Q13

QX13-ShorQuantumQuick.ipynb
File

Necestio que me expliques detalladamente cada celda del codigo

A continuación tienes el desglose detallado de cada una de las celdas de tu notebook **`QX13-ShorQuantumQuick.ipynb`**. He numerado las celdas según el orden en que aparecen:

Celda 1

```
python

from qiskit import *
from qiskit.algorithms import Shor
```

Explicación detallada

1. **`from qiskit import *`**
   - Importa todo el contenido del paquete principal de Qiskit. Incluye, entre otros, los módulos para construir y ejecutar circuitos cuánticos, widgets, utilidades y backends.
   - **Nota de estilo**: en producción es mejor importar sólo lo que se necesita (**`import qiskit`** o **`from qiskit import QuantumCircuit, Aer`**) para evitar sobrecarga de nombres.
2. **`from qiskit.algorithms import Shor`**
   - Importa la clase **`Shor`** que implementa el algoritmo de Shor (factoreo cuántico) dentro del submódulo **`algorithms`**.
   - A partir de Qiskit Terra 0.24+, el algoritmo de Shor se encuentra en **`qiskit.algorithms`**. Esta clase nos permite factorizar enteros clásicos usando un backend cuántico (simulado o real).

Celda 2

```
python

qinstance = Aer.get_backend('qasm_simulator')
shor = Shor(quantum_instance=qinstance)

result = shor.factor(N=15, a=7)
print('Factors:', result.factors)
```

Explicación detallada

1. `qinstance = Aer.get_backend('qasm_simulator')`
   - `Aer` es el simulador de Qiskit para circuitos de puerta (gate-model).
   - El método `get_backend('qasm_simulator')` devuelve un backend que emula la ejecución de circuitos cuánticos midiendo el registro de bits cuánticos (QASM).
   - Con esto podemos probar y depurar algoritmos cuánticos sin usar hardware real.
2. `shor = Shor(quantum_instance=qinstance)`
   - Crea una instancia del algoritmo de Shor, pasándole como parámetro el backend sobre el que se ejecutarán los circuitos cuánticos.
   - Internamente, `Shor` montará uno o varios circuitos para realizar la estimación de fase cuántica y luego procesará los resultados para hallar factores.
3. `result = shor.factor(N=15, a=7)`
   - Llama al método `factor` de la instancia de Shor para factorizar el número **N = 15**.
   - El parámetro `a = 7` es un entero **aleatorio** coprimo con 15; en general el algoritmo de Shor elige un valor `a` tal que `1 < a < N` y `gcd(a, N) = 1`. Aquí se fija manualmente en 7.
   - El método devuelve un objeto (`result`) con varios atributos, entre ellos `factors`, que contiene la lista de factores hallados (en este caso `[3, 5]`).
4. `print('Factors:', result.factors)`
   - Muestra en consola los factores encontrados. Salida esperada:

     ```lua
     lua

     Factors: [[3, 5]]
     ```

   - Nota: `result.factors` es una lista de posibles pares de factores.

Celda 3

```python
python

help(shor)
```

Explicación detallada

- Invoca la función integrada de Python `help` sobre el objeto `shor`.
- Esto despliega en pantalla la documentación (docstring) de la clase `Shor`, incluyendo:
  - Descripción general del algoritmo.
  - Parámetros del constructor (`quantum_instance`, opciones de medición, número de repeticiones, etc.).
  - Métodos disponibles (`factor`, `estimate_period`, etc.) y sus firmas.
- Útil para explorar las opciones que ofrece Qiskit sin salir del notebook.

Celda 4

```
python

# (vacía)
```

Explicación detallada

- Es una celda en blanco (sin contenido).
- Puede haber quedado como marcador o para separar secciones.
- No tiene efecto al ejecutar el notebook.

Resumen del flujo

1. **Importación** de Qiskit y del algoritmo de Shor.
2. **Configuración** del simulador cuántico (`qasm_simulator`).
3. **Instanciación** del algoritmo de Shor con el backend elegido.
4. **Ejecución** de la factorización de `N = 15` usando el valor `a = 7`.
5. **Visualización** de los factores obtenidos.
6. **Consulta** de la documentación interna de la clase `Shor` con `help`.

Con ello tienes un pipeline completo para probar el algoritmo de Shor en simulación: desde la importación hasta la inspección de sus opciones. Si quieres profundizar en alguna parte (por ejemplo, en cómo se construyen internamente los circuitos de estimación de fase o en el tratamiento de los resultados), ¡avísame y lo vemos!