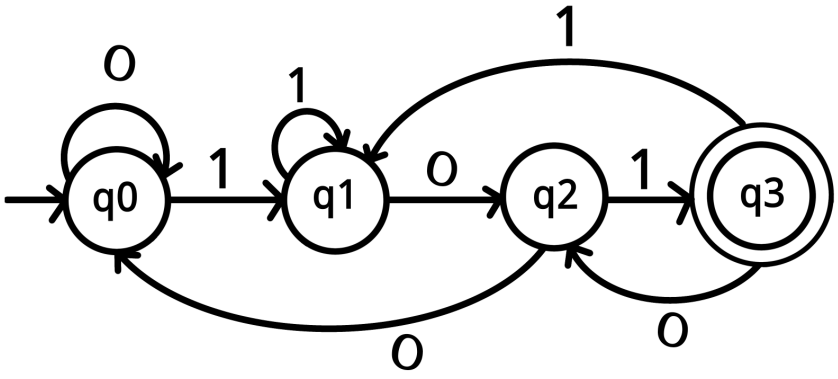
Prac1

```
# Design a Program for creating machine that accepts the string always ending with 101.
```

```
# pip install automata-lib
from automata.fa.dfa import DFA

dfa = DFA(
    states={"q0", "q1", "q2", "q3"},
    input_symbols={"0", "1"},
    transitions={
        "q0": {"0": "q0", "1": "q1"},
        "q1": {"0": "q2", "1": "q1"},
        "q2": {"0": "q0", "1": "q3"},
        "q3": {"0": "q2", "1": "q1"},
    },
    initial_state="q0",
    final_states={"q3"},
)
n = input("Enter string:")
if dfa.accepts_input(n):
    print("Accepted")
else:
    print("Rejected")
```

Q. DFA Ending with 101



github.com/bruhmaand

Prac 2

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

}

initalState="A"
finalSate="B"

def checkString(data):
   currentState=initalState
   for s in data:
      currentState=dfa[currentState][s]
   if currentState==finalSate:
      return True
   return False

data = input("Enter a string: ")
x= checkString(data)
print("Accepted" if x else "Not accepted")
```

Prac 2

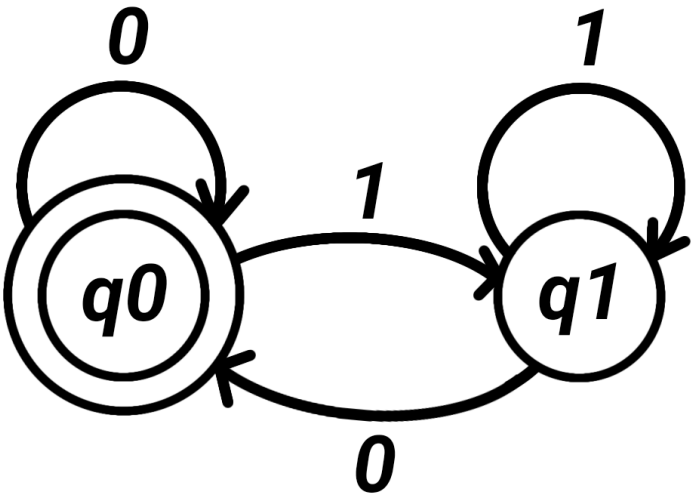
```
# Design a program for accepting Binary string divisible by 2.

# pip install automata-lib
from automata.fa.dfa import DFA

dfa = DFA(
   states={"q0", "q1"},
   input_symbols={"0", "1"},
   transitions={
     "q0": {"0": "q0", "1": "q1"},
```

```
      "q1": {"0": "q0", "1": "q1"}
    },
    initial_state="q0",
    final_states={"q0"},
)
n = input("Enter string:")
if dfa.accepts_input(n):
    print("Accepted")
else:
    print("Rejected")
```

Q. Binary strings divisible by 2



github.com/bruhmaand

Prac 3

```
userData=input("Enter a string : ")
print(f'String Entered: {userData}')
print(f"Tokenization: {userData.split()}")
```

Prac 3

```
# Write a program for tokenization of given input.
```

```
def tokenize(input_str):

    tokens = []
    word = ""

    for char in input_str:
        if char.isalnum() or char.isalpha():
            word += char

        else:
            if word:
                tokens.append(word)
                word = ""

    return tokens



if __name__ == "__main__":

    string = "This is an example of tokenization."
    print(tokenize(string))
```

Prac 4

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

```
    "1":"D",
    },

}

initalState="A"
finalSate="D"

def checkString(data):
    currentState=initalState
    for s in data:
        currentState=dfa[currentState][s]
    if currentState==finalSate:
        return True
    return False

data = input("Enter a string: ")
x= checkString(data)
print("Accepted" if x else "Not accepted")
```

Prac 4

```
# Design a Program for creating machine that accepts three consecutive one.

#only consecutive 111

# pip install automata-lib
from automata.fa.dfa import DFA

dfa = DFA(
    states={"q0", "q1", "q2", "q3", "q4", "q5", "q6", "q7"},
    input_symbols={"0", "1"},
    transitions={
        "q0": {"0": "q0", "1": "q1"},
        "q1": {"0": "q0", "1": "q2"},
        "q2": {"0": "q0", "1": "q3"},
        "q3": {"0": "q5", "1": "q4"},
        "q4": {"0": "q4", "1": "q4"},
        "q5": {"0": "q5", "1": "q6"},
        "q6": {"0": "q5", "1": "q7"},
        "q7": {"0": "q5", "1": "q3"},
    },
    initial_state="q0",
```

```
    final_states={"q3", "q5", "q6", "q7"},
)
n = input("Enter string:")
if dfa.accepts_input(n):
    print("Accepted")
else:
    print("Rejected")
```

Prac 5

```
# Design a program for Turing machine that’s accepts the even number of 1's.

states = {
    'A': {
        '0': ('A', '0', 'R'),
        '1': ('B', '0', 'R'),
        '_': ('C', '_', 'L')
    },
    'B': {
        '0': ('B', '0', 'R'),
        '1': ('A', '0', 'R'),
        '_': ('C', '_', 'L')
    },
    "C":{}
}

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]]
        tape[i_head] = write_value

        if move_dir == 'R':
            i_head += 1
        elif move_dir == 'L':
```

```
            i_head -= 1

        current_state = new_state

        if current_state in final_state and i_head >= len(tape):
            return True
        elif current_state not in states or i_head >= len(tape) or i_head < 0:
            return False



print(turing_machine(input("Enter a Stirng: ")))
```

Prac 6

Design a program for creating a machine which accepts string having equal no. of 1’s and 0’s.

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

initial_state = "A"
final_state = {"A"}


def check_string(string:str):
    curr_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
        curr_state = states[curr_state][s]
```

```
    if(curr_state in final_state and count_0 == count_1):
        return True
    return False




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

Prac 7

```
# Design a program for creating a machine which count number of 1's and 0's in a given string.

states = {
    "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"The Number of 1's is {count_1} and number of 0's is {count_0}"

if __name__ == "__main__":
    X = count_string(input("Enter the string : "))
    print(X)
```