In [1]:
```
from qiskit import *
from random import *
```

In [2]:
```
qr=QuantumRegister(1)       #defining quantum register
cr=ClassicalRegister(1)     #defining classical register
circuit=QuantumCircuit(qr,cr)   #defining quantum circuit
%matplotlib inline
circuit.h([0])  #after starting the game quanum computer will  first put the state of coin in superposition
```

Out[2]:
```
<qiskit.circuit.instructionset.InstructionSet at 0x16fab4e49d0>
```

In [3]:
```
# defining function for operation of coin flip
def player(p):

    if p>=0.5:
        circuit.x([0])      #fliping the coin by applying X operation
```

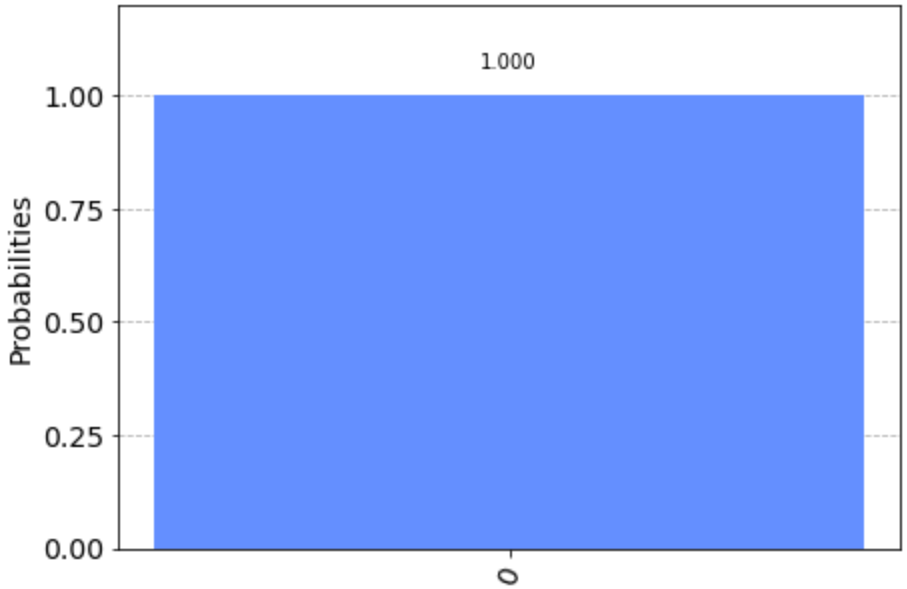
In [4]:
```

    rounds=randint(0,10)  #this decides at what point players reveal their measurement
    for i in range(rounds):
        c=random()
        d=random()
        if c>=0.5:
            a=random()
            player(a)
        else:
            circuit.id([0])
        if d>=0.5:
            b=random()
            player(b)
        else:
            circuit.id([0])
    circuit.h([0])       #before taking the measurement QC applies inverse hadamard in order to collapse the state.
    circuit.measure(qr,cr)   #taking quantum measurement
    circuit.draw(output='mpl')
```

Out[4]:

In [5]:
```
#simulation and obtaining the results
simulator= Aer.get_backend('qasm_simulator')
execute(circuit,backend=simulator)
result = execute(circuit,backend=simulator).result()
counts = result.get_counts()
from qiskit.tools.visualization import plot_histogram
plot_histogram(counts)
```

Out[5]:



In [ ]: