

# Python/Qiskit 入門

高中量子科技課程

2022 (C) 黃敦紀

# 課程安排

- Python 入門
- Qiskit 實作

# Python 入門

- 循序結構：
  - 函式 function
  - 物件 object
- 選擇結構：`if`
- 重複結構：迴圈 loop



# IBM Quantum



Graphically build circuits with  
**IBM Quantum Composer**

Launch Composer

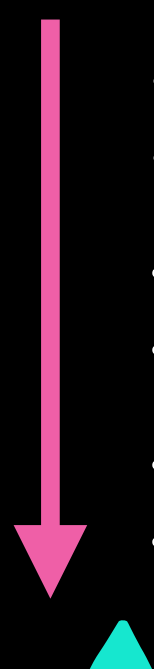


Develop quantum experiments in  
**IBM Quantum Lab**

Launch Lab

# 循序結構

- 程式一行一行循序執行

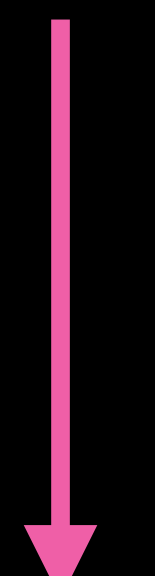


```
print ("Hello World")  
print ("123")  
print (34+25)
```

按照順序執行  
動作

程式

(請注意句首不可隨意空白)


$$\begin{aligned}x^2 - 5x + 6 &= 0 \\(x - 2)(x - 3) &= 0 \\x - 2 = 0 \text{ or } x - 3 &= 0 \\x = 2 \text{ or } x = 3\end{aligned}$$

等價、同義

數學演算

# 循序結構

- 練習：用 Python 程式輸出

```
I enjoy the class.
```

```
456
```

```
121212
```

# 函式 function

```
def fun():  
    print("hi")  
    return
```

```
fun()  
fun()  
fun()
```

... 定義 define

... 呼叫 (執行) call

... 呼叫 (執行) call

... 呼叫 (執行) call

```
def fun():  
    print ("hi")  
    return
```

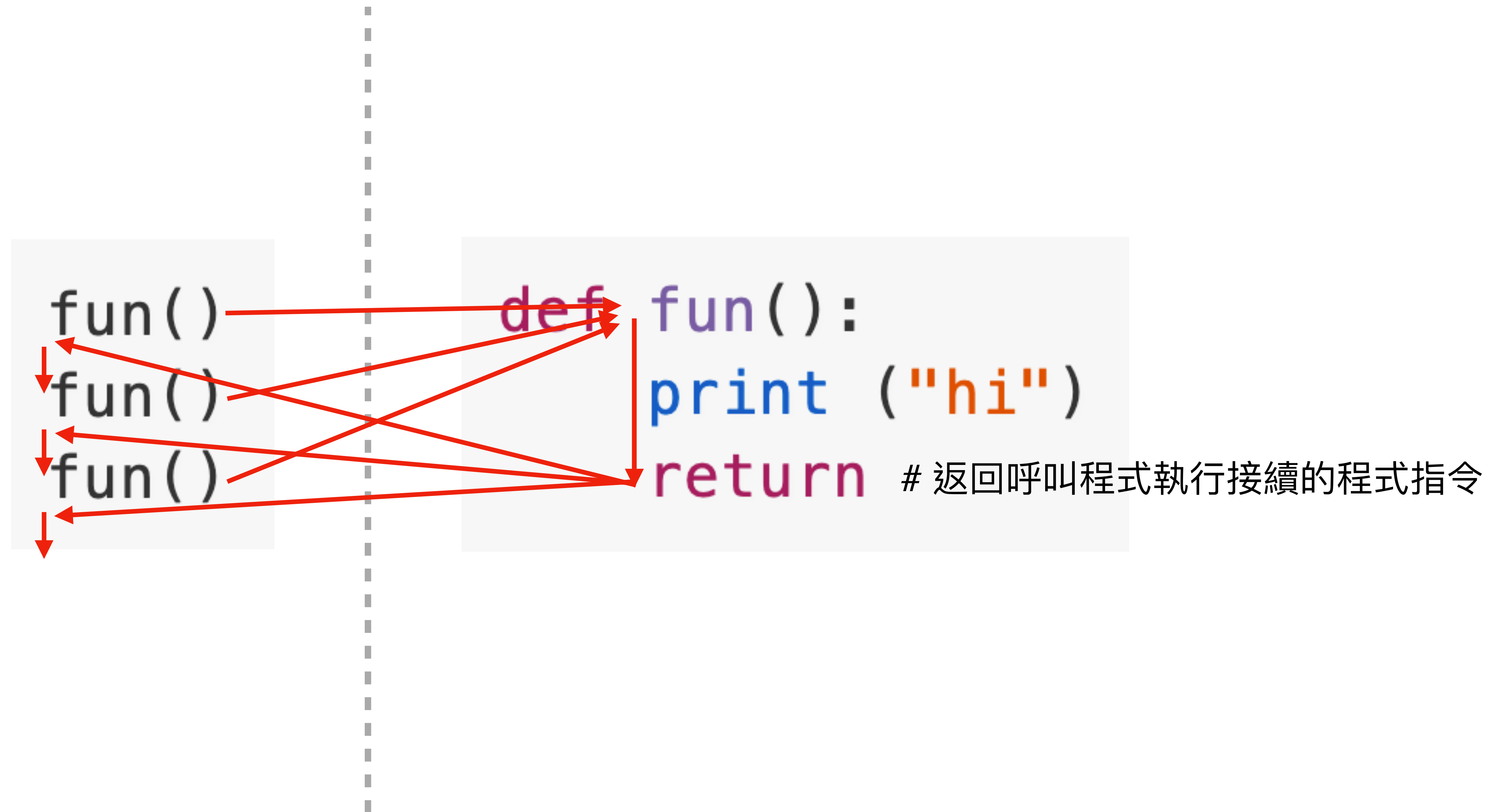
---

```
fun()  
fun()  
fun()
```



```
fun()  
fun()  
fun()
```

```
def fun():  
    print ("hi")  
    return
```



- 參數
- 回傳值

提供的功能： 量子計算 機器學習 數學工具 ...

modules/套件：

qiskit

tensorflow

numpy

...

語言/環境：

Python

開始建構一個 quantum circuit ...

(circuit 名稱可以自己取)

指定參數 (1 個 qubit, 1 個 classical bit)

qc = QuantumCircuit (1, 1)

指定為回傳值

qubit 0 接上 hadamard gate ...

動作      qubit 序號從 0 開始

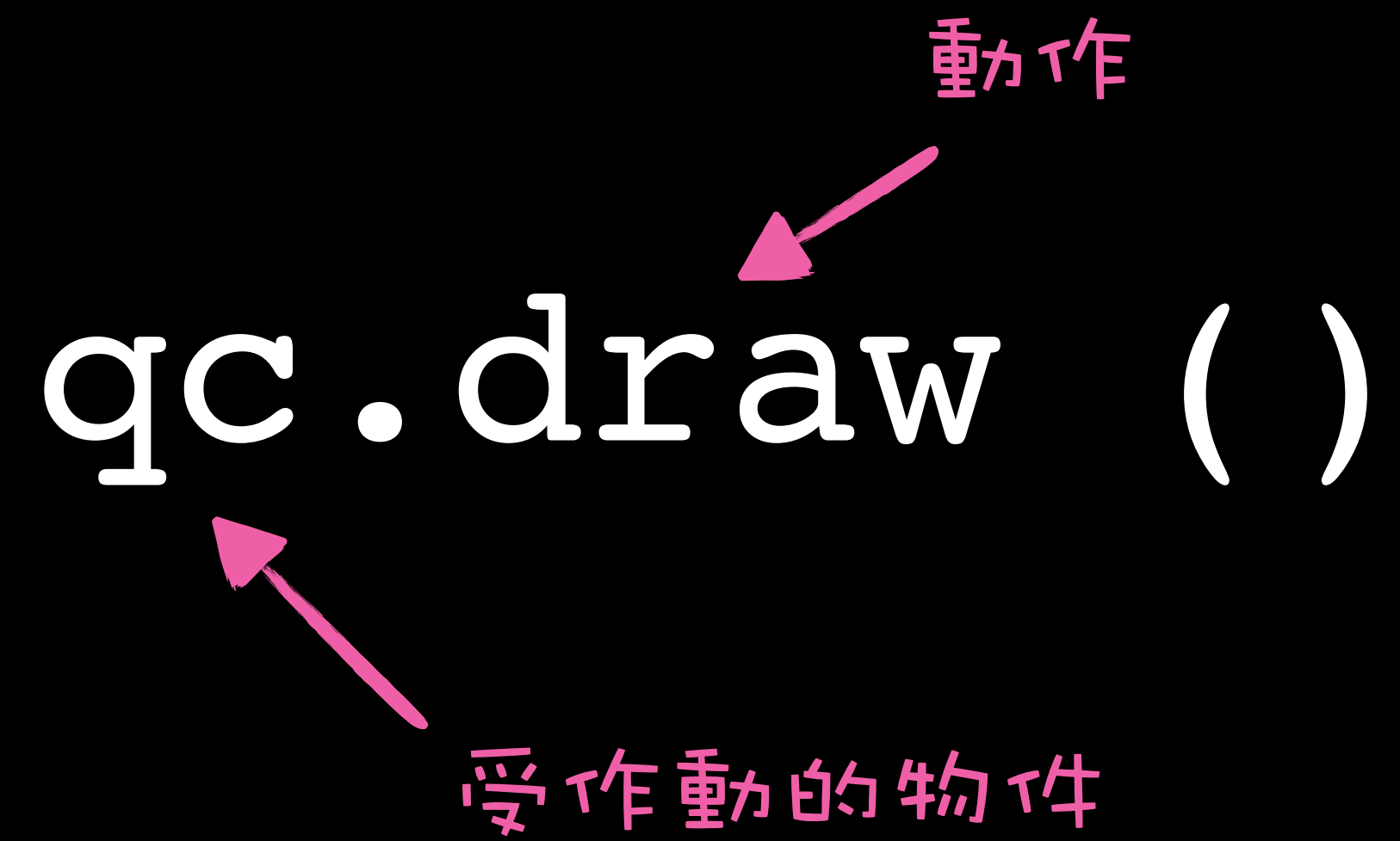
qc.h (0)

受作動的物件

The diagram shows the text 'qc.h (0)' in white. Three pink arrows point to its components: one from the text '動作' (Action) to the 'h', one from the text 'qubit 序號從 0 開始' (Qubit sequence starts from 0) to the '(0)', and one from the text '受作動的物件' (Object being acted upon) to the 'qc'.

先將目前的 circuit 畫出來看看 ...

qc.draw ()



動作

受作動的物件

# 接下來你可以做 2 (3) 件事

1. 用模擬器後端模擬執行 (量測)
2. 用實機執行 (量測)
3. 看狀態向量 (理論，狀態不塌縮)



在要量測的 qubit 上接上 measurements ...

`qc.measure` (0, 0)

動作

要量測的 qubit 序號

受作動的物件

指定對應的 classical bit 序號

在要量測的 qubit 上接上 measurements ...  
(如果 2 個 bits)

```
qc.measure (range (2), range (2))
```

# 1. 用模擬器後端模擬執行

```
sim = QasmSimulator()  
comp = transpile (qc, sim)  
sres = sim.run (comp, 1024).result().get_counts (qc)
```

看結果 ...

```
print (sres) # 文字  
plot_histogram(sres) # 統計 (直方) 圖
```

## 2. 用實機執行

```
from qiskit import execute
from qiskit.tools.monitor import job_monitor
```

```
backend = IBMQ.get_provider('ibm-q').get_backend('ibmq_lima')
job = execute(qc, backend = backend, shots = 1024)
job_monitor(job, interval = 5)
rres = job.result().get_counts(qc)
```

(找 5/7 qubits, Online, pending jobs 的)

看結果 ...

```
print(rres) # 文字
plot_histogram(rres) # 統計 (直方) 圖
```

## 2. 用實機執行 (續)

實機雜訊圖 ... (也可以直接在網頁上看)

```
plot_error_map(backend)
```

挑戰題：

如何用實機執行得到更接近理論的結果？

# 重複結構 (迴圈 loop)

```
print (1)  
print (2)  
print (3)
```

```
for i in range (3):  
    print (i)
```

(縮排：空四格)

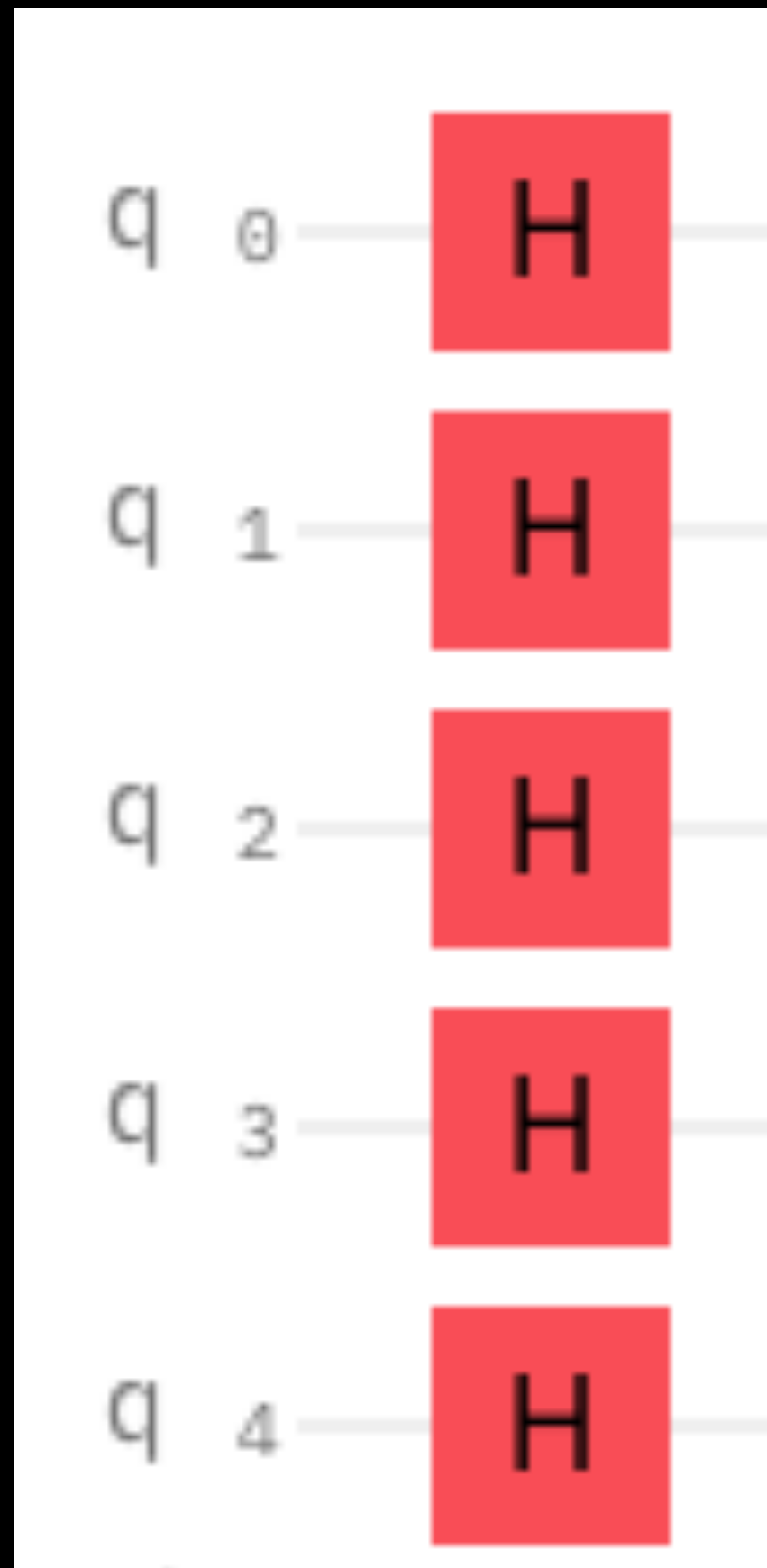
# 重複結構

- 練習：用 Python 迴圈輸出

```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

# 重複結構

- 練習：用迴圈建構以下 quantum circuit





# 3. 看狀態向量

```
from qiskit.quantum_info import Statevector
```

```
Statevector.from_instruction(qc).draw("latex", prefix=" ")
```

```
plot_bloch_multivector(Statevector.from_instruction(qc))
```

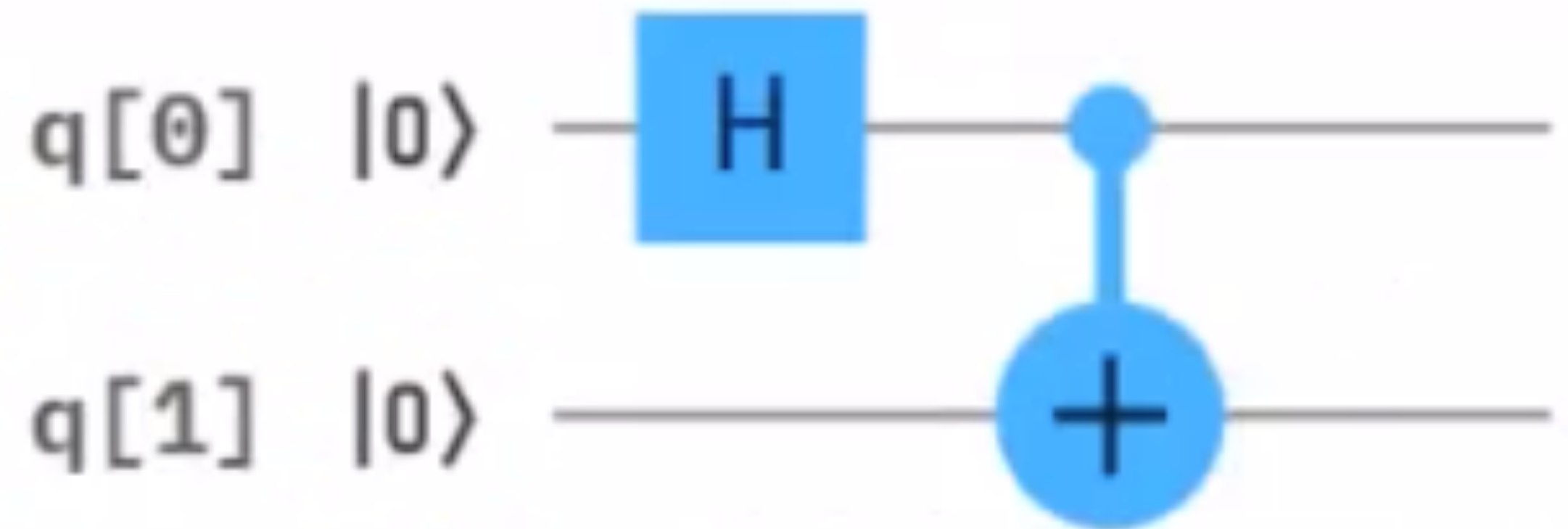
## Bell state

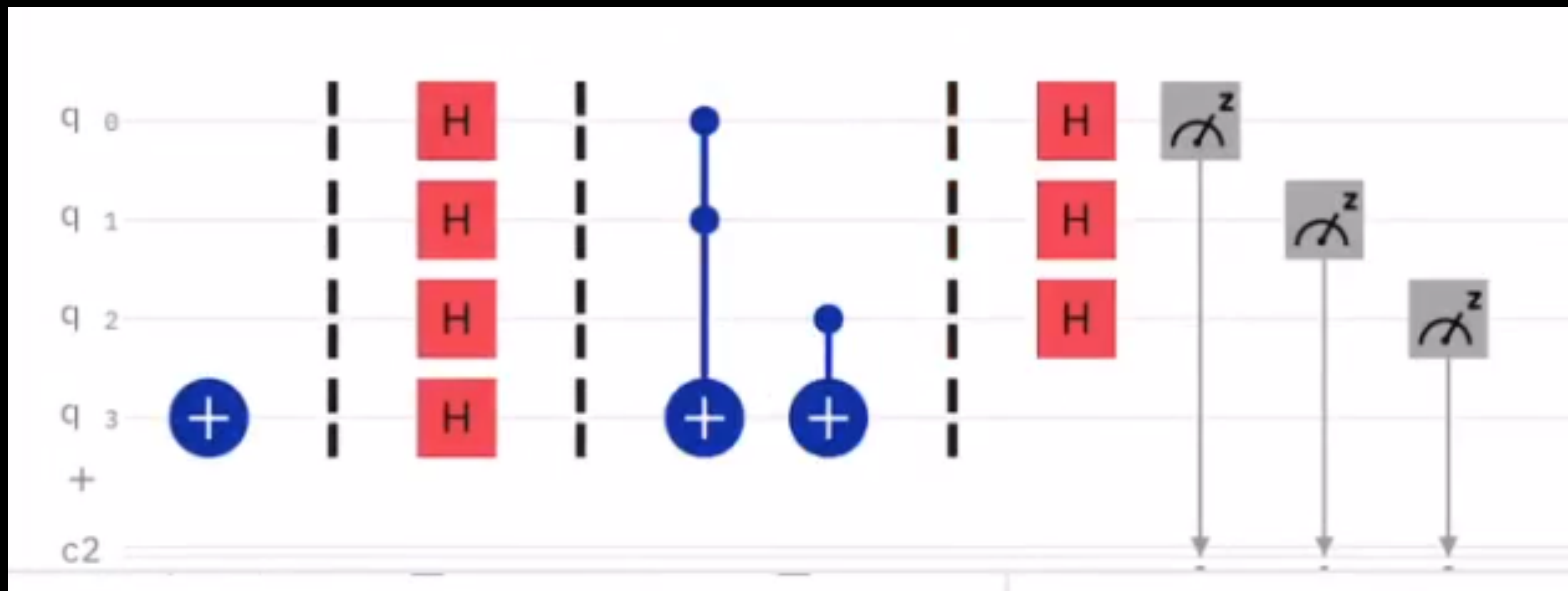
$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$$\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$

$$\frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$$

$$\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$





# Qiskit Applications

IBM Quantum Challenge Africa 2021

IBM Quantum Challenge Fall 2021