Name: Gudur Krishna Chaitanya Roll no: 102117049 email used:
gchaitanya_be21@thapar.edu

Unsupported Cell Type. Double-Click to inspect/edit the content.

```
from qiskit_ibm_provider import IBMProvider

provider = IBMProvider(token='f55702335547d565b44eb80fd6708f3b82b4d0147236062

active_account = provider.active_account()

print("Active Account Details:")

print(active_account)
```

```
Active Account Details:
{'channel': 'ibm_quantum', 'token': 'f55702335547d565b44eb80fd6708f3b82b4d0
```

```
import numpy as np
from qiskit import QuantumCircuit, transpile
import qiskit_aer
from qiskit.visualization import plot_histogram
```

```
n = 3
```

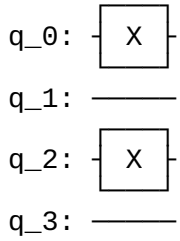
```
n = 3
const_oracle = QuantumCircuit(n+1)
output = np.random.randint(2)
if output == 1:
    const_oracle.x(n)
const_oracle.draw(output='text',style='bw')
```

```
q_0: ─────
q_1: ─────
q_2: ─────
q_3: ┤ X ├
```

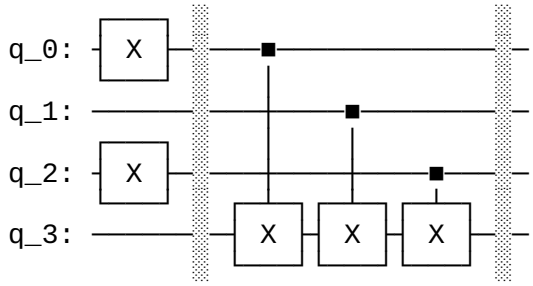
```
balanced_oracle = QuantumCircuit(n+1)
b_str = "101"
```

```
balanced_oracle = QuantumCircuit(n+1)
b_str = "101"
for qubit in range(len(b_str)):
    if b_str[qubit] == '1':
```

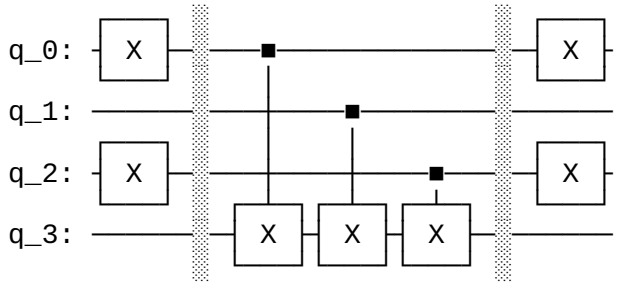
```
        balanced_oracle.x(qubit)
balanced_oracle.draw(output='text',style='bw')
```




```
balanced_oracle = QuantumCircuit(n+1)
b_str = "101"
for qubit in range(len(b_str)):
    if b_str[qubit] == '1':
        balanced_oracle.x(qubit)
balanced_oracle.barrier()
for qubit in range(n):
    balanced_oracle.cx(qubit, n)
balanced_oracle.barrier()
balanced_oracle.draw(output='text',style='bw')
```

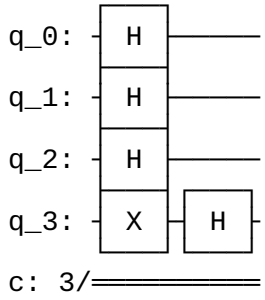



```
balanced_oracle = QuantumCircuit(n+1)
b_str = "101"
for qubit in range(len(b_str)):
    if b_str[qubit] == '1':
        balanced_oracle.x(qubit)
balanced_oracle.barrier()
for qubit in range(n):
    balanced_oracle.cx(qubit, n)
balanced_oracle.barrier()
for qubit in range(len(b_str)):
    if b_str[qubit] == '1':
        balanced_oracle.x(qubit)
balanced_oracle.draw(output='text',style='bw')
```

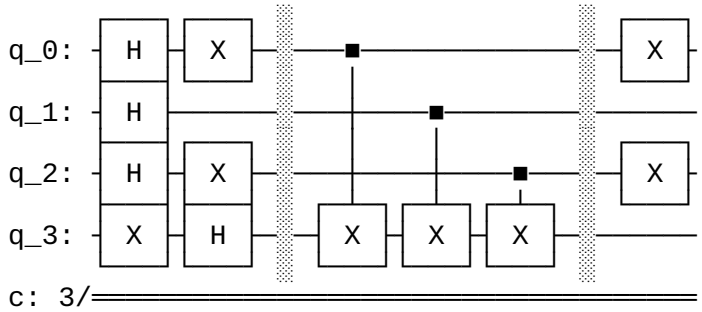



```
dj_circuit = QuantumCircuit(n+1, n)
```

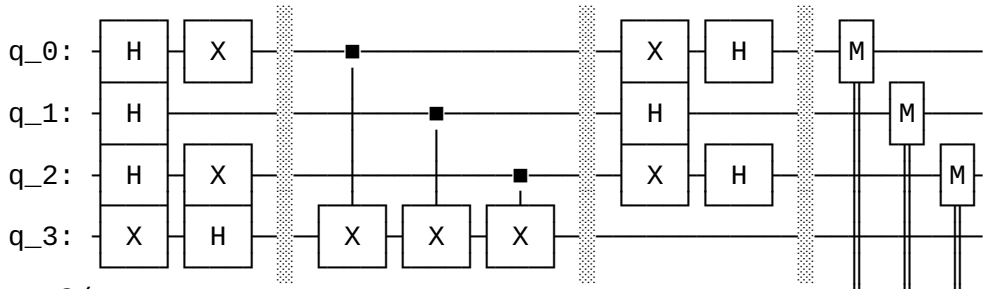
```
for qubit in range(n):
    dj_circuit.h(qubit)
dj_circuit.x(n)
dj_circuit.h(n)
dj_circuit.draw(output='text',style='bw')
```




```
dj_circuit = QuantumCircuit(n+1, n)
for qubit in range(n):
    dj_circuit.h(qubit)
dj_circuit.x(n)
dj_circuit.h(n)
dj_circuit = dj_circuit.compose(balanced_oracle)
dj_circuit.draw(output='text',style='bw')
```




```
dj_circuit = QuantumCircuit(n+1, n)
for qubit in range(n):
    dj_circuit.h(qubit)
dj_circuit.x(n)
dj_circuit.h(n)
dj_circuit = dj_circuit.compose(balanced_oracle)
for qubit in range(n):
    dj_circuit.h(qubit)
dj_circuit.barrier()
for i in range(n):
    dj_circuit.measure(i, i)
dj_circuit.draw(output='text',style='bw')
```



```
c: 3/═══════════════════════════════════════════════════════════╩═══╩═══╩═══
                                                                0   1   2
```

```
1 aer_sim = qiskit_aer.Aer.get_backend('aer_simulator')
2 results = aer_sim.run(dj_circuit).result()
3 answer = results.get_counts()
4 plot_histogram(answer)
```