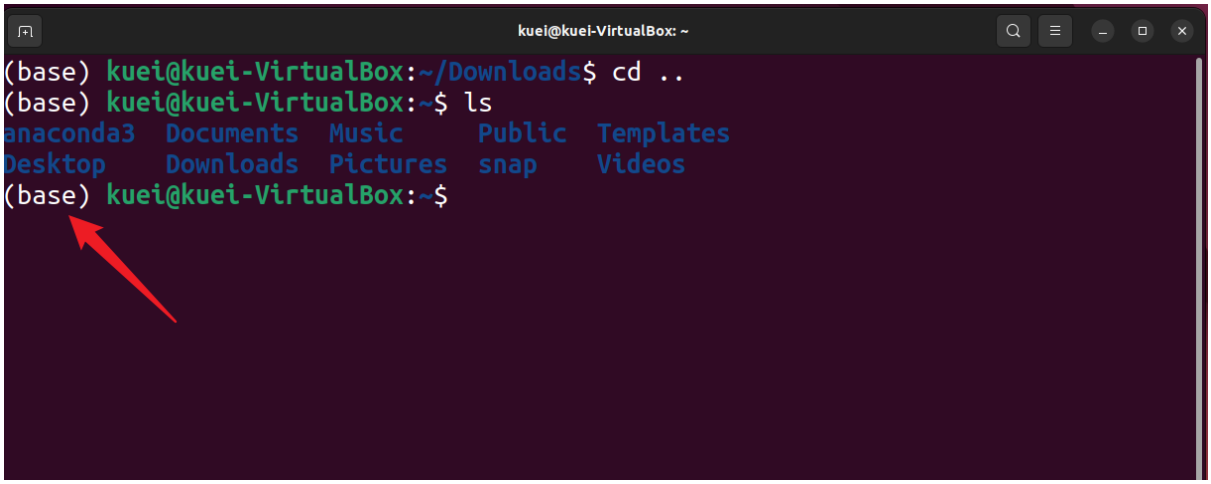
0. local 環境建置

jupyter notebook

- 安裝 Anaconda
- 使用 jupyter notebook 執行第一個 qiskit 程式

---

1. 建置 conda env




```
# 建置 env
conda create --name [myenv]
# 執行以上指令後
conda activate myQ
```

2. 安裝相關套件

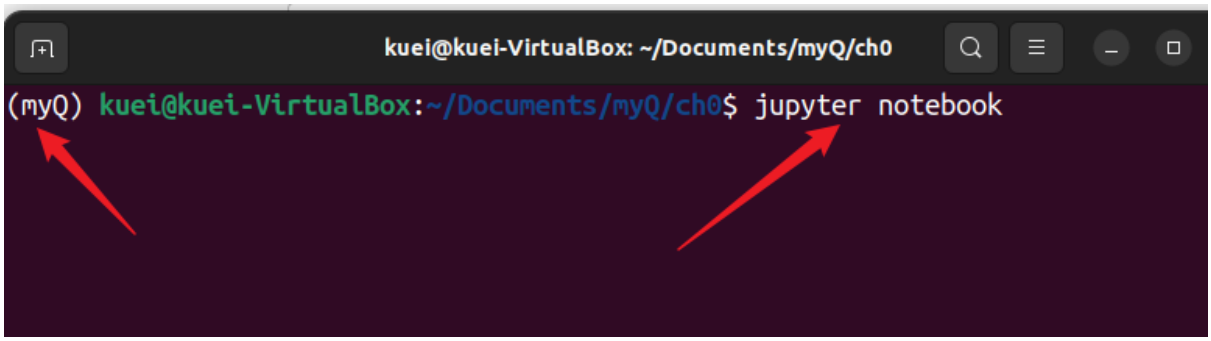
會使用到以下套件

- jupyter notbook → coding 環境
- qiskit → quantum lib
- matplotlib → 用於繪圖

```
# 在環境下安裝
conda install jupyter notebook
# 安裝 qiskit 套件
python -m pip install qiskit matplotlib
```

3.檢視環境安裝是否正常

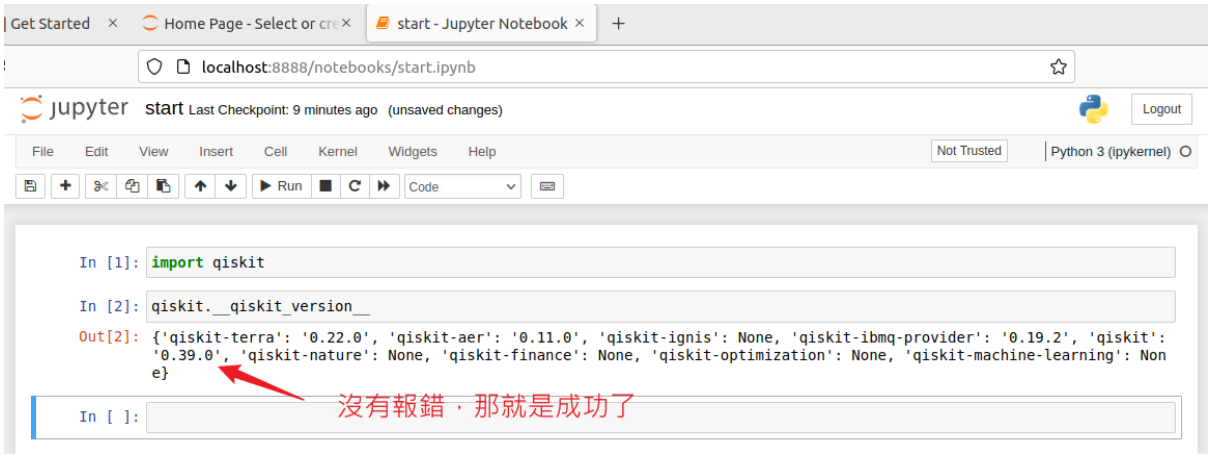
```
# 在環境下指令
(env) $ jupyter notebook
```



成功會有類似以下畫面

新增 Python3 後進入撰寫程式環境，並在環境中 run 以下指令，檢視 qiskit 是否安裝成功

```
import qiskit
qiskit.__qiskit_version__
```



4. 第一個程式

```
from qiskit import QuantumCircuit, assemble, Aer
from qiskit.visualization import plot_histogram, plot_bloch_vector
from math import sqrt, pi
```

```
qc = QuantumCircuit(1) # Create a quantum circuit with one qubit
qc = QuantumCircuit(1)  # Create a quantum circuit with one qubit
initial_state = [0,1]   # Define initial_state as |1>
```

```
qc.initialize(initial_state, 0) # Apply initialisation operation to the 0th qubit
qc.draw()  # Let's view our circuit
```

In [4]:
```
qc = QuantumCircuit(1) # Create a quantum circuit with one qubit
qc = QuantumCircuit(1)  # Create a quantum circuit with one qubit
initial_state = [0,1]   # Define initial_state as |1>
qc.initialize(initial_state, 0) # Apply initialisation operation to the 0th qubit
qc.draw()  # Let's view our circuit
```

Out[4]:

```
q: ┤ Initialize(0,1) ├
```

```
qc = QuantumCircuit(1)  # Create a quantum circuit with one qubit
initial_state = [0,1]   # Define initial_state as |1>
qc.initialize(initial_state, 0) # Apply initialisation operation to the 0th qubit
qc.save_statevector()   # Tell simulator to save statevector
qobj = assemble(qc)     # Create a Qobj from the circuit for the simulator to run
result = sim.run(qobj).result() # Do the simulation and return the result

out_state = result.get_statevector()
print(out_state) # Display the output state vector
```

In [6]:
```
out_state = result.get_statevector()
print(out_state) # Display the output state vector
```

```
Statevector([0.+0.j, 1.+0.j],
            dims=(2,))
```

```
qc.measure_all()
qc.draw()
```

In [7]:
```
qc.measure_all()
qc.draw()
```

Out[7]: