Experiment No:- 2 (Implementation of Linear algebra on Quiskit)

Aim :- To Implement State Vectors Using Qiskit -

A) Creation of state Vectros.

B) Check the vectors are valid or not.

C) Display the vectors using histogram.

```
!pip install qiskit
```

```
Requirement already satisfied: qiskit in /usr/local/lib/python3.10/dist-packages (1.2.4)
Requirement already satisfied: rustworkx>=0.15.0 in /usr/local/lib/python3.10/dist-packages (from qiskit) (0.15.1)
Requirement already satisfied: numpy<3,>=1.17 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.26.4)
Requirement already satisfied: scipy>=1.5 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.13.1)
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.13.3)
Requirement already satisfied: dill>=0.3 in /usr/local/lib/python3.10/dist-packages (from qiskit) (0.3.9)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.10/dist-packages (from qiskit) (2.8.2)
Requirement already satisfied: stevedore>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from qiskit) (5.3.0)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from qiskit) (4.12.2)
Requirement already satisfied: symengine<0.14,>=0.11 in /usr/local/lib/python3.10/dist-packages (from qiskit) (0.13.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.0->qiskit) (1.16.0)
Requirement already satisfied: pbr>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from stevedore>=3.0.0->qiskit) (6.1.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy>=1.3->qiskit) (1.3.0)
```

```
from qiskit import quantum_info
from qiskit.quantum_info import Statevector
from numpy import sqrt
u =Statevector([1 / sqrt(2), 1 / sqrt(2)])
v =Statevector([(1 + 2.0j) / 3,-2 / 3])
w=Statevector([1 / 3, 2 / 3])
x =Statevector([ 3, 2])
print("State vectors u, v, and w have been defined.")
```

```
State vectors u, v, and w have been defined.
```

```
display(u.draw("latex"))

display(v.draw("latex"))

display(w.draw("latex"))

display(x.draw("latex"))
```

$$\frac{\sqrt{2}}{2}|0\rangle + \frac{\sqrt{2}}{2}|1\rangle$$

$$(\frac{1}{3} + \frac{2i}{3})|0\rangle - \frac{2}{3}|1\rangle$$

$$\frac{1}{3}|0\rangle + \frac{2}{3}|1\rangle$$

$$3|0\rangle + 2|1\rangle$$

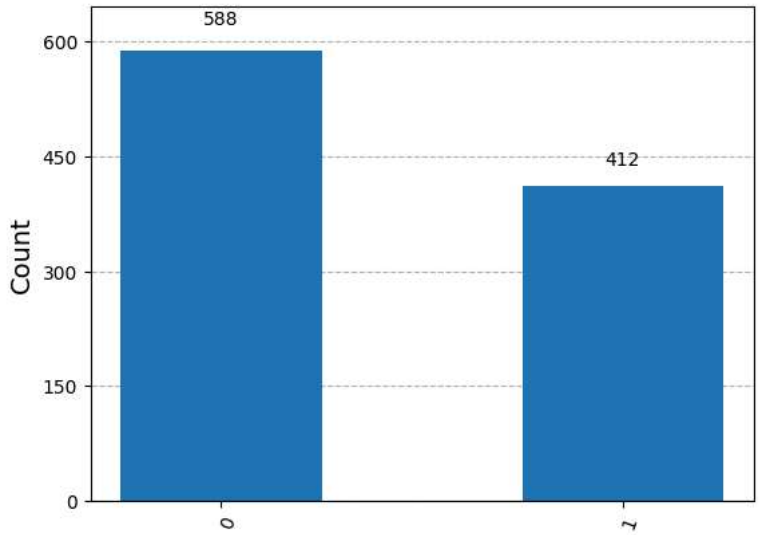
+ Code + Text

```
display(u.is_valid())
display(v.is_valid())
display(w.is_valid())
display(x.is_valid())
```

```
True
True
False
False
```

```
from qiskit.visualization import plot_histogram
statistics = v.sample_counts(1000)
display(statistics)
plot_histogram(statistics)
```

```
{'0': 588, '1': 412}
```



Experiment No:- 3 (Linear algebra: Vector operations, Vector multiplication, Tensor products).

Aim :- To perform the Linear Algebra Operations like Vector Operations, Vector Multiplication, And Tensor Products Successfully.

```
!pip install qiskit
```

```
Collecting qiskit
  Downloading qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting rustworkx>=0.15.0 (from qiskit)
  Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.9 kB)
Requirement already satisfied: numpy<3,>=1.17 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.26.4)
Requirement already satisfied: scipy>=1.5 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.13.1)
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.13.3)
Collecting dill>=0.3 (from qiskit)
  Downloading dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.10/dist-packages (from qiskit) (2.8.2)
Collecting stevedore>=3.0.0 (from qiskit)
  Downloading stevedore-5.3.0-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from qiskit) (4.12.2)
Collecting symengine<0.14,>=0.11 (from qiskit)
  Downloading symengine-0.13.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (1.2 kB)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.0->qiskit) (1.16.0)
Collecting pbr>=2.0.0 (from stevedore>=3.0.0->qiskit)
  Downloading pbr-6.1.0-py2.py3-none-any.whl.metadata (3.4 kB)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy>=1.3->qiskit) (1.3.0)
Downloading qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.8/4.8 MB 26.0 MB/s eta 0:00:00
Downloading dill-0.3.9-py3-none-any.whl (119 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 119.4/119.4 kB 6.4 MB/s eta 0:00:00
Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 36.7 MB/s eta 0:00:00
Downloading stevedore-5.3.0-py3-none-any.whl (49 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 kB 2.5 MB/s eta 0:00:00
Downloading symengine-0.13.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (49.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 MB 13.8 MB/s eta 0:00:00
Downloading pbr-6.1.0-py2.py3-none-any.whl (108 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 108.5/108.5 kB 4.7 MB/s eta 0:00:00
Installing collected packages: symengine, rustworkx, pbr, dill, stevedore, qiskit
Successfully installed dill-0.3.9 pbr-6.1.0 qiskit-1.2.4 rustworkx-0.15.1 stevedore-5.3.0 symengine-0.13.0
```

Inner Product / Dot Product Of Two Vectors:-

```
from qiskit.quantum_info import Statevector
import numpy as np
# State vector for |0⟩
state_vector_0 = Statevector.from_label('0')
# State vector for |1⟩
state_vector_1 = Statevector.from_label('1')
# Convert state vectors to numpy arrays
vec_0 = state_vector_0.data
vec_1 = state_vector_1.data
# Compute the dot product (inner product)
dot_product = np.vdot(vec_0, vec_1)
print("Dot product:", dot_product)
```

```
Dot product: 0j
```

Tensor Product Of Two Vectors:-

```
# Compute the tensor product
tensor_product = state_vector_0.tensor(state_vector_1)
print("Tensor product:", tensor_product)
from qiskit.quantum_info import Statevector
import numpy as np

# Step 1: Create quantum state vectors
state_vector_0 = Statevector.from_label('0')
state_vector_1 = Statevector.from_label('1')

# Step 2: Compute the dot product (inner product)
vec_0 = state_vector_0.data
vec_1 = state_vector_1.data
dot_product = np.vdot(vec_0, vec_1)
print("Dot product:", dot_product)
```

```
# Step 3: Compute the tensor product
tensor_product = state_vector_0.tensor(state_vector_1)
print("Tensor product:", tensor_product)
```

```
Tensor product: Statevector([0.+0.j, 1.+0.j, 0.+0.j, 0.+0.j],
            dims=(2, 2))
Dot product: 0j
Tensor product: Statevector([0.+0.j, 1.+0.j, 0.+0.j, 0.+0.j],
            dims=(2, 2))
```

Experiment No :- 4 (Implementation of Identity matrix:1 Qubit, 2 Qubits, 3 Qubits).

Aim :- To Implement The Identity Matrix Of 1 Qubit, 2 Qubits And 3 Qubits Using Qiksit.

```
!pip install qiskit
```

```
Collecting qiskit
  Downloading qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting rustworkx>=0.15.0 (from qiskit)
  Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.9 kB)
Requirement already satisfied: numpy<3,>=1.17 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.26.4)
Requirement already satisfied: scipy>=1.5 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.13.1)
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.10/dist-packages (from qiskit) (1.13.3)
Collecting dill>=0.3 (from qiskit)
  Downloading dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.10/dist-packages (from qiskit) (2.8.2)
Collecting stevedore>=3.0.0 (from qiskit)
  Downloading stevedore-5.3.0-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from qiskit) (4.12.2)
Collecting symengine<0.14,>=0.11 (from qiskit)
  Downloading symengine-0.13.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (1.2 kB)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.0->qiskit) (1.16.0)
Collecting pbr>=2.0.0 (from stevedore>=3.0.0->qiskit)
  Downloading pbr-6.1.0-py2.py3-none-any.whl.metadata (3.4 kB)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy>=1.3->qiskit) (1.3.0)
Downloading qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.8/4.8 MB 16.1 MB/s eta 0:00:00
Downloading dill-0.3.9-py3-none-any.whl (119 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 119.4/119.4 kB 5.0 MB/s eta 0:00:00
Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 36.7 MB/s eta 0:00:00
Downloading stevedore-5.3.0-py3-none-any.whl (49 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 kB 3.0 MB/s eta 0:00:00
Downloading symengine-0.13.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (49.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 MB 10.3 MB/s eta 0:00:00
Downloading pbr-6.1.0-py2.py3-none-any.whl (108 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 108.5/108.5 kB 2.2 MB/s eta 0:00:00
Installing collected packages: symengine, rustworkx, pbr, dill, stevedore, qiskit
Successfully installed dill-0.3.9 pbr-6.1.0 qiskit-1.2.4 rustworkx-0.15.1 stevedore-5.3.0 symengine-0.13.0
```

```
from qiskit import QuantumCircuit
from qiskit.quantum_info import Operator

# Function to create identity matrix for n qubits
def identity_matrix(n_qubits):
    """
    Creates an identity matrix for a given number of qubits.

    Args:
        n_qubits (int): The number of qubits.

    Returns:
        numpy.ndarray: The identity matrix.
    """
    # Create a quantum circuit with n qubits
    qc = QuantumCircuit(n_qubits)
    # Apply the identity gate to all qubits using the `id()` method
    for i in range(n_qubits):
        qc.id(i)
    # Convert the quantum circuit to an operator (matrix)
    identity_matrix_nq = Operator(qc).data
    return identity_matrix_nq

# Example: Identity matrix for 1, 2, and 3 qubits
identity_matrix_1q = identity_matrix(1)  # Pass 1 as the argument for n_qubits
identity_matrix_2q = identity_matrix(2)  # Pass 2 as the argument for n_qubits
identity_matrix_3q = identity_matrix(3)  # Pass 3 as the argument for n_qubits

print("Identity matrix for 1 qubit:")
print(identity_matrix_1q)
print("\nIdentity matrix for 2 qubits:")
print(identity_matrix_2q)
print("\nIdentity matrix for 3 qubits:")
print(identity_matrix_3q)
```

```
Identity matrix for 1 qubit:
[[1.+0.j 0.+0.j]
 [0.+0.j 1.+0.j]]

Identity matrix for 2 qubits:
[[1.+0.j 0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 1.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 1.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j 1.+0.j]]

Identity matrix for 3 qubits:
[[1.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 1.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 1.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j 1.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j 0.+0.j 1.+0.j 0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 1.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 1.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 0.+0.j 1.+0.j]]
```

Experiment No :- 5 (Implementation of 1- Qubit Gate).

Aim :- Implementation of 1- Qubit Gates -

a) Pauli-X, b) Pauli Y, c) Pauli Z gate, d) HadamardGate.

```
!pip install qiskit
```

```
Requirement already satisfied: qiskit in /usr/local/lib/python3.10/dist-packages (1.2.4)
Requirement already satisfied: rustworkx>=0.15.0 in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: numpy<3,>=1.17 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: scipy>=1.5 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: dill>=0.3 in /usr/local/lib/python3.10/dist-packages (frc
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: stevedore>=3.0.0 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: symengine<0.14,>=0.11 in /usr/local/lib/python3.10/dist-p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: pbr>=2.0.0 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-pack
```

Create a Quantum Circuit :-

```
from qiskit import QuantumCircuit
qc = QuantumCircuit(2)
qc.qubits
```

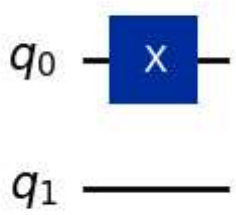
```
[Qubit(QuantumRegister(2, 'q'), 0), Qubit(QuantumRegister(2, 'q'), 1)]
```

Add X-Gate to Qubits 0 :-

```
qc.x(0) # Add X-gate to qubit 0
qc.data
```

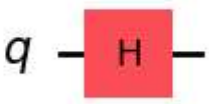
```
[CircuitInstruction(operation=Instruction(name='x', num_qubits=1, num_clbits=0, params=
[]), qubits=(Qubit(QuantumRegister(2, 'q'), 0),), clbits=())]
```

```
qc.draw("mpl")
```



Add H-gate to Qubit 0 :-

```
from qiskit.circuit.library import HGate
qc = QuantumCircuit(1)
qc.append(
HGate(), # New HGate instruction
[0]
)
# Apply to qubit 0
qc.draw("mpl")
```

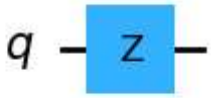


Add Y-Gate to Qubits 0 :-

```
from qiskit import QuantumCircuit
# Create a quantum circuit with 1 qubit
qc = QuantumCircuit(1)
# Apply a Y gate to qubit 0
qc.y(0)
# Draw the circuit using matplotlib
qc.draw('mpl')
```

Add Z-Gate to Qubits 0 :-

```
from qiskit import QuantumCircuit
# Create a quantum circuit with 1 qubit
qc = QuantumCircuit(1)
# Apply a Z gate to qubit 0
qc.z(0)
# Drawthe circuit using matplotlib
qc.draw('mpl')
```



Experiment No :- 5 (Implementation of 2- Quibits Quantum Gates).

Aim :- To Do The Implementation Of 2-Qubits Quantum Gates :-
A)CNOT Gate. B) Phase Gate. C)Swap Gate.

```
import numpy as np
# Function to apply a quantum gate to a two-qubit state
def apply_gate(state, gate):
  return np.dot(gate, state)

# Define the initial state |00>
initial_state = np.array([1, 0, 0, 0]) # Corresponds to |00>

# CNOTGate
CNOT=np.array([[1, 0, 0, 0],
[0, 1, 0, 0],
[0, 0, 0, 1],
[0, 0, 1, 0]])

# Apply CNOT Gate
final_state_CNOT = apply_gate(initial_state, CNOT)
print("State after CNOT Gate:", final_state_CNOT)

# Phase Gate (S Gate)
Phase = np.array([[1, 0],
[0, 1]]) # Phase gate with theta = pi/2

# Apply Phase Gate to the first qubit
# Weneedtocreate a newstate that represents |0> on the first qubit
state_after_phase = np.kron(Phase, np.eye(2)).dot(initial_state)
print("State after Phase Gate:", state_after_phase)

# SWAPGate
SWAP=np.array([[1, 0, 0, 0],
[0, 0, 1, 0],
[0, 1, 0, 0],
[0, 0, 0, 1]])

# Apply SWAP Gate
final_state_SWAP = apply_gate(initial_state, SWAP)
print("State after SWAP Gate:", final_state_SWAP)
```

```
State after CNOT Gate: [1 0 0 0]
State after Phase Gate: [1. 0. 0. 0.]
State after SWAP Gate: [1 0 0 0]
```

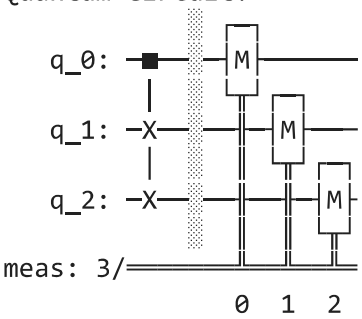
Experiment No :- 6 (Implementation of three Qubits FREDKIN Gate.).

Aim :- To Do The Implementation Of Three Qubits i.e.,FREDKIN gate.

```
# Install qiskit-aer within the current kernel using %pip
!pip install qiskit-aer

# Restart the kernel here before executing the next cell

#Import necessary modules
from qiskit import QuantumCircuit, transpile, assemble
from qiskit.visualization import plot_histogram
# Import AerSimulator explicitly
from qiskit_aer import Aer # Import the Aer provider
from qiskit.primitives import Sampler


# Create a Quantum Circuit with 3 qubits
qc = QuantumCircuit(3)
# Apply the Fredkin gate (control qubit 0, target qubits 1 and 2)
qc.cswap(0, 1, 2)
# Add measurement to all qubits
qc.measure_all()
# Print the circuit diagram
print("Quantum Circuit:")
print(qc.draw())
# Simulate the circuit
# Getting the AerSimulator backend
backend = Aer.get_backend('aer_simulator') # Get a specific simulator backend
compiled_circuit = transpile(qc, backend)
# qobj = assemble(compiled_circuit) # No need for assemble with Sampler
# Initialize Sampler without any arguments
sampler = Sampler()
# Run the sampler and specify the backend in run_options
result = sampler.run(compiled_circuit, shots=1024, run_options={"backend": backend})
# Get the quasi-probabilities instead of counts
quasi_dists = result.quasi_dists
# The quasi_dists will contain a list of dictionaries
# representing the quasi-probability distribution for each circuit.
```

```
Requirement already satisfied: qiskit-aer in /usr/local/lib/python3.10/dist-packages (0.
Requirement already satisfied: qiskit>=1.1.0 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: numpy>=1.16.3 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: scipy>=1.0 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: psutil>=5 in /usr/local/lib/python3.10/dist-packages (frc
Requirement already satisfied: rustworkx>=0.15.0 in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: dill>=0.3 in /usr/local/lib/python3.10/dist-packages (frc
```

```
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: stevedore>=3.0.0 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: symengine<0.14,>=0.11 in /usr/local/lib/python3.10/dist-p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: pbr>=2.0.0 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-pack
Quantum Circuit:
```




```
<ipython-input-8-2a768092a4ad>:29: DeprecationWarning: The class ``qiskit.primitives.sam
  sampler = Sampler()
```

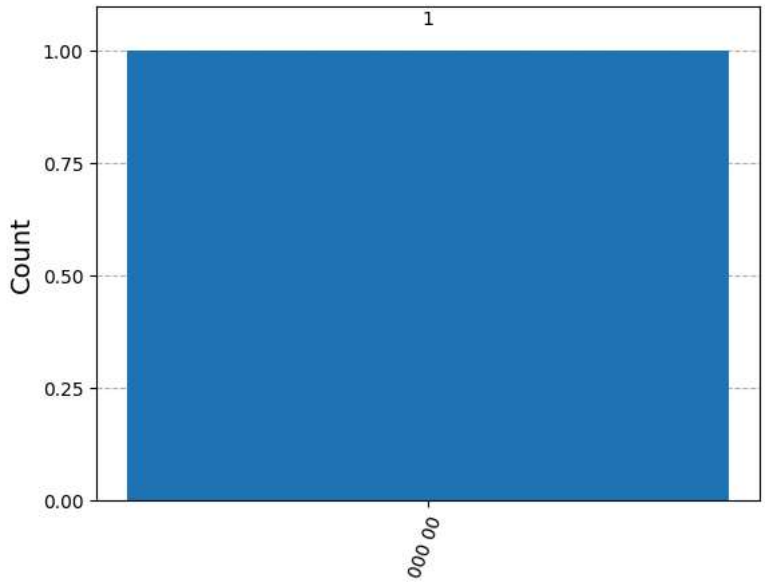
Experiment No:- 7 (Implementation of Circuit Formation-1)

Aim :- Implementation of Circuit Formation-1

```
!pip install pylatexenc
```

```
Collecting pylatexenc
  Downloading pylatexenc-2.10.tar.gz (162 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 162.6/162.6 kB 2.8 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: pylatexenc
  Building wheel for pylatexenc (setup.py) ... done
  Created wheel for pylatexenc: filename=pylatexenc-2.10-py3-none-any.whl size=136817 sha256=1619a55db12750807e3495a0d5809ea38d3b8cc167c
  Stored in directory: /root/.cache/pip/wheels/d3/31/8b/e09b0386afd80cfc556c00408c9aeea5c35c4d484a9c762fd5
Successfully built pylatexenc
Installing collected packages: pylatexenc
Successfully installed pylatexenc-2.10
```

```
import pylatexenc


# Importing necessary libraries
from qiskit import QuantumCircuit, transpile # Import QuantumCircuit and transpile instead of execute
from qiskit_aer import Aer # Import Aer from qiskit_aer
from qiskit.visualization import plot_bloch_multivector, plot_histogram
import numpy as np
import pylatexenc # Import the pylatexenc library

# Create a Quantum Circuit
def quantum_teleportation():
  # Create a Quantum Circuit with 3 qubits and 2 classical bits
  qc = QuantumCircuit(3, 2)
  # Step 1: Prepare the entangled pair (qubit 1 and 2)
  qc.h(1) # Apply Hadamard gate to qubit 1
  qc.cx(1, 2) # Apply CNOT gate to create entanglement
  # Step 2: Prepare the state to be teleported (qubit 0)
  qc.rx(np.pi/2, 0) # Rotate qubit 0 to prepare its state (|+>)
  # Step 3: Bell-state measurement
  qc.cx(0, 1) # CNOT gate
  qc.h(0) # Hadamard gate
  qc.measure(0, 0) # Measure qubit 0
  qc.measure(1, 1) # Measure qubit 1
  # Step 4: Apply corrections based on the measurement results
  qc.cx(1, 2) # CNOT gate
  qc.cz(0, 2) # Control-Z gate
  return qc

# Run the circuit
qc = quantum_teleportation()
qc.draw('text') # Visualize the circuit # This line caused the error

# Simulate the circuit
backend = Aer.get_backend('statevector_simulator')

# Instead of using execute, transpile the circuit and then run it on the backend
qc_compiled = transpile(qc, backend)  # Transpile for the specific backend
result = backend.run(qc_compiled).result()  # Run the transpiled circuit

output_state = result.get_statevector()
# Visualize the Bloch sphere
plot_bloch_multivector(output_state)
# Measure results to classical bits
qc.measure_all()

# Transpile and run for the measurement part as well
qc_compiled = transpile(qc, backend)
counts = backend.run(qc_compiled).result().get_counts()

plot_histogram(counts)
```


1
1.00
0.75
0.50
0.25
0.00
Count
00 000

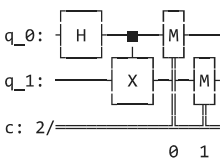
Experiment No:- 8 (Implementation of Circuit Formation-2)

Aim :- To do the successfull Implementation of Circuit Formation-2

```
!pip install qiskit-aer
#importing necessary libraries
from qiskit import QuantumCircuit
from qiskit_aer import Aer
from qiskit.primitives import Sampler #import sample for executing the circuit


#Create a quantum circuit with 2 qubits and 2 classical bits
qc = QuantumCircuit(2,2)
# Apply Hadamard gate to the first qubits to create superposition
qc.h(0)
# Apply CNOT gate with qubit 0 as control and qubit 1 as target
qc.cx(0,1)
# Measure the qubits and store the results in classical bits
qc.measure([0,1],[0,1])
#print the circuit
print("Quantum Circuits : ")
print(qc.draw())
#Use the Aer's qasm_simulator
simulator = Aer.get_backend('qasm_simulator')

# Execute the Circuit using the sampler primitive
sampler = Sampler()
job = sampler.run(qc, shots=1000)

# Grab results from the job
result = job.result()
# Return Counts
counts = result.quasi_dists[0].binary_probabilities() # Get counts from quasi-dists

# Output the results
print("\nMeasurement Results : ")
print(counts)
```

```
Collecting qiskit-aer
  Downloading qiskit_aer-0.15.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (8.0 kB)
Collecting qiskit>=1.1.0 (from qiskit-aer)
  Downloading qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Requirement already satisfied: numpy>=1.16.3 in /usr/local/lib/python3.10/dist-packages (from qiskit-aer) (1.26.4)
Requirement already satisfied: scipy>=1.0 in /usr/local/lib/python3.10/dist-packages (from qiskit-aer) (1.13.1)
Requirement already satisfied: psutil>=5 in /usr/local/lib/python3.10/dist-packages (from qiskit-aer) (5.9.5)
Collecting rustworkx>=0.15.0 (from qiskit>=1.1.0->qiskit-aer)
  Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.9 kB)
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.10/dist-packages (from qiskit>=1.1.0->qiskit-aer) (1.13.3)
Collecting dill>=0.3 (from qiskit>=1.1.0->qiskit-aer)
  Downloading dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.10/dist-packages (from qiskit>=1.1.0->qiskit-aer) (2.8.2
Collecting stevedore>=3.0.0 (from qiskit>=1.1.0->qiskit-aer)
  Downloading stevedore-5.3.0-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from qiskit>=1.1.0->qiskit-aer) (4.12.2)
Collecting symengine<0.14,>=0.11 (from qiskit>=1.1.0->qiskit-aer)
  Downloading symengine-0.13.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (1.2 kB)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.0->qiskit>=1.1.0->qiskit-a
Collecting pbr>=2.0.0 (from stevedore>=3.0.0->qiskit>=1.1.0->qiskit-aer)
  Downloading pbr-6.1.0-py2.py3-none-any.whl.metadata (3.4 kB)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy>=1.3->qiskit>=1.1.0->qiskit-aer
Downloading qiskit_aer-0.15.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.3/12.3 MB 82.7 MB/s eta 0:00:00
Downloading qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.8/4.8 MB 64.6 MB/s eta 0:00:00
Downloading dill-0.3.9-py3-none-any.whl (119 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 119.4/119.4 kB 8.1 MB/s eta 0:00:00
Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 64.0 MB/s eta 0:00:00
Downloading stevedore-5.3.0-py3-none-any.whl (49 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 kB 3.5 MB/s eta 0:00:00
Downloading symengine-0.13.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (49.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 MB 11.0 MB/s eta 0:00:00
Downloading pbr-6.1.0-py2.py3-none-any.whl (108 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 108.5/108.5 kB 7.2 MB/s eta 0:00:00
Installing collected packages: symengine, rustworkx, pbr, dill, stevedore, qiskit, qiskit-aer
Successfully installed dill-0.3.9 pbr-6.1.0 qiskit-1.2.4 qiskit-aer-0.15.1 rustworkx-0.15.1 stevedore-5.3.0 symengine-0.13.0
```

```
Quantum Circuits :
```




```
Measurement Results :
{'00': 0.503, '11': 0.497}
<ipython-input-2-499a31b26de8>:23: DeprecationWarning: The class ``qiskit.primitives.sampler.Sampler`` is deprecated as of qiskit 1.2. I
  sampler = Sampler()
```