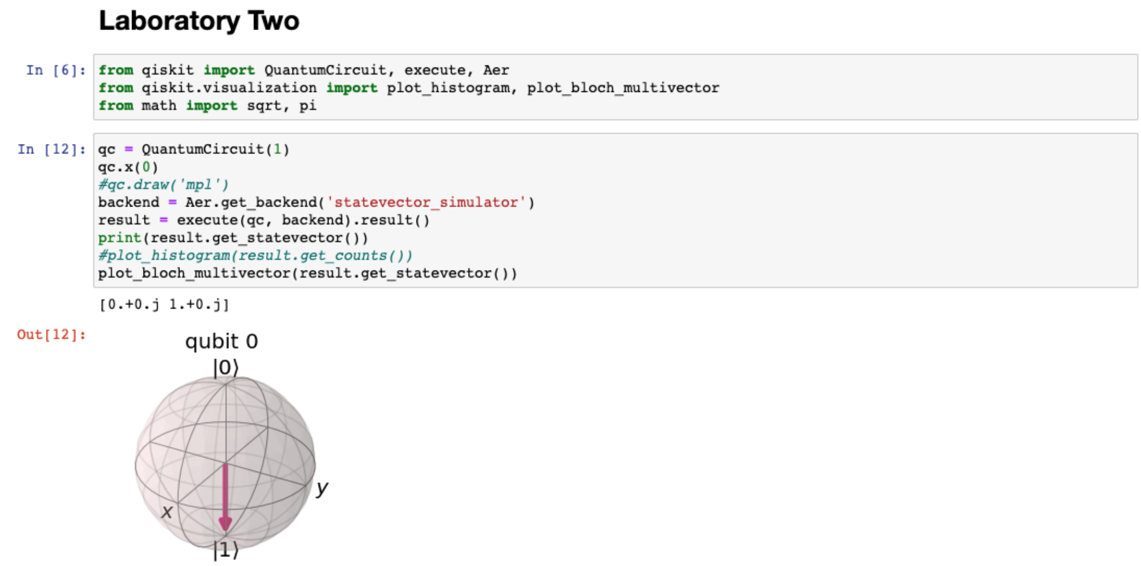
Name: Okoye, Adunife Kizito
Student ID: 100611918
CSCI 4140U



Laboratory Two

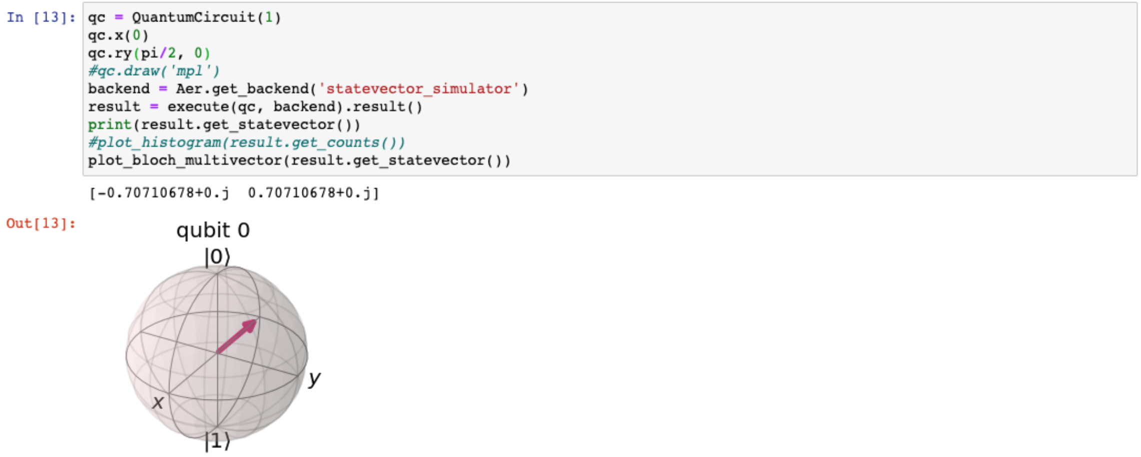
In [6]:
```
from qiskit import QuantumCircuit, execute, Aer
from qiskit.visualization import plot_histogram, plot_bloch_multivector
from math import sqrt, pi
```

In [12]:
```
qc = QuantumCircuit(1)
qc.x(0)
#qc.draw('mpl')
backend = Aer.get_backend('statevector_simulator')
result = execute(qc, backend).result()
print(result.get_statevector())
#plot_histogram(result.get_counts())
plot_bloch_multivector(result.get_statevector())
```

```
[0.+0.j 1.+0.j]
```

Out[12]:

Rotated with pi/2

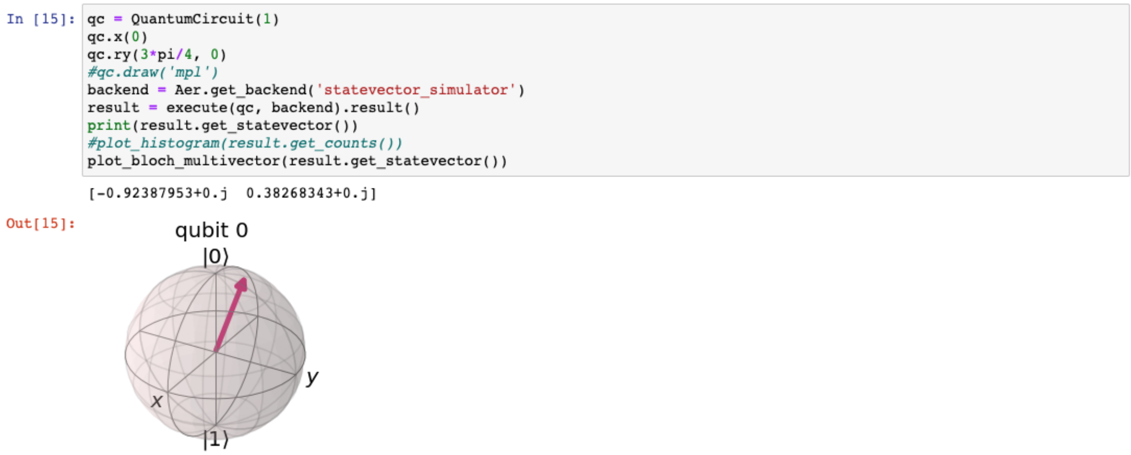


In [13]:
```
qc = QuantumCircuit(1)
qc.x(0)
qc.ry(pi/2, 0)
#qc.draw('mpl')
backend = Aer.get_backend('statevector_simulator')
result = execute(qc, backend).result()
print(result.get_statevector())
#plot_histogram(result.get_counts())
plot_bloch_multivector(result.get_statevector())
```

```
[-0.70710678+0.j  0.70710678+0.j]
```

Out[13]:

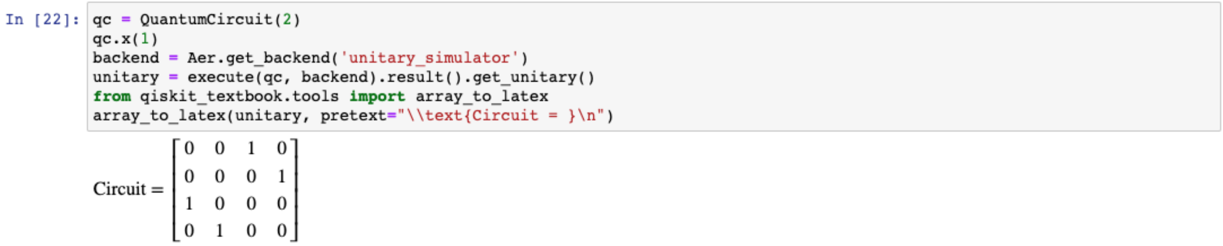
Rotated with 3*pi/4



```
In [15]: qc = QuantumCircuit(1)
         qc.x(0)
         qc.ry(3*pi/4, 0)
         #qc.draw('mpl')
         backend = Aer.get_backend('statevector_simulator')
         result = execute(qc, backend).result()
         print(result.get_statevector())
         #plot_histogram(result.get_counts())
         plot_bloch_multivector(result.get_statevector())

         [-0.92387953+0.j  0.38268343+0.j]
Out[15]:
```

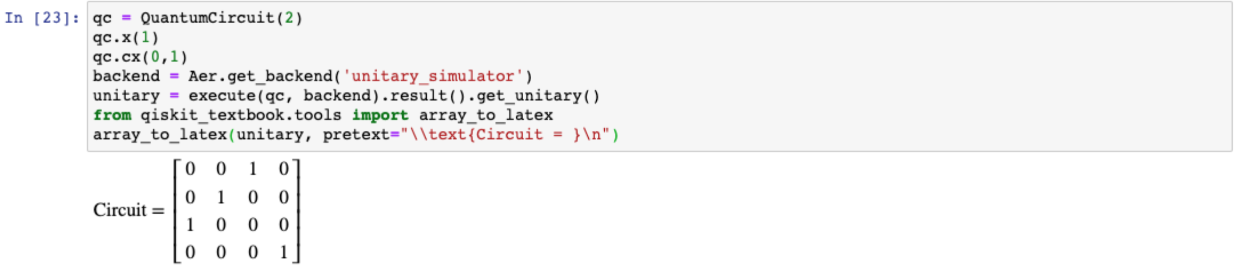
Simulated Qubit



```
In [22]: qc = QuantumCircuit(2)
         qc.x(1)
         backend = Aer.get_backend('unitary_simulator')
         unitary = execute(qc, backend).result().get_unitary()
         from qiskit_textbook.tools import array_to_latex
         array_to_latex(unitary, pretext="\\text{Circuit = }\n")
```

$$\text{Circuit} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

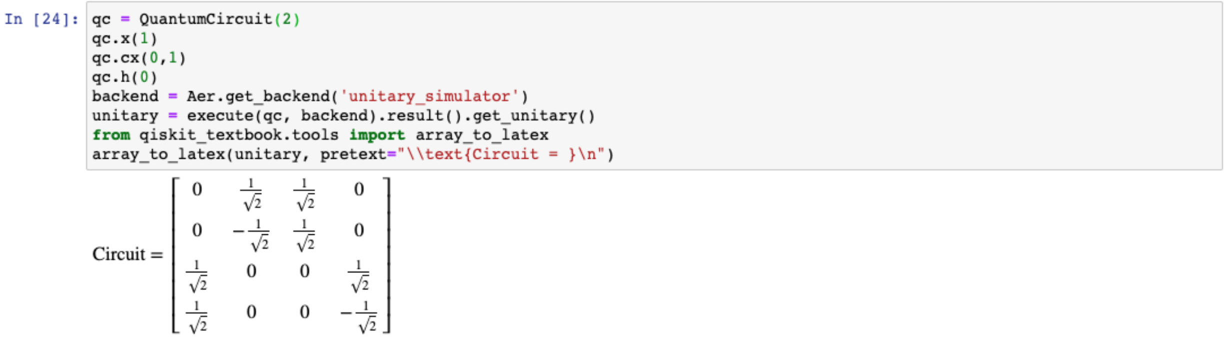
After adding “qc.cx(0.1)” columns 2, 3, and 4 in the matrix changed



```
In [23]: qc = QuantumCircuit(2)
         qc.x(1)
         qc.cx(0,1)
         backend = Aer.get_backend('unitary_simulator')
         unitary = execute(qc, backend).result().get_unitary()
         from qiskit_textbook.tools import array_to_latex
         array_to_latex(unitary, pretext="\\text{Circuit = }\n")
```

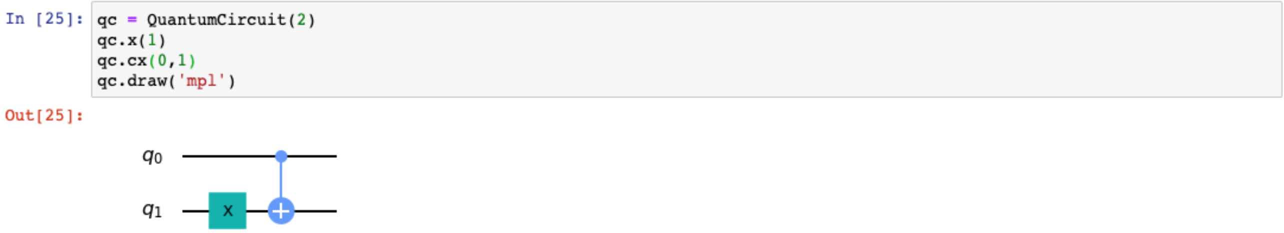
$$\text{Circuit} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

After adding “qc.h(0)” columns 2, 3, and 4 in the matrix changed



```
In [24]: qc = QuantumCircuit(2)
         qc.x(1)
         qc.cx(0,1)
         qc.h(0)
         backend = Aer.get_backend('unitary_simulator')
         unitary = execute(qc, backend).result().get_unitary()
         from qiskit_textbook.tools import array_to_latex
         array_to_latex(unitary, pretext="\\text{Circuit = }\n")
```

$$\text{Circuit} = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & 0 & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

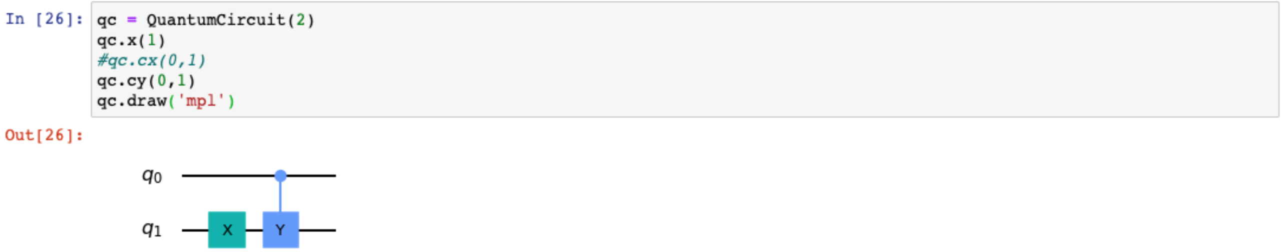


In [25]:

```
qc = QuantumCircuit(2)
qc.x(1)
qc.cx(0,1)
qc.draw('mpl')
```

Out[25]:

After replacing “qc.cx(0,1)” with “qc.cy(0,1)”

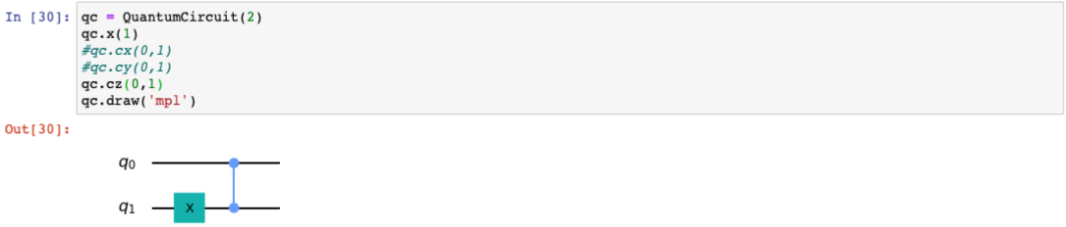


In [26]:

```
qc = QuantumCircuit(2)
qc.x(1)
#qc.cx(0,1)
qc.cy(0,1)
qc.draw('mpl')
```

Out[26]:

After replacing “qc.cy(0,1)” with “qc.cz(0,1)”

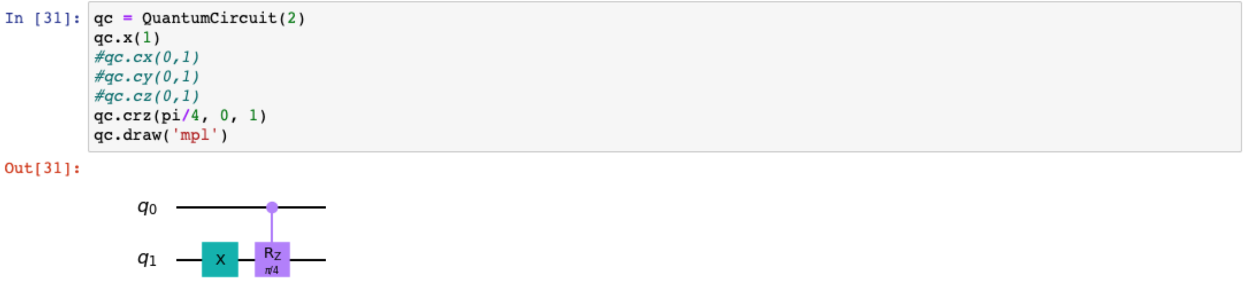


In [30]:

```
qc = QuantumCircuit(2)
qc.x(1)
#qc.cx(0,1)
#qc.cy(0,1)
qc.cz(0,1)
qc.draw('mpl')
```

Out[30]:

After replacing “qc.cz(0,1)” with “qc.crz(pi/40,1)”

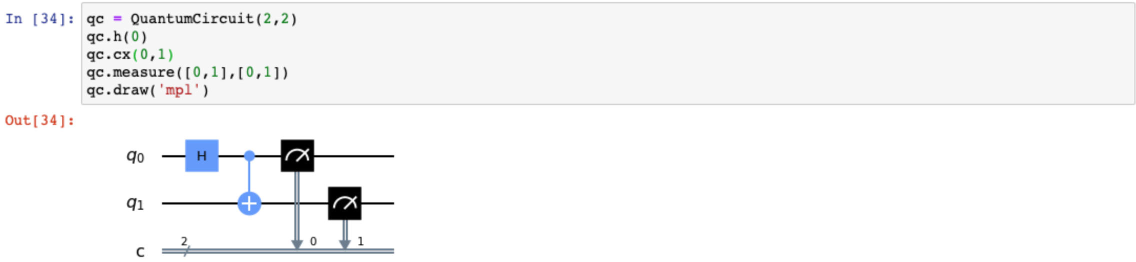


In [31]:

```
qc = QuantumCircuit(2)
qc.x(1)
#qc.cx(0,1)
#qc.cy(0,1)
#qc.cz(0,1)
qc.crz(pi/4, 0, 1)
qc.draw('mpl')
```

Out[31]:

Measurements

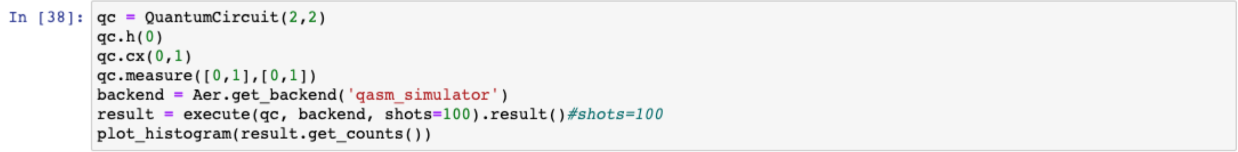


In [34]:

```
qc = QuantumCircuit(2,2)
qc.h(0)
qc.cx(0,1)
qc.measure([0,1],[0,1])
qc.draw('mpl')
```

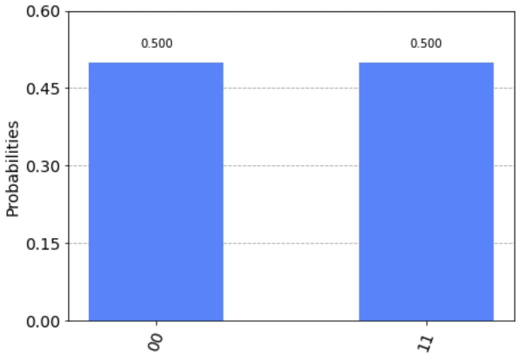
Out[34]:

Histogram plot for shots=100

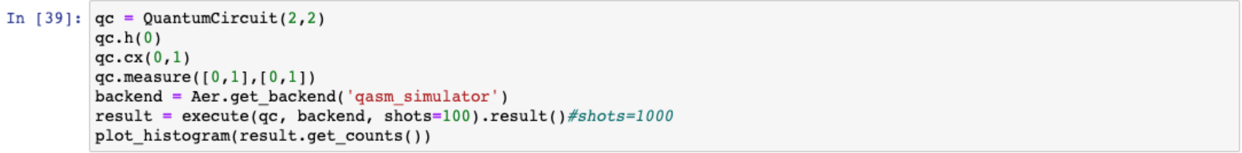


```
In [38]: qc = QuantumCircuit(2,2)
         qc.h(0)
         qc.cx(0,1)
         qc.measure([0,1],[0,1])
         backend = Aer.get_backend('qasm_simulator')
         result = execute(qc, backend, shots=100).result()#shots=100
         plot_histogram(result.get_counts())
```

Out[38]:

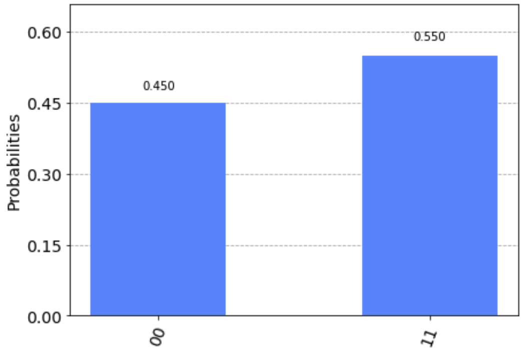


Histogram plot for shots=1000

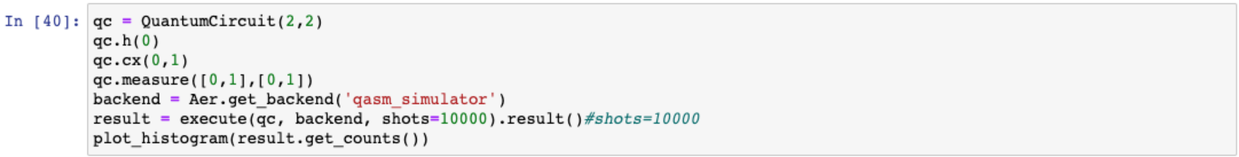


```
In [39]: qc = QuantumCircuit(2,2)
         qc.h(0)
         qc.cx(0,1)
         qc.measure([0,1],[0,1])
         backend = Aer.get_backend('qasm_simulator')
         result = execute(qc, backend, shots=100).result()#shots=1000
         plot_histogram(result.get_counts())
```

Out[39]:

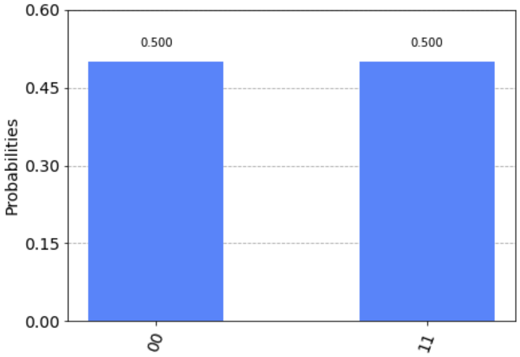


Histogram plot for shots=10000

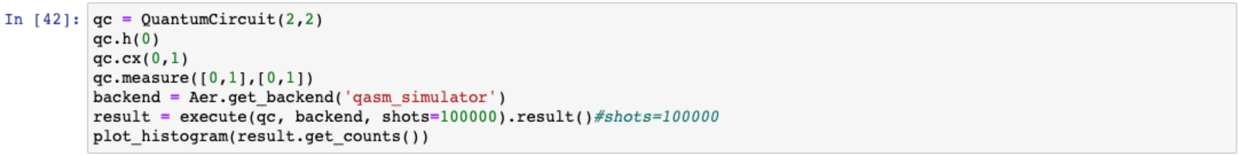


```
In [40]: qc = QuantumCircuit(2,2)
         qc.h(0)
         qc.cx(0,1)
         qc.measure([0,1],[0,1])
         backend = Aer.get_backend('qasm_simulator')
         result = execute(qc, backend, shots=10000).result()#shots=10000
         plot_histogram(result.get_counts())
```

Out[40]:

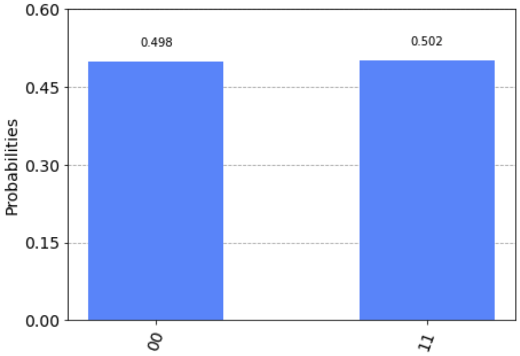


Histogram plot for shots=100000



```
In [42]: qc = QuantumCircuit(2,2)
         qc.h(0)
         qc.cx(0,1)
         qc.measure([0,1],[0,1])
         backend = Aer.get_backend('qasm_simulator')
         result = execute(qc, backend, shots=100000).result()#shots=100000
         plot_histogram(result.get_counts())
```

Out[42]:


Probabilities