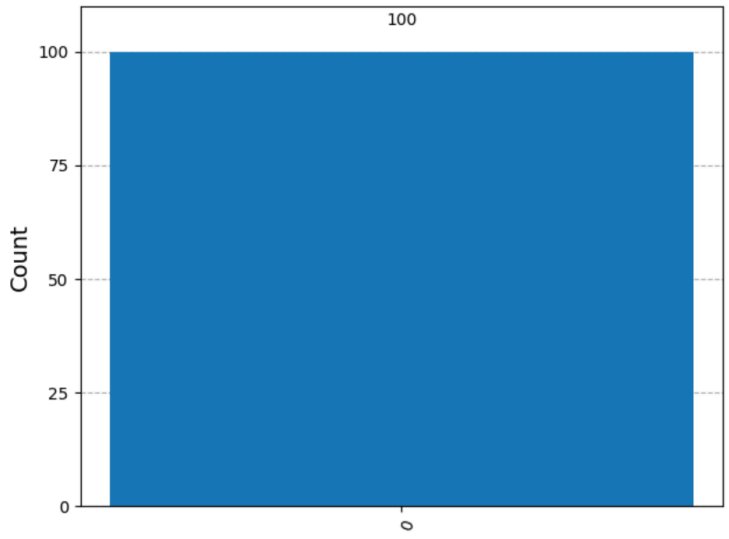
CASE 1:

```python
from qiskit import QuantumCircuit, transpile
from qiskit_aer import AerSimulator
from qiskit.visualization import plot_histogram

qc = QuantumCircuit(1, 1)


qc.h(0)
qc.h(0)
qc.h(0)
qc.h(0)

qc.measure(0, 0)

simulator = AerSimulator()

compiled_circuit = transpile(qc, simulator)

job = simulator.run(compiled_circuit, shots=100)
result = job.result()
counts = result.get_counts()

print(qc.draw())
print("Measurement counts:", counts)

total_shots = sum(counts.values())
print("Probabilities:")
for state, count in counts.items():
    print(f"State |{state}>: {count / total_shots:.3f}")

max_state = max(counts, key=counts.get)
print(f"Most frequent collapsed state after measurement: |{max_state}>")

plot_histogram(counts)
```

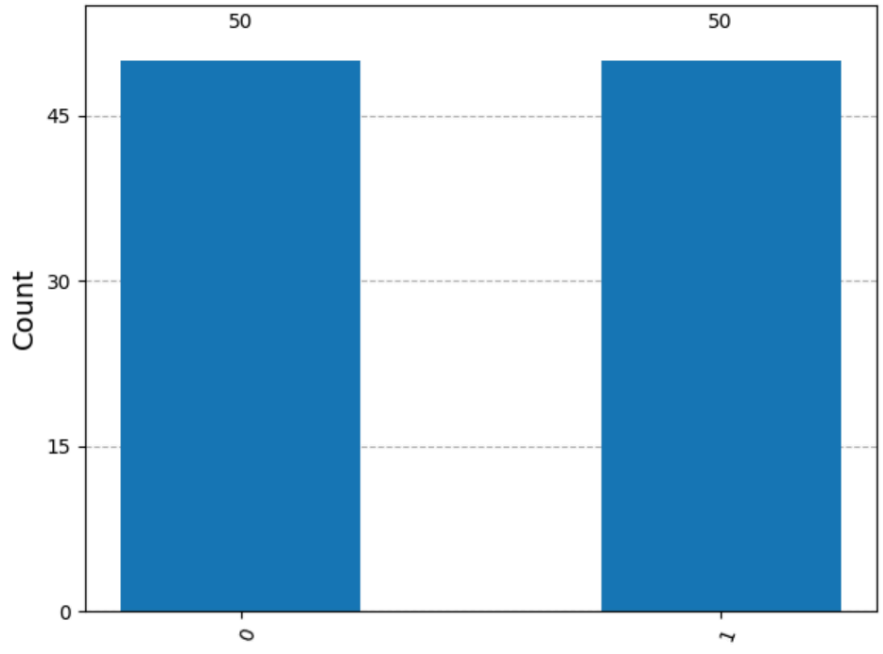
```
q: ┤ H ├┤ H ├┤ H ├┤ H ├┤ M ├
c: 1/═══════════════════╩═══
                           0
Measurement counts: {'0': 100}
Probabilities:
State |0>: 1.000
Most frequent collapsed state after measurement: |0>
```



CASE : 2

```python
from qiskit import QuantumCircuit, transpile
from qiskit_aer import AerSimulator
from qiskit.visualization import plot_histogram

qc = QuantumCircuit(1, 1)
qc.x(0)
qc.h(0)
qc.h(0)
qc.h(0)
qc.measure(0, 0)

simulator = AerSimulator()
compiled_circuit = transpile(qc, simulator)
job = simulator.run(compiled_circuit, shots=100)
result = job.result()
counts = result.get_counts()

print(qc.draw())
print("Measurement counts:", counts)

total_shots = sum(counts.values())
print("Probabilities:")
for state, count in counts.items():
    print(f"State |{state}>: {count / total_shots:.3f}")

max_state = max(counts, key=counts.get)
print(f"Most frequent collapsed state after measurement: |{max_state}>")

plot_histogram(counts)
```

```
     ┌───┐┌───┐┌───┐┌───┐┌─┐
q: ──┤ X ├┤ H ├┤ H ├┤ H ├┤M├
     └───┘└───┘└───┘└───┘└╥┘
c: 1/═══════════════════════╩═
                            0
Measurement counts: {'0': 50, '1': 50}
Probabilities:
State |0>: 0.500
State |1>: 0.500
Most frequent collapsed state after measurement: |0>
```



CASE : 3

```python
from qiskit import QuantumCircuit, transpile
from qiskit_aer import AerSimulator
from qiskit.visualization import plot_histogram

qc = QuantumCircuit(1, 1)
qc.x(0)
qc.z(0)
qc.h(0)
qc.y(0)
qc.s(0)
qc.measure(0, 0)

simulator = AerSimulator()
compiled_circuit = transpile(qc, simulator)
job = simulator.run(compiled_circuit, shots=100)
result = job.result()
counts = result.get_counts()

print(qc.draw())
print("Measurement counts:", counts)

total_shots = sum(counts.values())
print("Probabilities:")
for state, count in counts.items():
    print(f"State |{state}>: {count / total_shots:.3f}")

max_state = max(counts, key=counts.get)
print(f"Most frequent collapsed state after measurement: |{max_state}>")

plot_histogram(counts)
```

```
     ┌───┐┌───┐┌───┐┌───┐┌───┐┌─┐
q: ──┤ X ├┤ Z ├┤ H ├┤ Y ├┤ S ├┤M├
     └───┘└───┘└───┘└───┘└───┘└╥┘
c: 1/══════════════════════════╩═
                               0
Measurement counts: {'0': 56, '1': 44}
Probabilities:
State |0>: 0.560
State |1>: 0.440
Most frequent collapsed state after measurement: |0>
```