```python
from qiskit.quantum_info import Statevector, Operator
from numpy import sqrt
```

[1] ✓ 1.4s Python

Tensor products

The State vector class has a tensor method which returns the tensor product of itself and another State vector .

For example, below we create two state vectors representing $|0\rangle$ and $|1\rangle$, and use the tensor method to create a new vector, $|0\rangle \otimes |1\rangle$.

```python
zero, one = Statevector.from_label("0"), Statevector.from_label("1")
zero.tensor(one).draw("latex")
```

[2] ✓ 1.3s Python

...

$$|01\rangle$$

In another example below, we create state vectors representing the $|+\rangle$ and $\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$ states, and combine them to create a new state vector. We'll assign this new vector to the variable `psi` .

```python
plus = Statevector.from_label("+")
i_state = Statevector([1 / sqrt(2), 1j / sqrt(2)])
psi = plus.tensor(i_state)

psi.draw("latex")
```

[3] ✓ 0.1s Python

...

$$\frac{1}{2}|00\rangle + \frac{i}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{i}{2}|11\rangle$$

The Operator class also has a tensor method. In the example below, we create the *X* and I gates and display their tensor product.

```python
X = Operator([[0, 1], [1, 0]])
I = Operator([[1, 0], [0, 1]])

X.tensor(I)
```

[4] ✓ 0.0s Python

```
Operator([[0.+0.j, 0.+0.j, 1.+0.j, 0.+0.j],
          [0.+0.j, 0.+0.j, 0.+0.j, 1.+0.j],
          [1.+0.j, 0.+0.j, 0.+0.j, 0.+0.j],
          [0.+0.j, 1.+0.j, 0.+0.j, 0.+0.j]],
         input_dims=(2, 2), output_dims=(2, 2))
```

We can then treat these compound states and operations as we did single systems in the previous lesson. For example, in the cell below we calculate
$(I \otimes X)|\psi\rangle$

For the state psi we defined above. (The ^ operator tensors matrices together.)

```python
psi.evolve(I ^ X).draw("latex")
```

[5] ✓ 0.0s Python

$$\frac{i}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{i}{2}|10\rangle + \frac{1}{2}|11\rangle$$

Below, we create a CX operator and calculate CX|ψ⟩.

```
CX = Operator(
    [
        [1, 0, 0, 0],
        [0, 1, 0, 0],
        [0, 0, 0, 1],
        [0, 0, 1, 0],
    ]
)

psi.evolve(CX).draw("latex")
```

[6] ✓ 0.0s Python

...

$$\frac{1}{2}|00\rangle + \frac{i}{2}|01\rangle + \frac{i}{2}|10\rangle + \frac{1}{2}|11\rangle$$

In the previous Lab , we used the measure method to simulate a measurement of the quantum state vector. This method returns two items: the simulated measurement result, and the new State vector given this measurement.

By default, measure measures all qubits in the state vector, but we can provide a list of integers to only measure the qubits at those indices. To demonstrate, the cell below creates the state

$$W = \frac{1}{\sqrt{3}}(|001\rangle + |010\rangle + |100\rangle).$$

```
W = Statevector([0, 1, 1, 0, 1, 0, 0, 0] / sqrt(3))
W.draw("latex")
```

[7] ✓ 0.0s Python

...

$$\frac{\sqrt{3}}{3}|001\rangle + \frac{\sqrt{3}}{3}|010\rangle + \frac{\sqrt{3}}{3}|100\rangle$$

The cell below simulates a measurement on the rightmost qubit (which has index 0). The other two qubits are not measured.

```python
result, new_sv = W.measure([0])  # measure qubit 0
print(f"Measured: {result}\nState after measurement:")
new_sv.draw("latex")
```

[8] ✓ 0.0s Python

```
Measured: 1
State after measurement:
```

$$|001\rangle$$

Try running the cell a few times to see different results. Notice that measuring a `1` means that we know both the other qubits are $|0\rangle$, but measuring a `0` means the remaining two qubits are in the state $\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$.