QUANTUM PHASE ESTIMATOR (QPE)

Lina LOUATI et Rania FATHALLAH

Tester L'algorithme Quantum Phase Estimator sur T-Gate

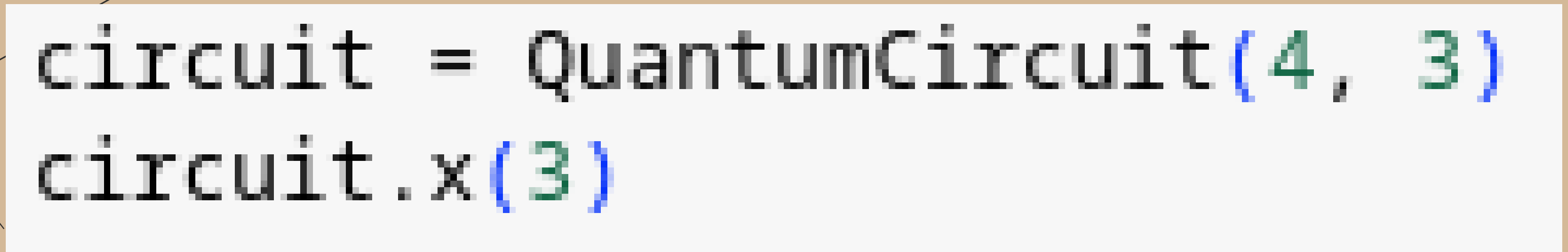
$$T|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = e^{\frac{i\pi}{4}}|1\rangle$$

$$\theta = \frac{1}{8}$$

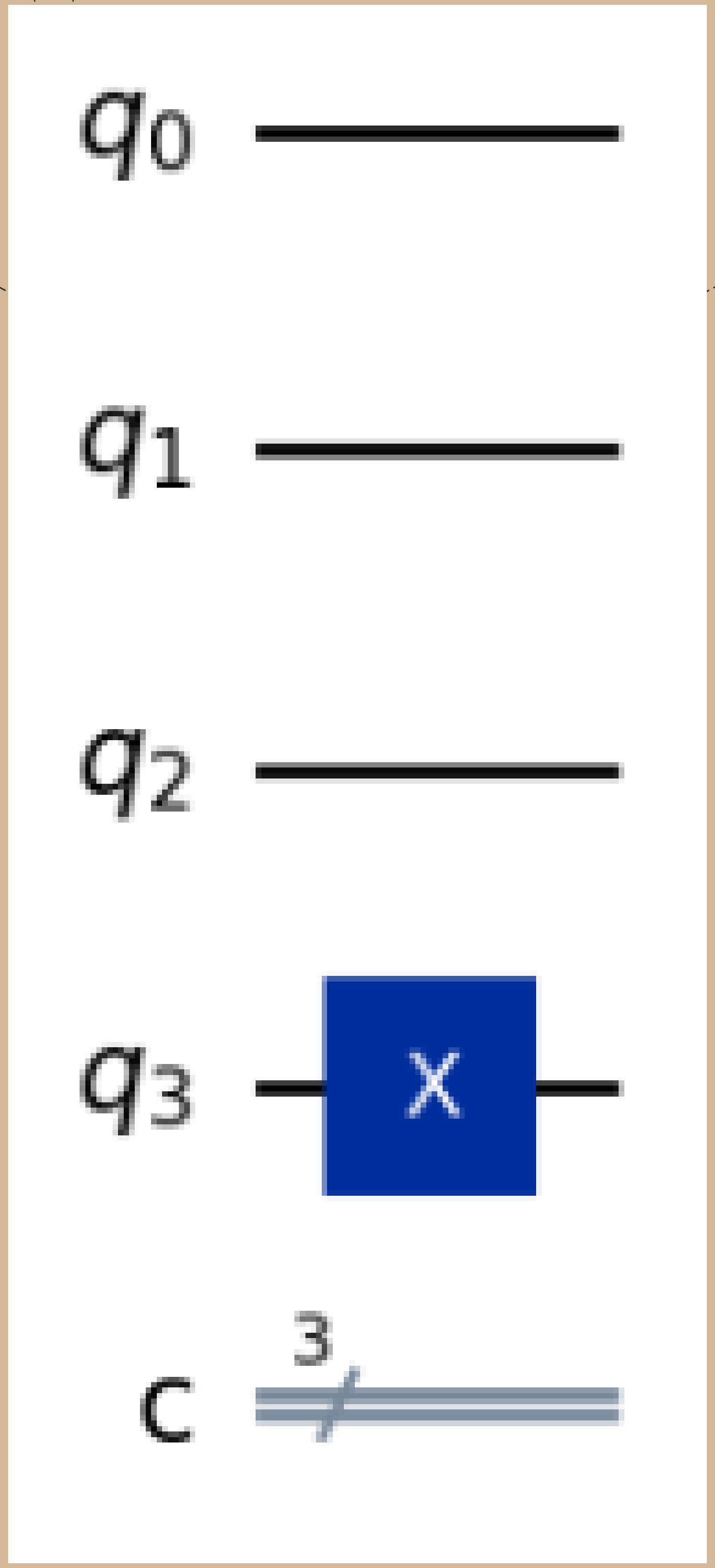
Etapes

1- Initialiser les registres
2- Appliquer les portes de Hadamard au registre de valeur propre
3- Appliquer les portes contrôlées-U
4- Appliquer la Transformée de Fourier Quantique Inverse (QFT Inverse)
5- Mesurer le registre de valeur propre

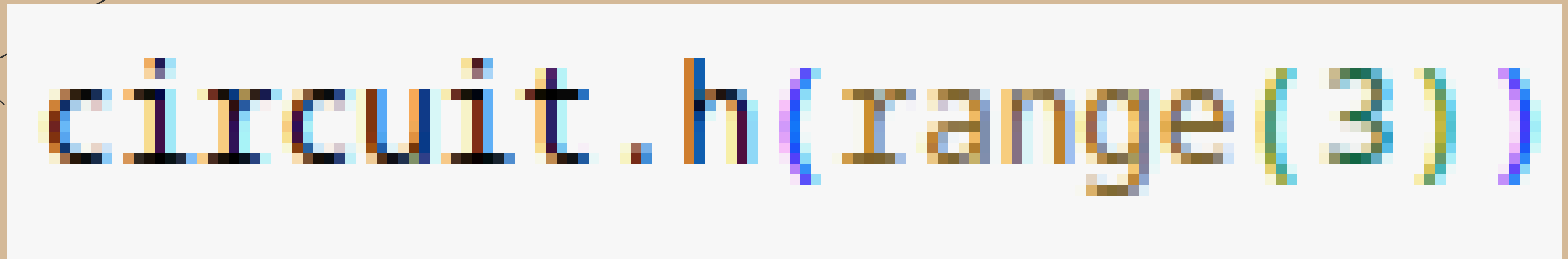
1– Initialiser les registres



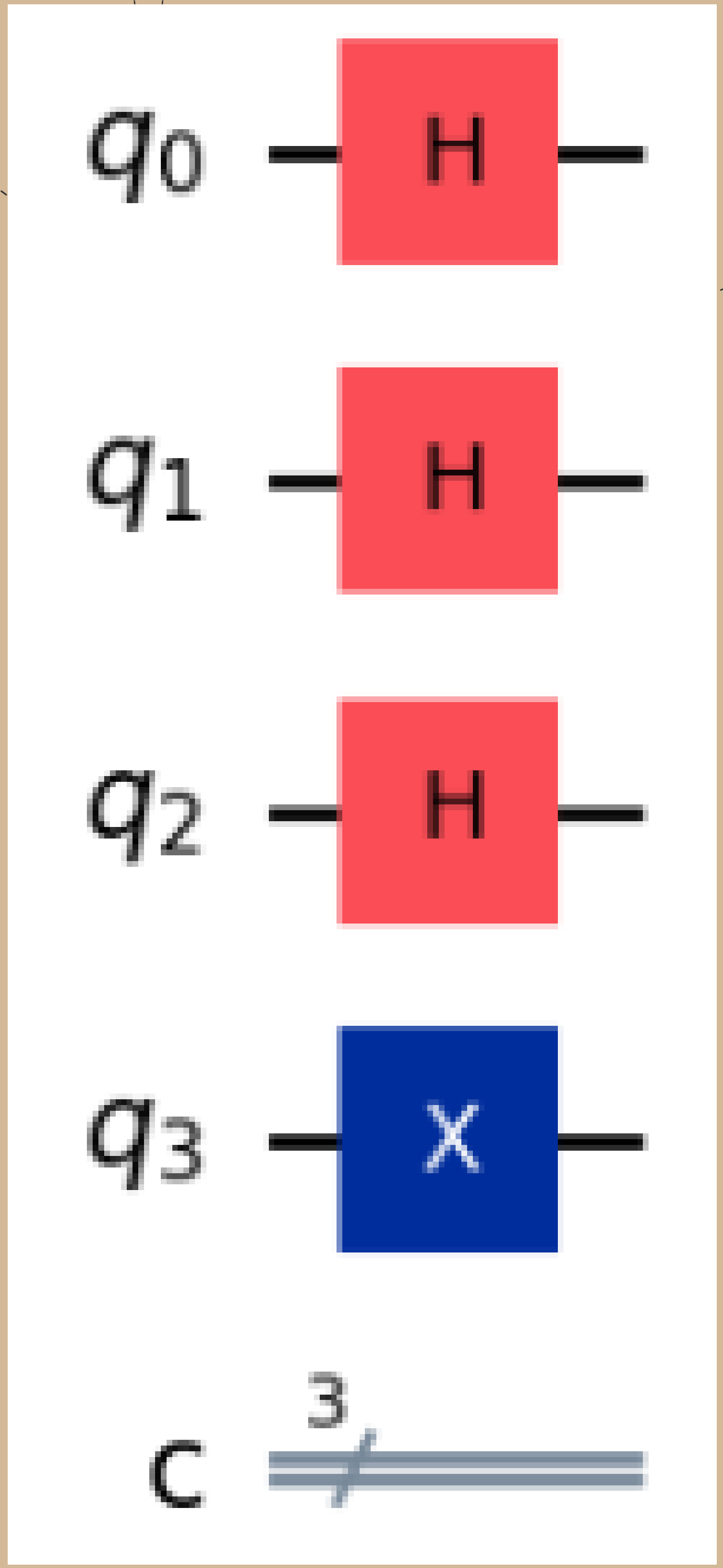
```
circuit = QuantumCircuit(4, 3)
circuit.x(3)
```



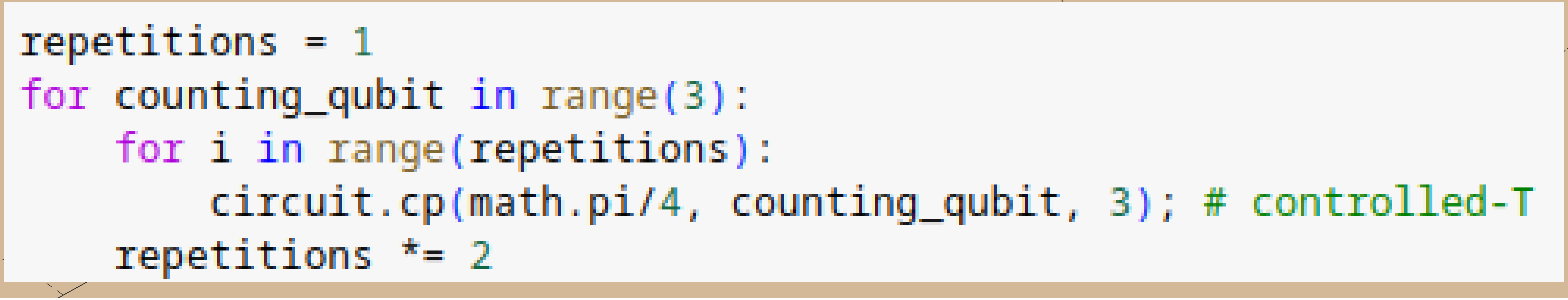
2- Appliquer les portes de Hadamard au registre de valeur propre



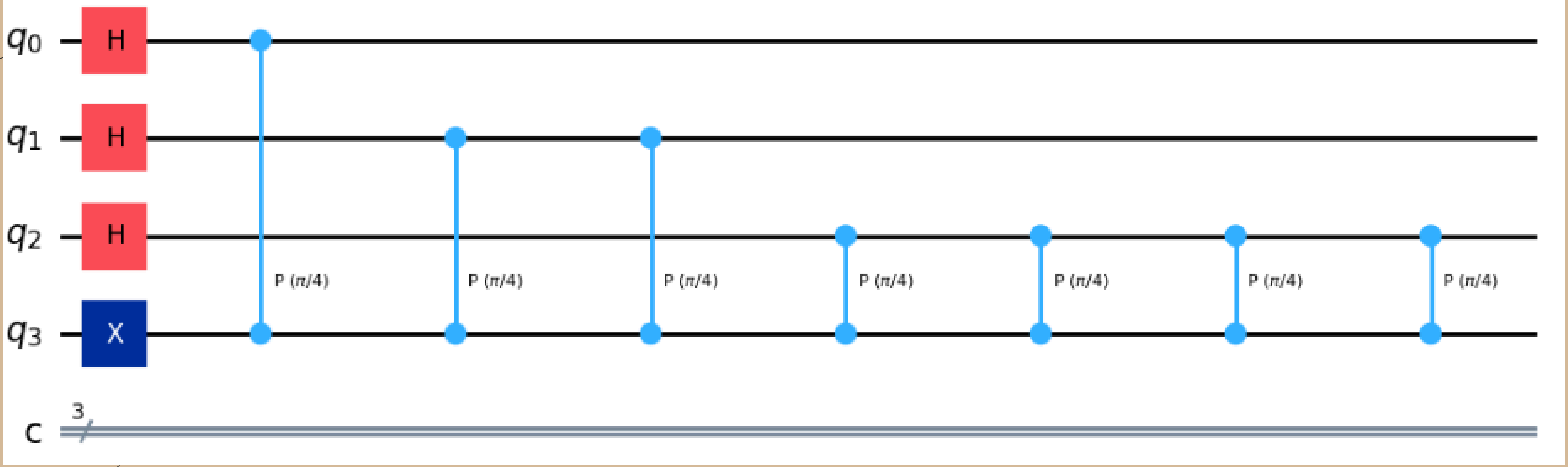
```
circuit.h(range(3))
```



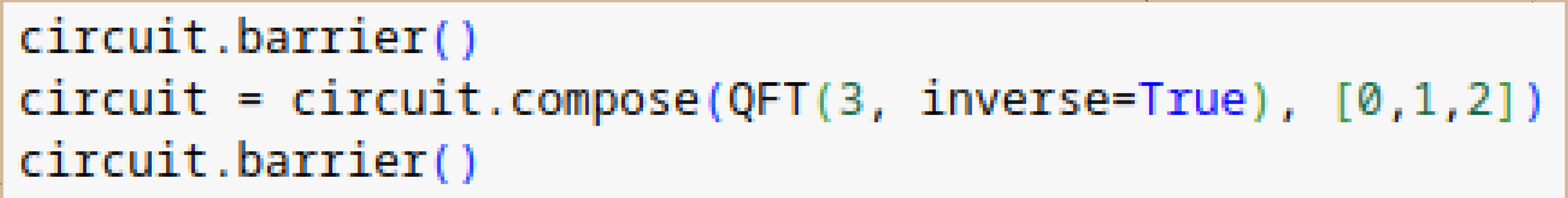
3- Appliquer les portes contrôlées-U



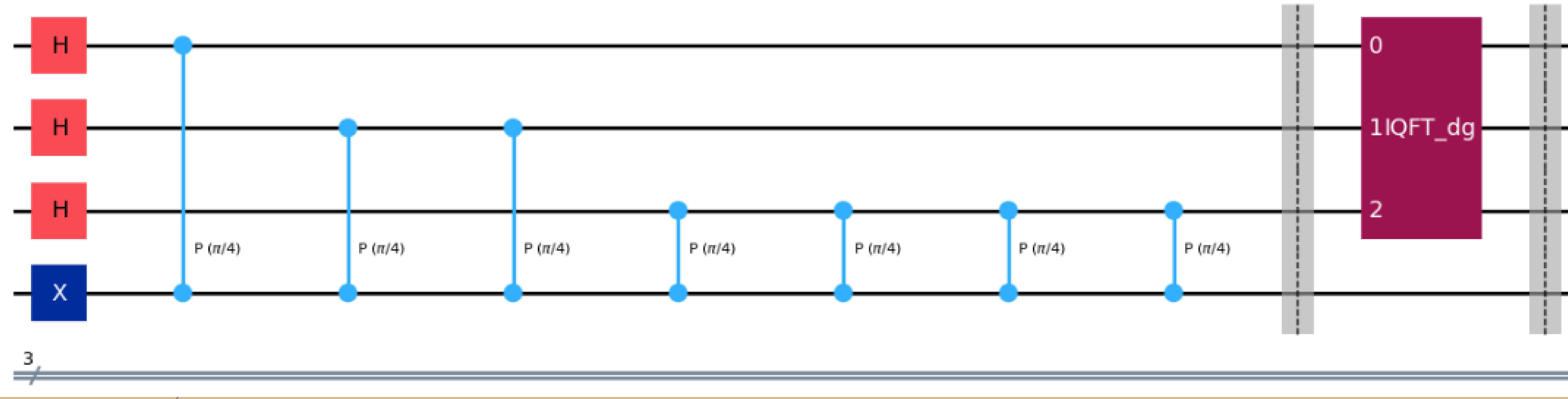
```
repetitions = 1
for counting_qubit in range(3):
    for i in range(repetitions):
        circuit.cp(math.pi/4, counting_qubit, 3); # controlled-T
    repetitions *= 2
```



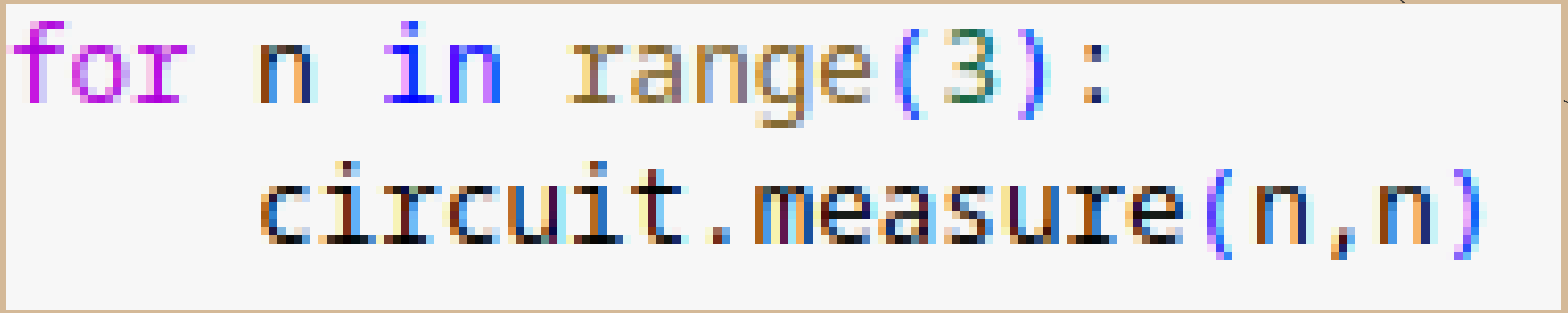
4– Appliquer la Transformée de Fourier Quantique Inverse (QFT Inverse)



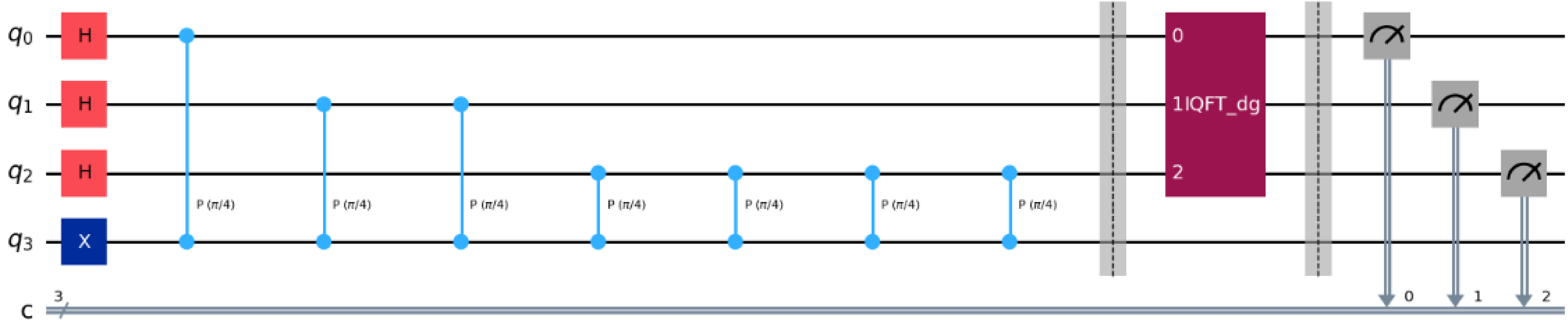
```
circuit.barrier()
circuit = circuit.compose(QFT(3, inverse=True), [0,1,2])
circuit.barrier()
```



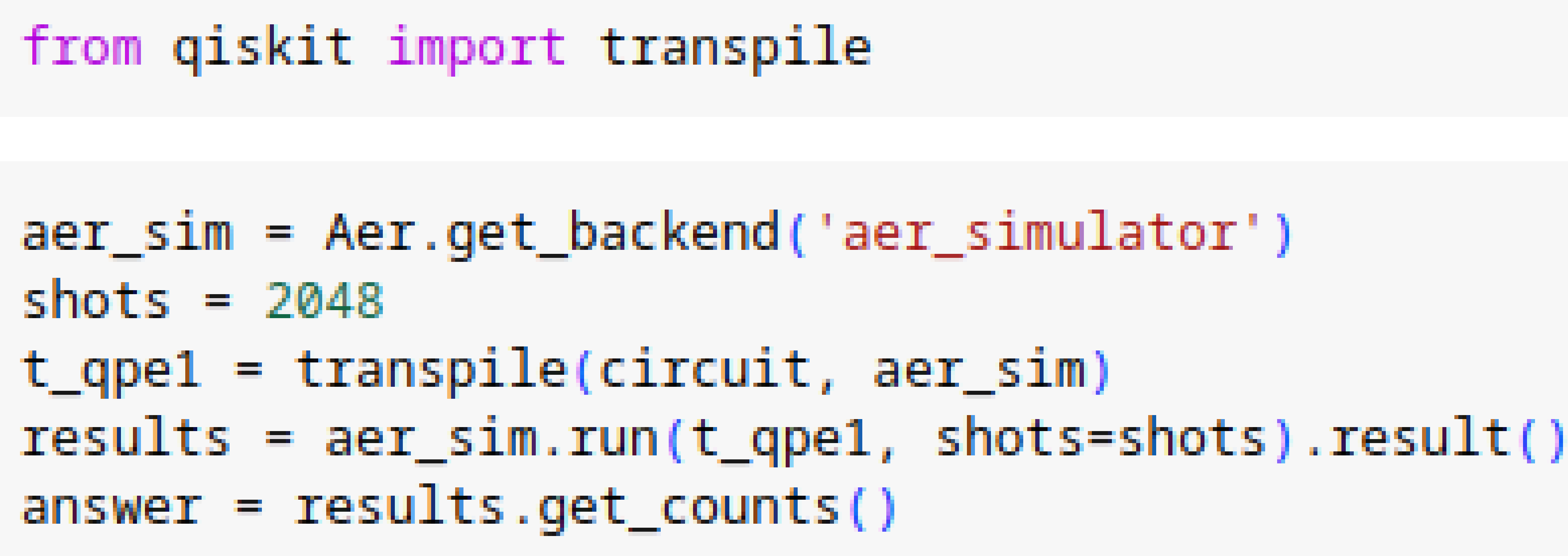
5– Mesurer le registre de valeur propre



```
for n in range(3):
    circuit.measure(n,n)
```

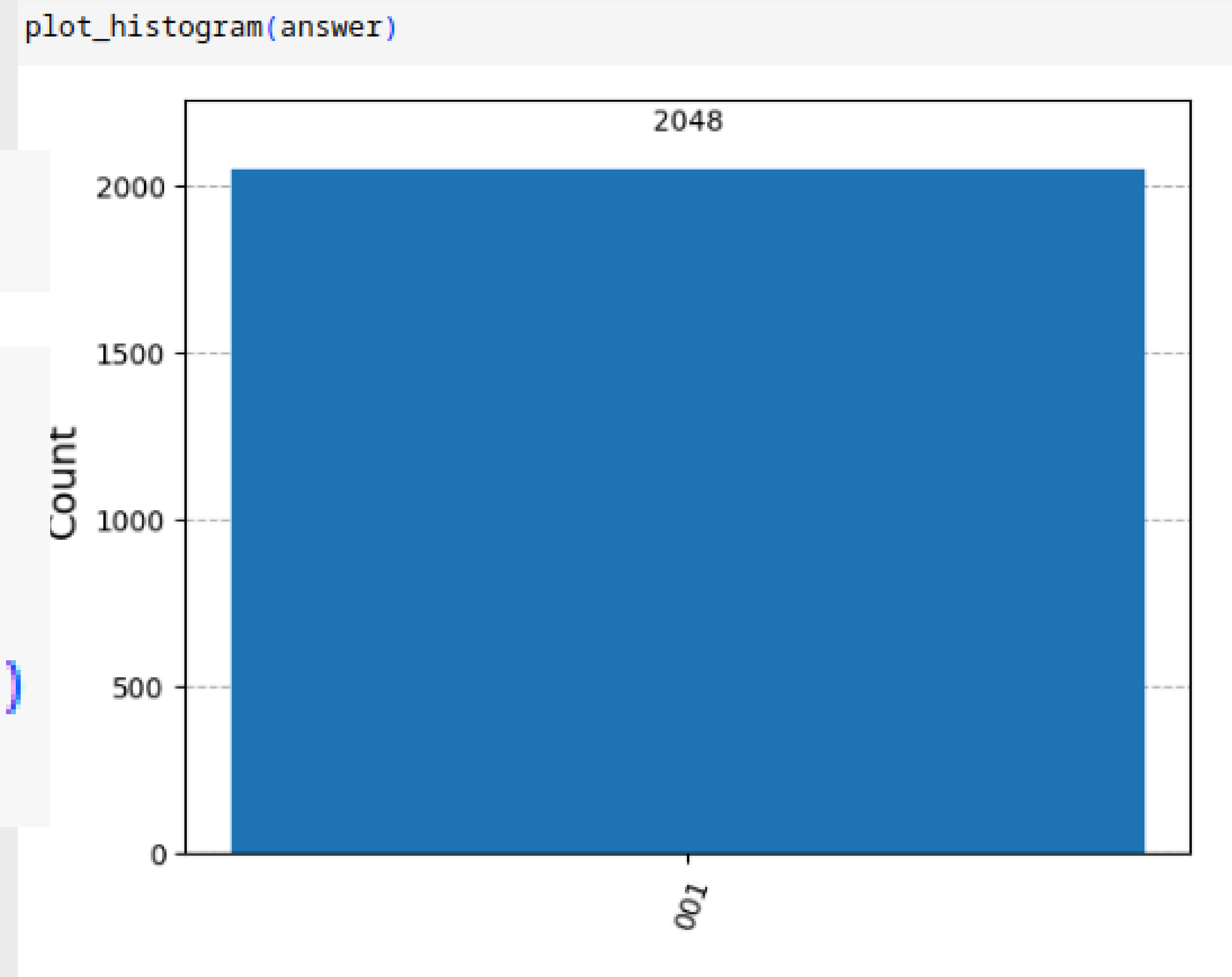


Simulation



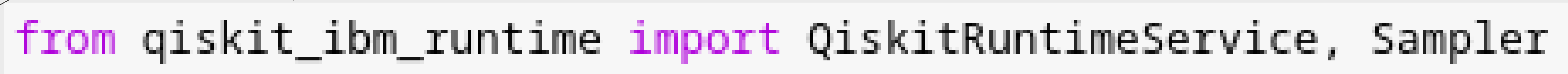
```python
from qiskit import transpile
```

```python
aer_sim = Aer.get_backend('aer_simulator')
shots = 2048
t_qpe1 = transpile(circuit, aer_sim)
results = aer_sim.run(t_qpe1, shots=shots).result()
answer = results.get_counts()
```

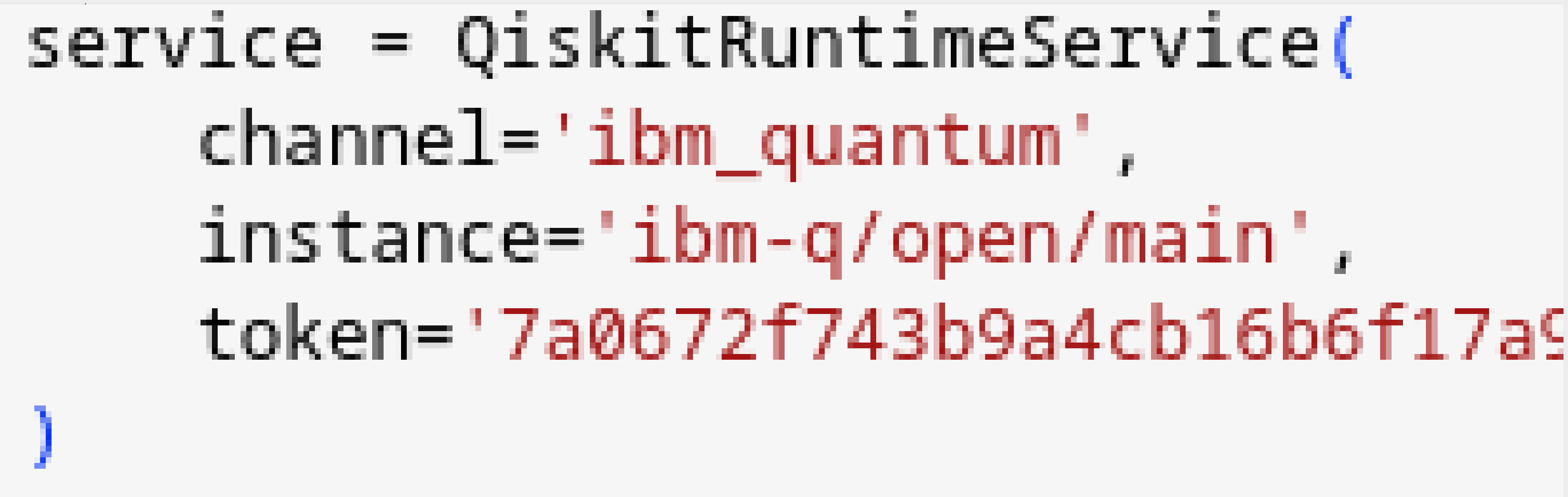


```python
plot_histogram(answer)
```

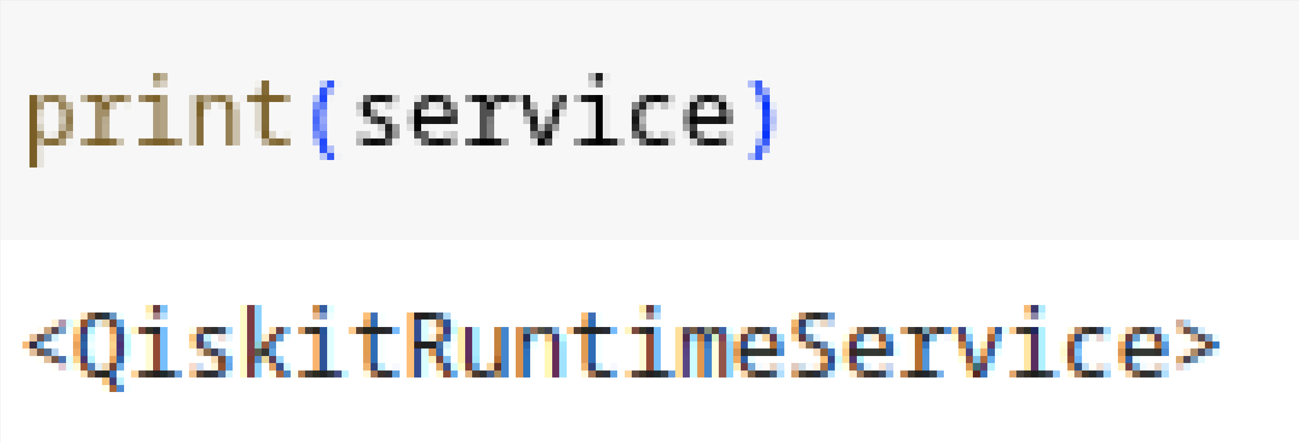
Simulation à un ordinateur quantique réel



```
from qiskit_ibm_runtime import QiskitRuntimeService, Sampler
```



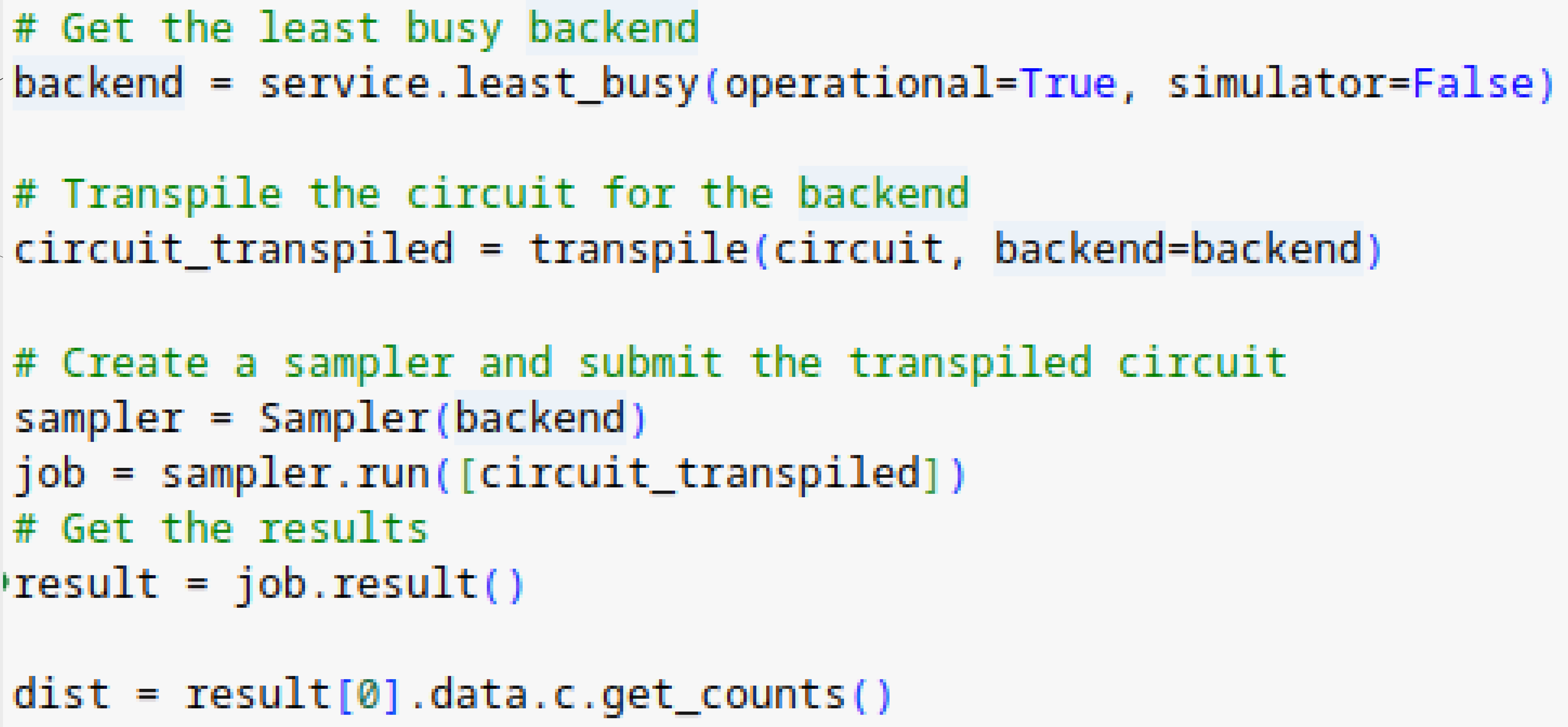
```
service = QiskitRuntimeService(
    channel='ibm_quantum',
    instance='ibm-q/open/main',
    token='7a0672f743b9a4cb16b6f17a9
)
```



```
print(service)
```

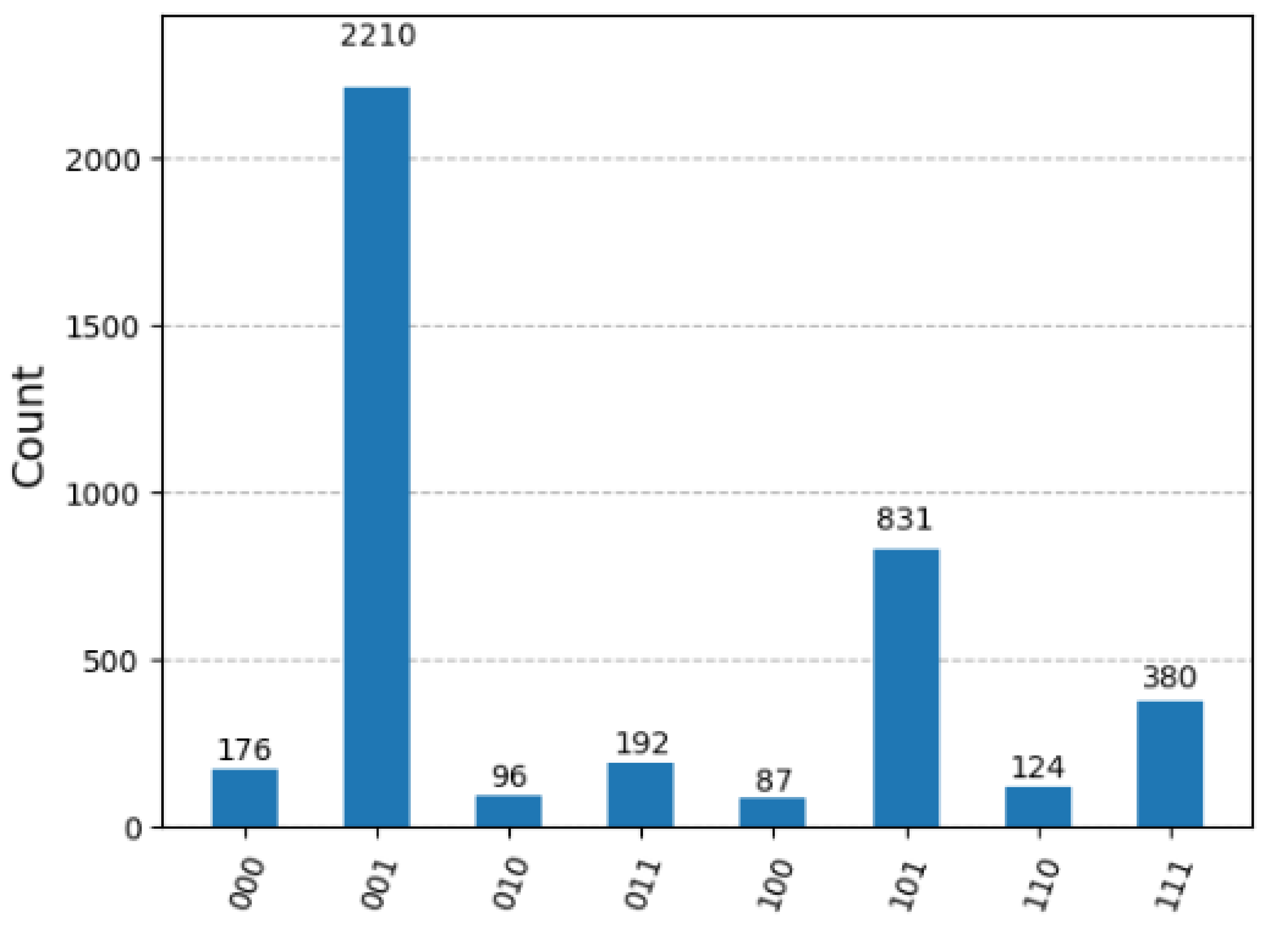
```
<QiskitRuntimeService>
```

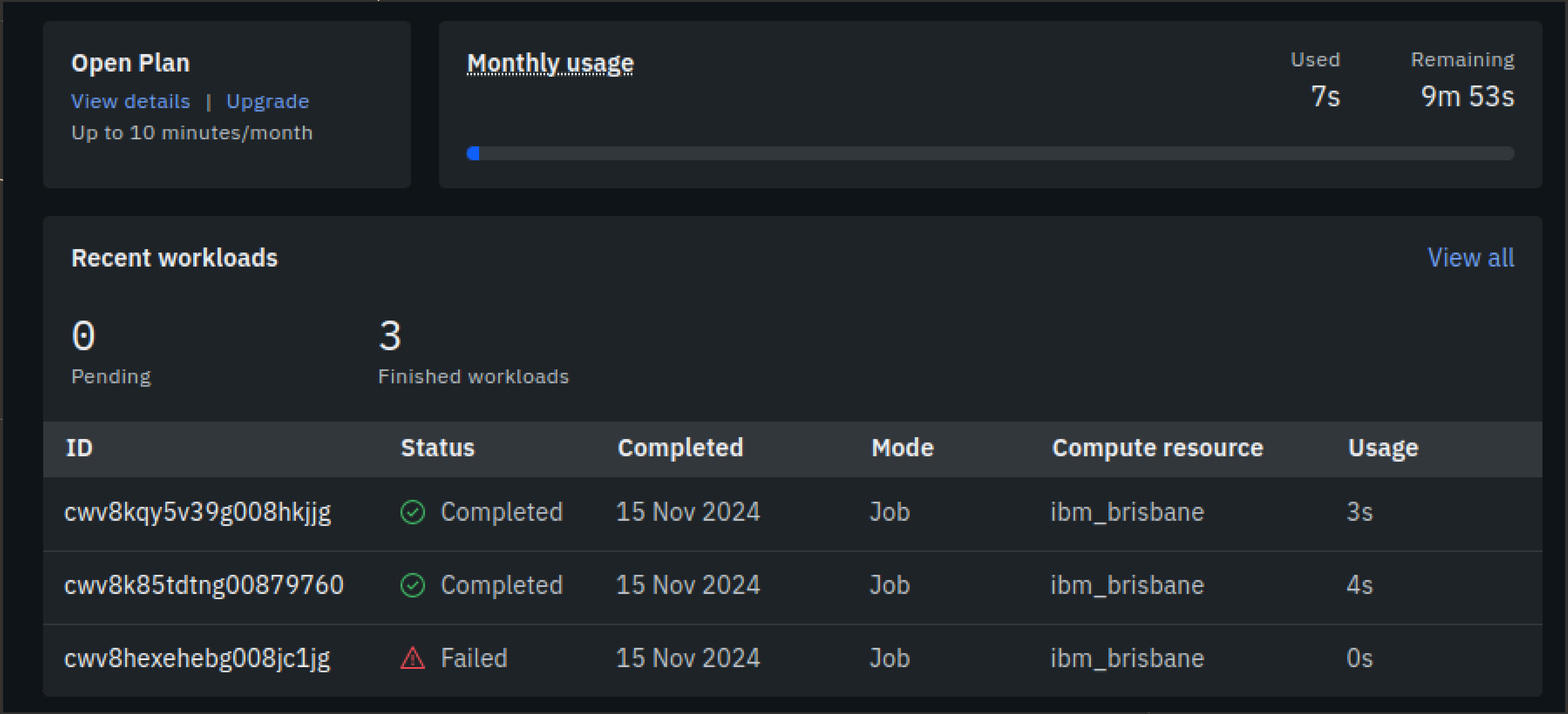
Simulation à un ordinateur quantique réel



```
# Get the least busy backend
backend = service.least_busy(operational=True, simulator=False)

# Transpile the circuit for the backend
circuit_transpiled = transpile(circuit, backend=backend)

# Create a sampler and submit the transpiled circuit
sampler = Sampler(backend)
job = sampler.run([circuit_transpiled])
# Get the results
result = job.result()

dist = result[0].data.c.get_counts()
```

Simulation à un ordinateur quantique réel





Open Plan

View details | Upgrade

Up to 10 minutes/month

Monthly usage

Used 7s

Remaining 9m 53s

Recent workloads

View all

0 Pending

3 Finished workloads

ID	Status	Completed	Mode	Compute resource	Usage
cwv8kqy5v39g008hkjjg	✓ Completed	15 Nov 2024	Job	ibm_brisbane	3s
cwv8k85tdtng00879760	✓ Completed	15 Nov 2024	Job	ibm_brisbane	4s
cwv8hexehebg008jc1jg	⚠ Failed	15 Nov 2024	Job	ibm_brisbane	0s

Merci!