|  |  |
| --- | --- |
|  | |
| **Solution** :  dev = qml.device("default.qubit", wires=1)  U = np.array([[1, 1], [1, -1]]) / np.sqrt(2)  @qml.qnode(dev)  def apply\_u():  ##################  qml.QubitUnitary(U, wires=0)  ##################  # USE QubitUnitary TO APPLY U TO THE QUBIT  # Return the state  return qml.state() | |
| **Qiskit Program**:  import numpy as np  import random  from qiskit.quantum\_info import Statevector  import pennylane as qml  import matplotlib.pyplot as plt  dev = qml.device("default.qubit", wires=1)  U = np.array([[1, 1], [1, -1]]) / np.sqrt(2)  @qml.qnode(dev)  def apply\_u():  ##################  qml.QubitUnitary(U, wires=0)  ##################  # USE QubitUnitary TO APPLY U TO THE QUBIT  # Return the state  return qml.state()  circuit = qml.QNode(apply\_u, dev)  qml.drawer.use\_style("pennylane")  result = qml.draw\_mpl(circuit)()  plt.show()  **O/P:** |

|  |
| --- |
|  |
| **Solution:**  dev = qml.device("default.qubit", wires=1)  @qml.qnode(dev)  def apply\_u\_as\_rot(phi, theta, omega):  ##################  qml.Rot(phi, theta, omega, wires=0)  ##################  # APPLY A ROT GATE USING THE PROVIDED INPUT PARAMETERS  # RETURN THE QUANTUM STATE VECTOR  return qml.state() |
| **Qiskit Program**:  import numpy as np  import random  from qiskit.quantum\_info import Statevector  import pennylane as qml  import matplotlib.pyplot as plt  dev = qml.device("default.qubit", wires=1)  U = np.array([[1, 1], [1, -1]]) / np.sqrt(2)  @qml.qnode(dev)  def apply\_u\_as\_rot(phi, theta, omega):  ##################  qml.Rot(phi, theta, omega, wires=0)  ##################  # APPLY A ROT GATE USING THE PROVIDED INPUT PARAMETERS  # RETURN THE QUANTUM STATE VECTOR  return qml.state()    # We set up some values for the input parameters  theta, phi, omega = 0.1, 0.2, 0.3  circuit = qml.QNode(apply\_u\_as\_rot, dev)  specs\_func = qml.specs(circuit)  specs\_func(theta, phi,omega) |