```
In [8]:  %matplotlib inline
         from qiskit import QuantumCircuit, execute, Aer, QuantumRegister, ClassicalRegister, BasicAer
         from qiskit.compiler import transpile, assemble
         from qiskit.tools.jupyter import *
         from qiskit.visualization import *
         from qiskit_ibm_provider import IBMProvider

         provider = IBMProvider()
```

```
In [9]:  def controlled_Z(circuit, control, target):
             circuit.h(target)
             circuit.cnot(control, target)
             circuit.h(target)
```

```
In [10]: def controlled_cz(circuit, control1, control2, target):
             circuit.h(target)
             circuit.ccx(control1, control2, target)
             circuit.h(target)
```

```
In [11]: def phase_oracle(circuit, registers):
             circuit.cnot(registers[0], registers[1])
             circuit.ccx(registers[0], registers[1], registers[2])
             circuit.h(registers[2])
             circuit.ccx(registers[0], registers[1], registers[2])
             circuit.h(registers[2])
             circuit.ccx(registers[0], registers[1], registers[2])
             circuit.cnot(registers[0], registers[1])
```

```
In [12]: def grover_diffusion ( circuit , registers ):
             # Apply Hadamard and X gates on all qubits
             circuit .h( registers )
             circuit .x( registers )

             # Create a barrier that isolates different sections of the circuit
             circuit.barrier()

             # Apply CZ gate with target as qubit 1
             controlled_cz(circuit, registers[0], registers[1], registers[2])

             circuit.barrier()
             circuit.x(registers)
             circuit.h(registers)
```

```
In [13]: # Define circuit constants
         Qubits = 3
         tests = 2

         # Define register objects
         qr = QuantumRegister (Qubits )
         cr = ClassicalRegister (Qubits )
         groverCircuit = QuantumCircuit (qr, cr)

         # Initialize circuit with Hadamard gates
         groverCircuit.h(qr)

         # Run phase oracle and diffusion operators .
         # Can be run multiple times depending on the variable , tests .
         for test in range (0, tests):
             groverCircuit.barrier()
             phase_oracle(groverCircuit, qr)
             groverCircuit.barrier()
             grover_diffusion(groverCircuit, qr)

         # Measure quantum registers
         groverCircuit.barrier ()
         groverCircuit.measure (qr ,cr)

         # Draw Circuit
         groverCircuit.draw (output="mpl")
```
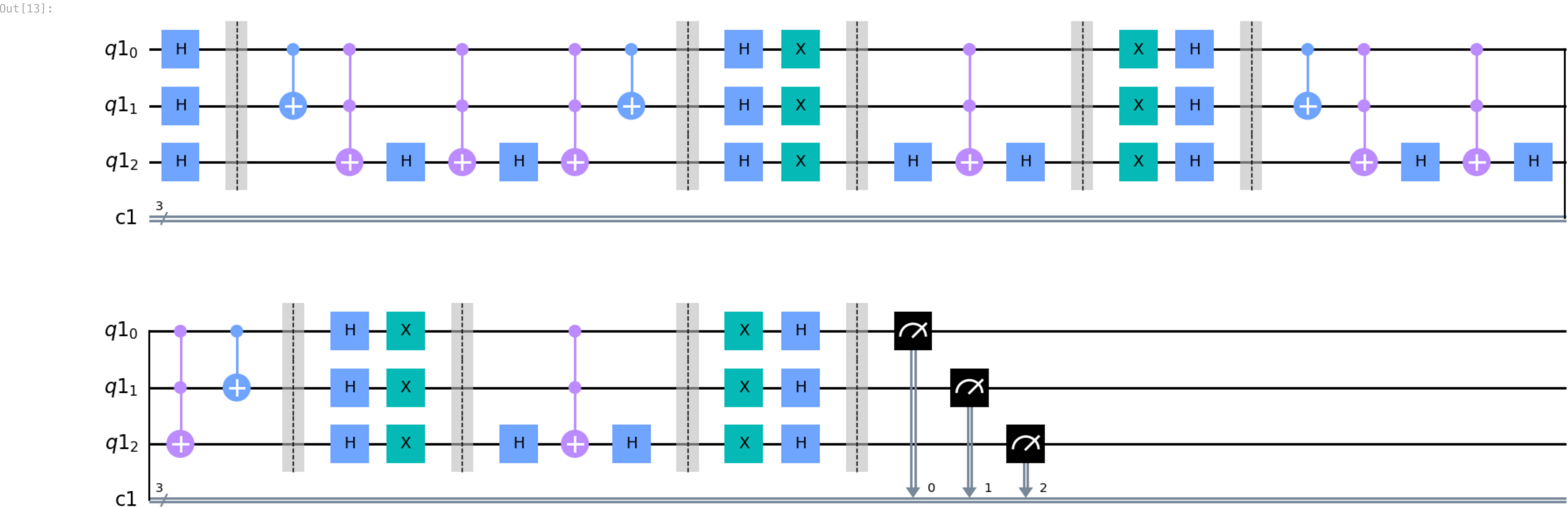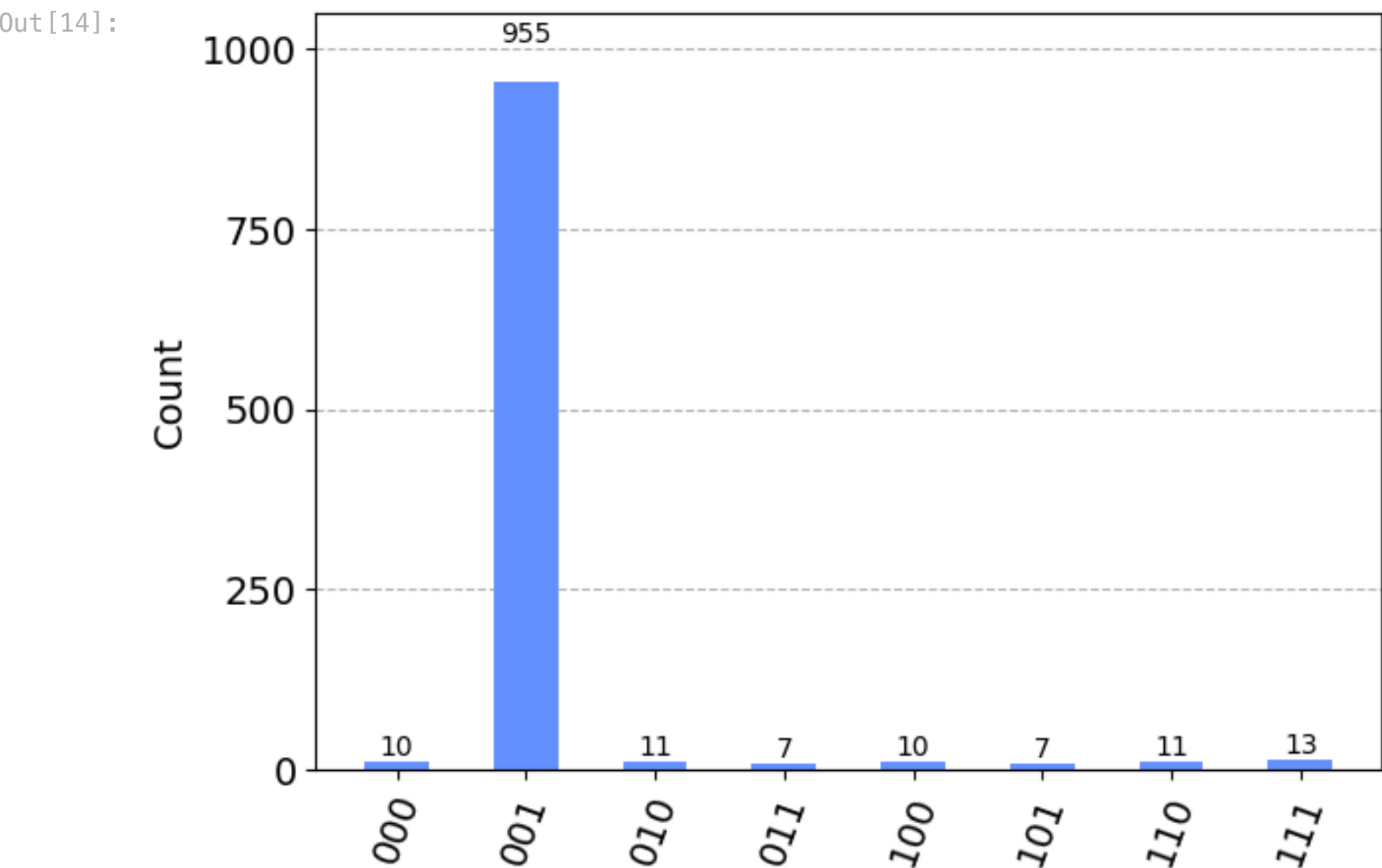
Out[13]:


```
In [14]: # Define backend that will simulate quantum circuit
         backend = BasicAer.get_backend('qasm_simulator')
         # Number of times the circuit is run
         shots = 1024
         # Execute circuit and plot results on histogram
         results = execute (groverCircuit, backend=backend, shots=shots).result()
         answer = results.get_counts()
         plot_histogram(answer)
```

Out[14]:

In [ ]: