In [1]:
```
backend_publicism = Public_Provider.get_backend('ibmq_qasm_simulator')
backend_publiclon = Public_Provider.get_backend('ibmq_ourense')
print(backend_publicism)
print(backend_publiclon)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-8b5f7c7096cd> in <module>
----> 1 backend_publicism = Public_Provider.get_backend('ibmq_qasm_simulator')
      2 backend_publiclon = Public_Provider.get_backend('ibmq_ourense')
      3 print(backend_publicism)
      4 print(backend_publiclon)
      5

NameError: name 'Public_Provider' is not defined
```

In [2]:
```
from qiskit import IBMQ
API_TOKEN ='bf24308d4b9628ea5d8f4a213416453f23eb280986828dbe5d3b691a9a9cd0b40b0b40a2e7741cc22abc447b4feb2ba88a4d51f4d2512c7bca3aa964ef6b1a2c'
IBMQ.save_account(API_TOKEN)

Public_Provider = IBMQ.load_account()
Public_Provider.backends()
```

```
configrc.store_credentials:WARNING:2020-09-10 19:25:30,250: Credentials already present. Set overwrite=True to overwrite.
```

Out[2]:
```
[<IBMQSimulator('ibmq_qasm_simulator') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmqx2') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_16_melbourne') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_vigo') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_ourense') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_valencia') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_london') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_burlington') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_essex') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_armonk') from IBMQ(hub='ibm-q', group='open', project='main')>,
 <IBMQBackend('ibmq_santiago') from IBMQ(hub='ibm-q', group='open', project='main')>]
```

In [3]:
```
backend_publicism = Public_Provider.get_backend('ibmq_qasm_simulator')
backend_publiclon = Public_Provider.get_backend('ibmq_ourense')
print(backend_publicism)
print(backend_publiclon)
```

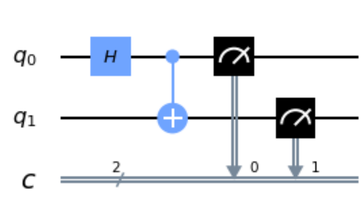
```
ibmq_qasm_simulator
ibmq_ourense
```

In [4]:
```
from qiskit import QuantumRegister, QuantumCircuit, ClassicalRegister
%matplotlib inline
```

In [5]:
```
q = QuantumRegister(2, name = 'q')
c = ClassicalRegister(2, name = 'c')
bell_state = QuantumCircuit(q,c)
```

In [6]:
```
bell_state.h(q[0])
bell_state.cx(q[0], q[1])
bell_state.measure(q,c)
bell_state.draw(output = 'mpl')
```

Out[6]:



In [7]:
```
from qiskit import BasicAer, execute
```

In [8]:
```
from qiskit.tools.monitor import job_monitor
```

In [9]:
```
job = execute(bell_state,backend_publiclon)
```

In [10]:
```
job_monitor(job)
```

```
Job Status: job has successfully run
```

In [11]:
```
result = job.result()
```

In [12]:
```
count = result.get_counts()
```

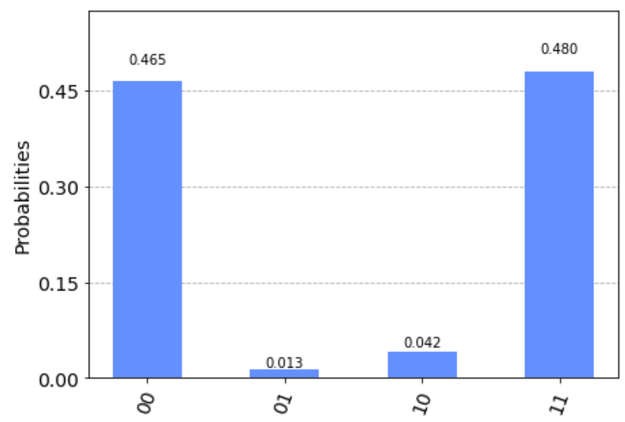
In [13]:
```
print(count)
```

```
{'01': 13, '11': 492, '10': 43, '00': 476}
```

In [14]:
```
from qiskit.tools.visualization import plot_histogram
```

In [15]:
```
plot_histogram(count)
```

Out[15]:



In [ ]:
```

```