Graded Notebook 1: Welcome and Hello World

- Difficulty: 1/5
- Estimated Time: 30 minutes

Welcome to the Qiskit Fall Fest!

Hi there and welcome to the **Qiskit Fall Fest**!

Whether you're a total beginner or a PhD candidate in quantum physics, there is something here for you. Thanks for joining us. 😀

As part of the Qiskit Fall Fest, IBM Quantum has created a series of notebooks for you to work through, which all include coding challenges and Qiskit tutorials. Each notebook has a difficulty and an estimated time, which you can find at the top.

The Qiskit Fall Fest is a massive event, featuring thousands of students worldwide who are all learning about quantum computing and Qiskit. Just by being here, you're helping to make history. Your participation is helping to shape what the future of the industry will look like. Congratulations and welcome!

Each of these notebooks builds upon the previous learning. This first one is meant for anyone to complete, even beginners, but later notebooks are more difficult. Most participants will need to do some outside research or use a bit of trial-and-error to finish the code challenges presented in the more advanced notebooks. Don't give up! We know you can do it.

In each notebook, you will find links to documentation, tutorials, and other helpful resources you might need to solve that particular problem. You can also find most of these resources on IBM's new home for quantum education: IBM Quantum Learning.

Table of Contents

Qiskit

Let's start with the basics. You are participating in the **Qiskit Fall Fest**. So, what is Qiskit?

The Qiskit SDK is the world's most popular software stack for quantum computing, with over 2,000 forks, over 8,000 contributions, and over 3 trillion circuits run.

Qiskit first emerged in 2017 as an open-source software development kit (SDK), but since then, it has evolved into a comprehensive software stack delivering the world's most performant software for building, optimizing, orchestrating, and executing quantum workloads. Today, the Qiskit SDK is just one component in this expanded software stack, but it remains vitally important. Its architecture, capabilities, and emphasis on the circuit model of quantum computation have all heavily informed the development of the rest of the Qiskit software stack.

The open-source Qiskit SDK provides tools for mapping problems to quantum circuits

and preparing for execution. Qiskit also provides services to run, optimize, and orchestrate quantum workloads.

It might be helpful to think of three separate buckets - the Qiskit SDK, the Qiskit Runtime Service, and the Qiskit Transpiler Service. Below is a diagram which shows the four steps involved in running a quantum circuit on real hardware (map, optimize, execute, and post-process). At each step, you can see a color-coded description of where each aspect of Qiskit plays a role.
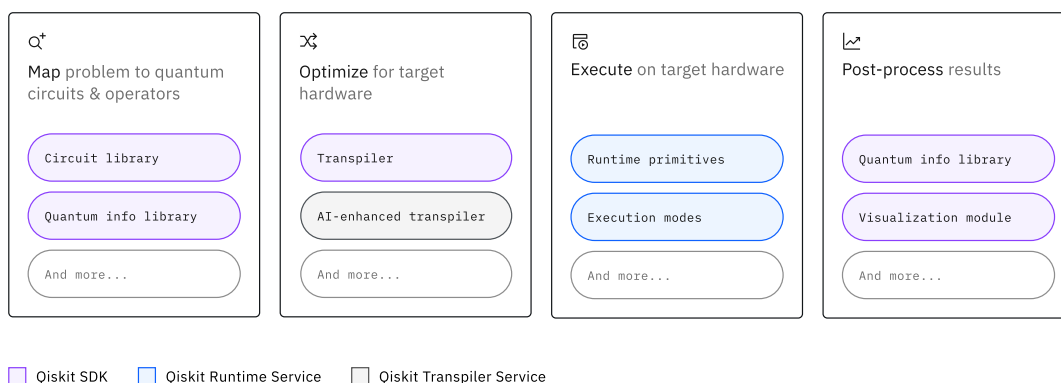
In addition, many open-source projects are part of the broader Qiskit ecosystem. These software tools are not part of Qiskit itself, but rather interface with Qiskit and can provide valuable additional functionality.

Simply put, Qiskit is software for doing real work on utility-scale quantum computers. The Qiskit Fall Fest is a celebration of all things Qiskit, and as you can see, there is quite a lot!

Notebook overview

In this notebook, we'll focus on teaching you how to manage the general flow of setting up, running, and evaluating the results of your quantum code. We'll also include some details about the proper way to configure each notebook you encounter.

Each notebook has **exercises** where you will be asked to fill in your own Qiskit code. There will be occasional hints, but it is up to you to talk with other people at your Qiskit Fall Fest event and solve each exercise. It's important to note that you should **run each code cell**, even if you didn't write any new code. This ensures that when you submit your answers later on, everything is up to date.

This notebook you are looking at is called a Jupyter Notebook, and it's really useful for writing text and code right next to each other. To complete this and the other notebooks, you'll need to know a little bit about the Python programming language.

Have you ever used Python before? If the answer is "yes" - feel free to skip forward to the Install Qiskit section. If you have never used Python before, you will **definitely** struggle with these notebooks. We recommend you go learn the basics of Python before continuing.

Install Qiskit

Where are you reading this sentence right now?

For about 95% of you, the answer is probably in a cloud-based tool (like Google Colab or qBraid), which means you're reading this in your browser. However, maybe you're seeing this on GitHub, or in the cozy comfort of your favorite IDE. For some of you wild folks out there, maybe you're deep in your command line.

No matter where this is reaching you, one thing remains true: you need to have Qiskit installed to successfully complete the each notebook in the Qiskit Fall Fest.

The next cell provides the install code for a cloud-based environment, like qBraid or Colab. If you're running locally you can watch this video to walk you through every step of installing Qiskit.

In [1]:
```
### INSTALL QISKIT inside your cloud-based environment

%pip install qiskit[visualization]
```

```
Requirement already satisfied: qiskit[visualization] in /home/jsoccerd/Cod
ing/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (1.2.4)
Requirement already satisfied: rustworkx>=0.15.0 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visuali
zation]) (0.15.1)
Requirement already satisfied: numpy<3,>=1.17 in /home/jsoccerd/Coding/Qis
kit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualizat
ion]) (2.1.2)
Requirement already satisfied: scipy>=1.5 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualizatio
n]) (1.14.1)
Requirement already satisfied: sympy>=1.3 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualizatio
n]) (1.13.3)
Requirement already satisfied: dill>=0.3 in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualization])
(0.3.9)
Requirement already satisfied: python-dateutil>=2.8.0 in /home/jsoccerd/Co
ding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[vi
sualization]) (2.9.0.post0)
Requirement already satisfied: stevedore>=3.0.0 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualiz
ation]) (5.3.0)
Requirement already satisfied: typing-extensions in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visuali
zation]) (4.12.2)
Requirement already satisfied: symengine<0.14,>=0.11 in /home/jsoccerd/Cod
ing/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[vis
ualization]) (0.11.0)
Requirement already satisfied: matplotlib>=3.3 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualiza
tion]) (3.9.2)
Requirement already satisfied: pydot in /home/jsoccerd/Coding/Qiskit-Fall-
Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualization])
(3.0.2)
Requirement already satisfied: Pillow>=4.2.1 in /home/jsoccerd/Coding/Qisk
it-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualizati
on]) (11.0.0)
Requirement already satisfied: pylatexenc>=1.4 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualiza
tion]) (2.10)
Requirement already satisfied: seaborn>=0.9.0 in /home/jsoccerd/Coding/Qis
kit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit[visualizat
ion]) (0.13.2)
Requirement already satisfied: contourpy>=1.0.1 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib>=
3.3->qiskit[visualization]) (1.3.0)
Requirement already satisfied: cycler>=0.10 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib>=3.3->qi
skit[visualization]) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib>=
3.3->qiskit[visualization]) (4.54.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib>=
3.3->qiskit[visualization]) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib>=3.3-
>qiskit[visualization]) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in /home/jsoccerd/Coding/Q
```

```
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib>=
3.3->qiskit[visualization]) (3.2.0)
Requirement already satisfied: six>=1.5 in /home/jsoccerd/Coding/Qiskit-Fa
ll-Fest-24/venv/lib/python3.12/site-packages (from python-dateutil>=2.8.0-
>qiskit[visualization]) (1.16.0)
Requirement already satisfied: pandas>=1.2 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from seaborn>=0.9.0->qis
kit[visualization]) (2.2.3)
Requirement already satisfied: pbr>=2.0.0 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from stevedore>=3.0.0->qis
kit[visualization]) (6.1.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from sympy>=1.3->
qiskit[visualization]) (1.3.0)
Requirement already satisfied: pytz>=2020.1 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from pandas>=1.2->seabor
n>=0.9.0->qiskit[visualization]) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /home/jsoccerd/Coding/Qis
kit-Fall-Fest-24/venv/lib/python3.12/site-packages (from pandas>=1.2->seab
orn>=0.9.0->qiskit[visualization]) (2024.2)
Note: you may need to restart the kernel to use updated packages.
```

Qiskit *should* now be installed and ready to go.

In the following cell, we've provided some very basic Qiskit code. This code will create a **quantum circuit**, apply a single **gate**, and then draw that circuit. Run the cell below to make sure your system is set up properly.

Hint: If those bolded words are foreign to you, we recommend taking a detour to learn about the basics of quantum circuits here.

In [2]:
```python
from qiskit import QuantumCircuit

# Create a new circuit with a single qubit
qc = QuantumCircuit(1)

# Add a Not gate to qubit 0
qc.x(0)

# Return a drawing of the circuit using MatPlotLib ("mpl"). This is the
# last line of the cell, so the drawing appears in the cell output.
qc.draw("mpl")
```

Out[2]:

You should see something similar to this as an output:

As long as you got the expected output, you're just about finished with this step. The last part is to install the other required packages you will need to complete this notebook. Some of these packages you will actually use, but others are only there because the auto-grader we use requires it. More on the grader in the next section 😃

In [3]:
```python
### Install the other required packages as well

!pip install qiskit_aer
!pip install qiskit_ibm_runtime
!pip install matplotlib
!pip install pylatexenc
!pip install git+https://github.com/qiskit-community/Quantum-Challenge-Gr
```

```
Requirement already satisfied: qiskit_aer in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (0.15.1)
Requirement already satisfied: qiskit>=1.1.0 in /home/jsoccerd/Coding/Qisk
it-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_aer)
(1.2.4)
Requirement already satisfied: numpy>=1.16.3 in /home/jsoccerd/Coding/Qisk
it-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_aer)
(2.1.2)
Requirement already satisfied: scipy>=1.0 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_aer) (1.14.1)
Requirement already satisfied: psutil>=5 in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from qiskit_aer) (6.1.0)
Requirement already satisfied: rustworkx>=0.15.0 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0-
>qiskit_aer) (0.15.1)
Requirement already satisfied: sympy>=1.3 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->qiskit
_aer) (1.13.3)
Requirement already satisfied: dill>=0.3 in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->qiskit_
aer) (0.3.9)
Requirement already satisfied: python-dateutil>=2.8.0 in /home/jsoccerd/Co
ding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=
1.1.0->qiskit_aer) (2.9.0.post0)
Requirement already satisfied: stevedore>=3.0.0 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->
qiskit_aer) (5.3.0)
Requirement already satisfied: typing-extensions in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0-
>qiskit_aer) (4.12.2)
Requirement already satisfied: symengine<0.14,>=0.11 in /home/jsoccerd/Cod
ing/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=
1.1.0->qiskit_aer) (0.11.0)
Requirement already satisfied: six>=1.5 in /home/jsoccerd/Coding/Qiskit-Fa
ll-Fest-24/venv/lib/python3.12/site-packages (from python-dateutil>=2.8.0-
>qiskit>=1.1.0->qiskit_aer) (1.16.0)
Requirement already satisfied: pbr>=2.0.0 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from stevedore>=3.0.0->qis
kit>=1.1.0->qiskit_aer) (6.1.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from sympy>=1.3->
qiskit>=1.1.0->qiskit_aer) (1.3.0)
Requirement already satisfied: qiskit_ibm_runtime in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (0.31.0)
Requirement already satisfied: requests>=2.19 in /home/jsoccerd/Coding/Qis
kit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ibm_runtim
e) (2.32.3)
Requirement already satisfied: requests-ntlm>=1.1.0 in /home/jsoccerd/Codi
ng/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ibm_
runtime) (1.3.0)
Requirement already satisfied: numpy>=1.13 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ibm_runtime)
(2.1.2)
Requirement already satisfied: urllib3>=1.21.1 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ibm_runti
me) (2.2.3)
Requirement already satisfied: python-dateutil>=2.8.0 in /home/jsoccerd/Co
ding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ib
m_runtime) (2.9.0.post0)
Requirement already satisfied: websocket-client>=1.5.1 in /home/jsoccerd/C
```

```
oding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_i
bm_runtime) (1.8.0)
Requirement already satisfied: ibm-platform-services>=0.22.6 in /home/jsoc
cerd/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qi
skit_ibm_runtime) (0.58.0)
Requirement already satisfied: pydantic>=2.5.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ibm_runti
me) (2.9.2)
Requirement already satisfied: qiskit>=1.1.0 in /home/jsoccerd/Coding/Qisk
it-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit_ibm_runtim
e) (1.2.4)
Requirement already satisfied: ibm-cloud-sdk-core<4.0.0,>=3.22.0 in /home/
jsoccerd/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (fro
m ibm-platform-services>=0.22.6->qiskit_ibm_runtime) (3.22.0)
Requirement already satisfied: annotated-types>=0.6.0 in /home/jsoccerd/Co
ding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from pydantic>
=2.5.0->qiskit_ibm_runtime) (0.7.0)
Requirement already satisfied: pydantic-core==2.23.4 in /home/jsoccerd/Cod
ing/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from pydantic>=
2.5.0->qiskit_ibm_runtime) (2.23.4)
Requirement already satisfied: typing-extensions>=4.6.1 in /home/jsoccerd/
Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from pydanti
c>=2.5.0->qiskit_ibm_runtime) (4.12.2)
Requirement already satisfied: six>=1.5 in /home/jsoccerd/Coding/Qiskit-Fa
ll-Fest-24/venv/lib/python3.12/site-packages (from python-dateutil>=2.8.0-
>qiskit_ibm_runtime) (1.16.0)
Requirement already satisfied: rustworkx>=0.15.0 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0-
>qiskit_ibm_runtime) (0.15.1)
Requirement already satisfied: scipy>=1.5 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->qiskit
_ibm_runtime) (1.14.1)
Requirement already satisfied: sympy>=1.3 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->qiskit
_ibm_runtime) (1.13.3)
Requirement already satisfied: dill>=0.3 in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->qiskit_
ibm_runtime) (0.3.9)
Requirement already satisfied: stevedore>=3.0.0 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=1.1.0->
qiskit_ibm_runtime) (5.3.0)
Requirement already satisfied: symengine<0.14,>=0.11 in /home/jsoccerd/Cod
ing/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qiskit>=
1.1.0->qiskit_ibm_runtime) (0.11.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /home/jsoccerd/
Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from request
s>=2.19->qiskit_ibm_runtime) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from requests>=2.19->qis
kit_ibm_runtime) (3.10)
Requirement already satisfied: certifi>=2017.4.17 in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from requests>=
2.19->qiskit_ibm_runtime) (2024.8.30)
Requirement already satisfied: cryptography>=1.3 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from requests-ntlm>
=1.1.0->qiskit_ibm_runtime) (43.0.3)
Requirement already satisfied: pyspnego>=0.4.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from requests-ntlm>=
1.1.0->qiskit_ibm_runtime) (0.11.1)
Requirement already satisfied: cffi>=1.12 in /home/jsoccerd/Coding/Qiskit-
```

```
Fall-Fest-24/venv/lib/python3.12/site-packages (from cryptography>=1.3->re
quests-ntlm>=1.1.0->qiskit_ibm_runtime) (1.17.1)
Requirement already satisfied: PyJWT<3.0.0,>=2.8.0 in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ibm-cloud-sd
k-core<4.0.0,>=3.22.0->ibm-platform-services>=0.22.6->qiskit_ibm_runtime)
(2.9.0)
Requirement already satisfied: pbr>=2.0.0 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from stevedore>=3.0.0->qis
kit>=1.1.0->qiskit_ibm_runtime) (6.1.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from sympy>=1.3->
qiskit>=1.1.0->qiskit_ibm_runtime) (1.3.0)
Requirement already satisfied: pycparser in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from cffi>=1.12->cryptograp
hy>=1.3->requests-ntlm>=1.1.0->qiskit_ibm_runtime) (2.22)
Requirement already satisfied: matplotlib in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib)
(1.3.0)
Requirement already satisfied: cycler>=0.10 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib) (0.1
2.1)
Requirement already satisfied: fonttools>=4.22.0 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib)
(4.54.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib)
(1.4.7)
Requirement already satisfied: numpy>=1.23 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib) (2.1.2)
Requirement already satisfied: packaging>=20.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib) (2
4.1)
Requirement already satisfied: pillow>=8 in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from matplotlib) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib)
(3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in /home/jsoccerd/Codi
ng/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from matplotlib)
(2.9.0.post0)
Requirement already satisfied: six>=1.5 in /home/jsoccerd/Coding/Qiskit-Fa
ll-Fest-24/venv/lib/python3.12/site-packages (from python-dateutil>=2.7->m
atplotlib) (1.16.0)
Requirement already satisfied: pylatexenc in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (2.10)
Collecting git+https://github.com/qiskit-community/Quantum-Challenge-Grade
r.git
  Cloning https://github.com/qiskit-community/Quantum-Challenge-Grader.git
to /tmp/pip-req-build-c5alt8w3
  Running command git clone --filter=blob:none --quiet https://github.com/
qiskit-community/Quantum-Challenge-Grader.git /tmp/pip-req-build-c5alt8w3
  Resolved https://github.com/qiskit-community/Quantum-Challenge-Grader.gi
t to commit c481fa96741396aa368a8f857275cf24f65124c4
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: typeguard in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from qc_grader==0.20.2)
```

```
(4.3.0)
Requirement already satisfied: jsonpickle==3.0.3 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qc_grader==0.2
0.2) (3.0.3)
Requirement already satisfied: ipycytoscape in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from qc_grader==0.20.2)
(1.3.3)
Requirement already satisfied: plotly in /home/jsoccerd/Coding/Qiskit-Fal
l-Fest-24/venv/lib/python3.12/site-packages (from qc_grader==0.20.2) (5.2
4.1)
Requirement already satisfied: networkx==3.2.1 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from qc_grader==0.2
0.2) (3.2.1)
Requirement already satisfied: graphviz in /home/jsoccerd/Coding/Qiskit-Fa
ll-Fest-24/venv/lib/python3.12/site-packages (from qc_grader==0.20.2) (0.2
0.3)
Requirement already satisfied: ipywidgets>=7.6.0 in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipycytoscape->
qc_grader==0.20.2) (8.1.5)
Requirement already satisfied: spectate>=1.0.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipycytoscape->qc
_grader==0.20.2) (1.0.1)
Requirement already satisfied: tenacity>=6.2.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from plotly->qc_grade
r==0.20.2) (9.0.0)
Requirement already satisfied: packaging in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from plotly->qc_grader==0.2
0.2) (24.1)
Requirement already satisfied: typing-extensions>=4.10.0 in /home/jsoccer
d/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from typeg
uard->qc_grader==0.20.2) (4.12.2)
Requirement already satisfied: comm>=0.1.3 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipywidgets>=7.6.0->
ipycytoscape->qc_grader==0.20.2) (0.2.2)
Requirement already satisfied: ipython>=6.1.0 in /home/jsoccerd/Coding/Qis
kit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipywidgets>=
7.6.0->ipycytoscape->qc_grader==0.20.2) (8.29.0)
Requirement already satisfied: traitlets>=4.3.1 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipywidgets>=
7.6.0->ipycytoscape->qc_grader==0.20.2) (5.14.3)
Requirement already satisfied: widgetsnbextension~=4.0.12 in /home/jsoccer
d/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipywi
dgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (4.0.13)
Requirement already satisfied: jupyterlab-widgets~=3.0.12 in /home/jsoccer
d/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipywi
dgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (3.0.13)
Requirement already satisfied: decorator in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from ipython>=6.1.0->ipywid
gets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from ipython>=6.1.0->ipywi
dgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (0.19.1)
Requirement already satisfied: matplotlib-inline in /home/jsoccerd/Coding/
Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipython>=
6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (0.1.7)
Requirement already satisfied: prompt-toolkit<3.1.0,>=3.0.41 in /home/jsoc
cerd/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ip
ython>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (3.0.48)
Requirement already satisfied: pygments>=2.4.0 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipython>=6.1.0->
```

```
ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (2.18.0)
Requirement already satisfied: stack-data in /home/jsoccerd/Coding/Qiskit-
Fall-Fest-24/venv/lib/python3.12/site-packages (from ipython>=6.1.0->ipywi
dgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (0.6.3)
Requirement already satisfied: pexpect>4.3 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from ipython>=6.1.0->ipy
widgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (4.9.0)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /home/jsoccerd/Codin
g/Qiskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from jedi>=0.16->
ipython>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2)
(0.8.4)
Requirement already satisfied: ptyprocess>=0.5 in /home/jsoccerd/Coding/Qi
skit-Fall-Fest-24/venv/lib/python3.12/site-packages (from pexpect>4.3->ipy
thon>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (0.7.0)
Requirement already satisfied: wcwidth in /home/jsoccerd/Coding/Qiskit-Fal
l-Fest-24/venv/lib/python3.12/site-packages (from prompt-toolkit<3.1.0,>=
3.0.41->ipython>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.2
0.2) (0.2.13)
Requirement already satisfied: executing>=1.2.0 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from stack-data->ipy
thon>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (2.1.0)
Requirement already satisfied: asttokens>=2.1.0 in /home/jsoccerd/Coding/Q
iskit-Fall-Fest-24/venv/lib/python3.12/site-packages (from stack-data->ipy
thon>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (2.4.1)
Requirement already satisfied: pure-eval in /home/jsoccerd/Coding/Qiskit-F
all-Fest-24/venv/lib/python3.12/site-packages (from stack-data->ipython>=
6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.20.2) (0.2.3)
Requirement already satisfied: six>=1.12.0 in /home/jsoccerd/Coding/Qiski
t-Fall-Fest-24/venv/lib/python3.12/site-packages (from asttokens>=2.1.0->s
tack-data->ipython>=6.1.0->ipywidgets>=7.6.0->ipycytoscape->qc_grader==0.2
0.2) (1.16.0)
```

For the record, you may see this error at the end of installs: `ERROR: pip's dependency resolver does not currently take into account all the packages that are installed.`

It is safe to ignore that particular error.

If everything else installed, then good news: you're ready to move on to configure your challenge environment.

Troubleshoot

Let's check your installs and versions to make sure everything is what it should be.

In [4]:
```python
### CHECK QISKIT VERSION
import qiskit
qiskit.__version__
```

Out[4]: '1.2.4'

You should have Qiskit version 1.2 or higher installed. If you see a version lower than that, you'll need to restart your kernel and re-run the code cells above.

While we're here, let's double-check that the other dependencies are correctly installed. You might be saying "*Is this really necessary after I just installed these packages just a few*

minutes ago?"

Probably not.

But, in the wild world of Jupyter Notebooks, understanding how to diagnose a problem is an invaluable asset. Checking to make sure all your packages are installed in the way that you expect is one of the first bug-fixing steps to take, and sometimes solves your entire problem right there.

Remember, if you need to reinstall any packages, restart your kernel (or runtime) first.

In [5]:
```python
### CHECK OTHER DEPENDENCIES
!pip show pylatexenc matplotlib qc_grader
```

```
Name: pylatexenc
Version: 2.10
Summary: Simple LaTeX parser providing latex-to-unicode and unicode-to-lat
ex conversion
Home-page: https://github.com/phfaist/pylatexenc
Author: Philippe Faist
Author-email: philippe.faist@bluewin.ch
License: MIT
Location: /home/jsoccerd/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/si
te-packages
Requires:
Required-by:
---
Name: matplotlib
Version: 3.9.2
Summary: Python plotting package
Home-page:
Author: John D. Hunter, Michael Droettboom
Author-email: Unknown <matplotlib-users@python.org>
License: License agreement for matplotlib versions 1.3.0 and later
=========================================================

1. This LICENSE AGREEMENT is between the Matplotlib Development Team
("MDT"), and the Individual or Organization ("Licensee") accessing and
otherwise using matplotlib software in source or binary form and its
associated documentation.

2. Subject to the terms and conditions of this License Agreement, MDT
hereby grants Licensee a nonexclusive, royalty-free, world-wide license
to reproduce, analyze, test, perform and/or display publicly, prepare
derivative works, distribute, and otherwise use matplotlib
alone or in any derivative version, provided, however, that MDT's
License Agreement and MDT's notice of copyright, i.e., "Copyright (c)
2012- Matplotlib Development Team; All Rights Reserved" are retained in
matplotlib  alone or in any derivative version prepared by
Licensee.

3. In the event Licensee prepares a derivative work that is based on or
incorporates matplotlib or any part thereof, and wants to
make the derivative work available to others as provided herein, then
Licensee hereby agrees to include in any such work a brief summary of
the changes made to matplotlib .

4. MDT is making matplotlib available to Licensee on an "AS
IS" basis.  MDT MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR
IMPLIED.  BY WAY OF EXAMPLE, BUT NOT LIMITATION, MDT MAKES NO AND
DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS
FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF MATPLOTLIB
WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. MDT SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF MATPLOTLIB
 FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR
LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING
MATPLOTLIB , OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF
THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material
breach of its terms and conditions.

7. Nothing in this License Agreement shall be deemed to create any
```

```
Requires: contourpy, cycler, fonttools, kiwisolver, numpy, packaging, pill
ow, pyparsing, python-dateutil
Required-by: seaborn
---
Name: qc_grader
Version: 0.20.2
Summary: Grading client for the IBM Quantum Challenge
Home-page: https://quantum.ibm.com/
Author: IBM Quantum Community Team
Author-email:
License: Apache 2.0
Location: /home/jsoccerd/Coding/Qiskit-Fall-Fest-24/venv/lib/python3.12/si
te-packages
Requires: graphviz, ipycytoscape, jsonpickle, networkx, plotly, typeguard
Required-by:
```

If the packages are installed properly, you will see a short list of attributes appear for each one above. Your `qc-grader` should be version 0.19.0 or higher.

Other common issues could be solved by the following:

- If you aren't seeing any output, make sure you've run each code cell.
- Try restarting your kernel. How you do this varies by the environment you're using.

If you're still running into problems, talk to others at your Qiskit Fall Fest event, check out this Coding with Qiskit video for additional tips, or explore the documentation for help.

Configure your environment

There are just a few last things to set up before you're ready to begin.

Critically, you need to **store your API token** to track your progress. If you do not properly store your API token, the grader will not work.

This is so important that we're also going to make it a step by step list.

1. Navigate to the IBM Quantum Platform
2. Make sure you log in using the same email you registered for this event with
3. Look to the top right corner, click and copy your API token (pictured above)
4. In the code cell below, replace `deleteThisAndPasteYourTokenHere` with your token

```
In [6]:  ## Save your API token to track your progress
         ## You do not need quotation marks around your API token!

         %set_env QXToken=e80b7be20d34579547963c7f5152af69c9ef960e46d34c1dbde9366a
```

```
# Make sure there is no space between the equal sign
# and the beginning of your token
```

env: QXToken=e80b7be20d34579547963c7f5152af69c9ef960e46d34c1dbde9366a57e7d
e4d932faed82f707925b5bfb2f93fb3a70aa9074caf5a31fa1cbff83cb5dcbbf594

You do not need quotation marks around your API token!

And finally, we will run all the required imports for the rest of the notebook in the next cell, to keep things clean. In the future, this will be done at the beginning of each notebook.

In [7]:
```python
### Imports

from qiskit import QuantumCircuit
from qiskit.quantum_info import SparsePauliOp
from qiskit_ibm_runtime import EstimatorV2 as Estimator
from qiskit_aer import AerSimulator
import matplotlib.pyplot as plt
from qc_grader.challenges.fall_fest24 import grade_lab1_ex1, grade_lab1_w
import os
```

Exercise 1: Testing the grader cell

Every exercise you encounter in the Qiskit Fall Fest is followed by a **grader cell**. The grader takes the code from the exercise you just completed and passes it to a *hidden auto-grader*. Sometimes it will simply tell you "right" or "wrong." However, for more complex exercises, we include hints or suggestions. Each grader cell is unique to each exercise. Make sure you run all the code cells prior to running the grader.

In this first exercise, all we're doing is checking to see if the variable `answer` has a value which is a string.

Your Task: Change the value of the variable `answer` to be a string. This string can be anything you like.

In [8]:
```python
### Exercise Cell - this is where you add your own code to solve the ques
### To pass the grader, the variable answer must be a "string"

answer = str(42)
```

You do not need to add any code into the grader cell. Simply run the grader cell to submit your answer.

Try it out below to see if you completed Exercise 1 properly.

In [9]:
```python
### Grader Cell - Submit your answer by running the following code
### You do not need to add any code into this cell

grade_lab1_welcome(answer)
```

```
Grading your answer. Please wait...
Welcome to the Qiskit Fall Fest!🎉If you are seeing this, that means you h
ave successfully figured out how to use the grader for these notebooks. Ro
ck on. We at IBM Quantum look forward to your participation in this event,
thanks for being here!
```

If you received an error, here are some potential fixes:

Error: `argument "answer" (int) is not an instance of str`

- *Solution:* You need to change the value of the `answer` variable to be a string. A string is a word or sentence contained within quotation marks.

Error: `Failed: 401 Client Error: Unauthorized for url: https://auth.quantum.ibm.com/api/users/loginWithToken`

- *Solution:* Return to the Configure your challenge environment section and properly store your API token

Error: `Failed: Unable to access service (Forbidden)`

- *Solution:* You did not properly install your grader package. Restart your kernel and run all above code cells before returning here

If you have a different error, work with the other folks at your Qiskit Fall Fest event to try and solve it!

Generate a two-qubit Bell state using Qiskit patterns

In this notebook we follow along with the third episode of Coding with Qiskit 1.x. This is the first graded notebook in the Qiskit Fall Fest, so you can expect it to be easy. Solving future notebooks may require much more thought and research.

Our goal in this notebook is to generate a two-qubit Bell state using the approach provided by **Qiskit patterns.**

A Qiskit pattern is a four step workflow for setting up, running, and getting results from a quantum computer. This is the workflow designed to help you use utility-scale quantum computers. It works as follows:

1. Map circuits and operators
2. Optimize the circuit
3. Execute the circuit
4. Post-process the results

Hopefully that workflow seems familiar to you.

Qiskit pattern step 1: Map circuits and operators

Mapping your problem to circuits and operators is fairly easy when you're only dealing with a handful of qubits, but can get more difficult as you scale up. To start, we will construct a Bell state. This is a classic (*or should I say, "quantum"*) example that showcases entanglement between two qubits.

First we apply a Hadamard gate to qubit 0, placing it in a superposition. Then we apply a CNOT gate between qubits 0 and 1. The CNOT effectively entangles our qubits, meaning that when we measure the output of this circuit we should expect to see that both qubits share a measurement result.

If that made sense to you, nice job, you're ready to go. However, if that went over your head, the graded notebooks will be quite difficult for you. Instead, you may want to take this time to review the course on Quantum Information and Computation or attend a workshop from your Qiskit Fall Fest event.
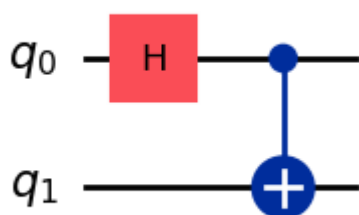
In [10]:
```python
# Create a new circuit with two qubits
qc = QuantumCircuit(2)

# Add a Hadamard gate to qubit 0
qc.h(0)

# Perform a CNOT gate on qubit 1, controlled by qubit 0
qc.cx(0, 1)

# Return a drawing of the circuit using MatPlotLib ("mpl"). This is the
# last line of the cell, so the drawing appears in the cell output.
qc.draw("mpl")
```

Out[10]:

We also want to be able to visualize that our circuit does what we expect it to do. To accomplish this, we need to set up some **operators**.

In Exercise 2 you will see multiple operators being created. Some (like XX) measure both qubits, while others (like XI) only measure one. Later on in this notebook we will check their expectation values to make sure our Bell-state circuit is entangled as we expect.

Because we've created an entangled Bell state, we should expect that our operators only measuring one qubit have an expectation value of 0. Similarly, we should expect the operators which measure both qubits to come back with some non-zero (hopefully

close to 1) expectation value.

So how do we actually make these operators? We simply use a combination of `X`, `Z`, and Identity (`I`) gates.

- Applying an `I` gate to a qubit is the same as doing nothing to a qubit. We can use a combination of `X` / `Z` and `I` to make the operators of `XI`, `IX`, `ZI`, and `IZ`.
- We only have two operators which measure both qubits: `XX` and `ZZ`. We use a combination of `X` or `Z` gates to make each one, respectively.

Later on in this notebook, you will graph the expectation values for each of these operators to visualize your circuit.

Exercise 2

In this exercise, you need to complete the list of operators. We have provided three for you already: the `ZZ`, `ZI`, and `IX` observables. We've also written comments in the code explaining what each observable is looking for.

Your Task: Following the same approach, create the last three operators: `IZ`, `XX`, and `XI`

In [11]:
```python
# The ZZ applies a Z operator on qubit 0, and a Z operator on qubit 1
ZZ = SparsePauliOp('ZZ')

# The ZI applies a Z operator on qubit 0, and an Identity operator on qub
ZI = SparsePauliOp('ZI')

# The IX applies an Identity operator on qubit 0, and an X operator on qu
IX = SparsePauliOp('IX')



### Write your code below here ###
### Follow the same naming convention we used above

IZ = SparsePauliOp("IZ")

XI = SparsePauliOp("XI")

XX = SparsePauliOp("XX")

## Don't change any code past this line, but remember to run the cell.

observables = [IZ, IX, ZI, XI, ZZ, XX]
```

In [12]:
```python
# Submit your answer using following code

grade_lab1_ex1(observables)
```

Grading your answer. Please wait...

Congratulations 🎉! Your answer is correct.

> If you got an error, here are some possible solutions:
>
> **Error:** `NameError: name 'observables' is not defined`
>
> - *Solution:* You need to run the code cell in Exercise 2 prior to running the Grader cell
>
> **Error:** `item 0 of argument "observables" (list) is not an instance of qiskit.quantum_info.operators.symplectic.pauli.Pauli`
>
> - *Solution:* The answers you provided were not Pauli objects. Revise your answer.

Qiskit pattern step 2: Optimize the circuit

A Qiskit pattern is the best framework for working with a quantum computer at *utility scale* — the point where quantum computers can perform reliable computations at a scale beyond brute force classical computing methods. These computations can use hundreds of qubits. However, in this intro section, we're only using two qubits. So, that means we do not need to do any work here to optimize the circuit.

To optimize the circuit, we can use the Qiskit transpiler. During transpilation, we can optimally layout our qubits to minimize the required number of gates, for example. You can learn more about this process in graded notebook 3. For now, you can move on to the next step of the pattern.

Qiskit pattern step 3: Execute the circuit

Time to run the circuit.

In this next cell, you might notice something new. We are going to run the circuit using the "Estimator," which is a part of the Qiskit Runtime.

Graded notebook 2 will cover Qiskit Runtime and the primitives in much greater detail. We won't talk about it too much here other than to say that Qiskit Runtime gives us the easiest means of running our circuit.

Here we run our circuit using the Estimator, then save the results as the variable `job`. You may now proceed to the next and final step of Qiskit patterns, where we will visualize this information.

```python
In [13]:   # Set up the Estimator
           estimator = Estimator(AerSimulator())

           # Submit the circuit to Estimator
           pub = (qc, observables)

           job = estimator.run(pubs=[pub])
```

Qiskit pattern step 4: Post-process the results

The last step of Qiskit patterns is where we process, analyze, and visualize the results of our job.

Here we are going to collect our data and plot it on a graph we construct.

In [14]:
```python
# Collect the data
data = ['IZ', 'IX', 'ZI', 'XI', 'ZZ', 'XX']
values = job.result()[0].data.evs

# Set up our graph
container = plt.plot(data, values, '-o')

# Label each axis
plt.xlabel('Observables')
plt.ylabel('Values')

# Draw the final graph
plt.show()
```
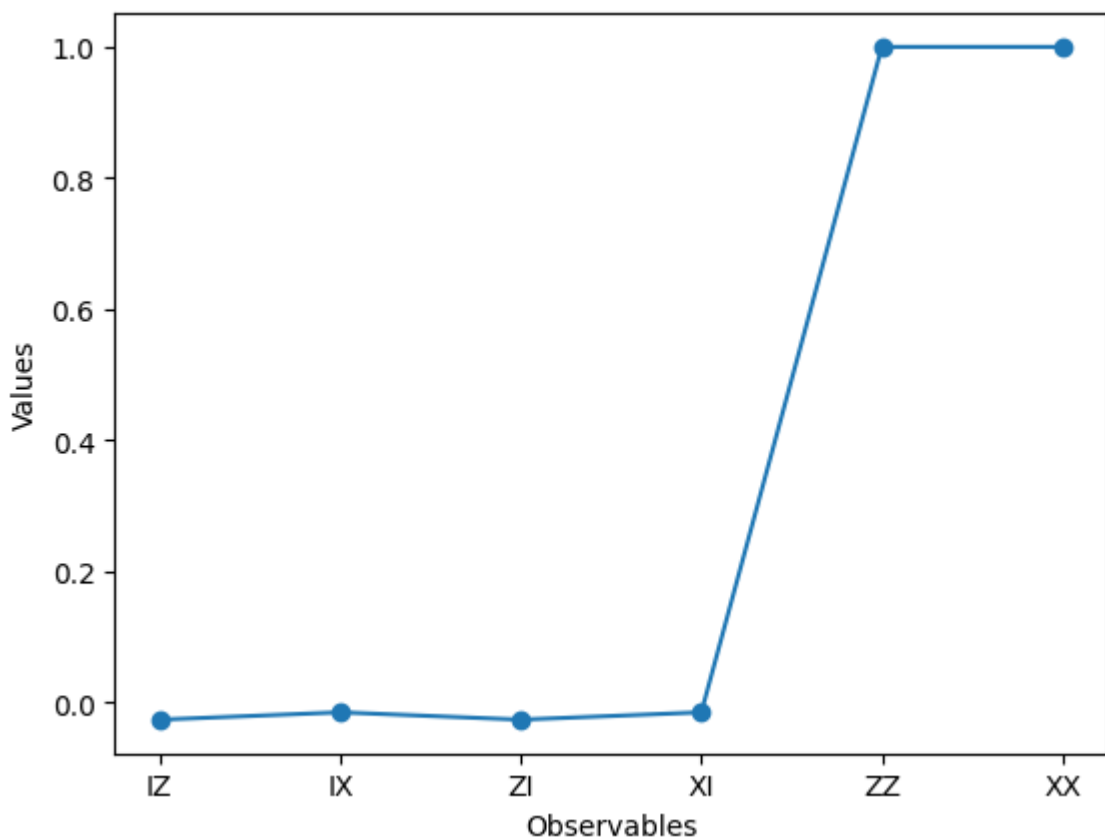

You *should* see the operators which measure both qubits (ZZ and XX) have a value that's close to 1, and the other operators have a value that's close to 0. This shows that your circuit is entangled. Great work.

The plot displays as a line graph, but you can also visualize this as a bar graph.

In [15]:
```python
container = plt.bar(data, values, width=0.8)

plt.xlabel('Observables')
```

```
plt.ylabel('Values')
plt.show()
```
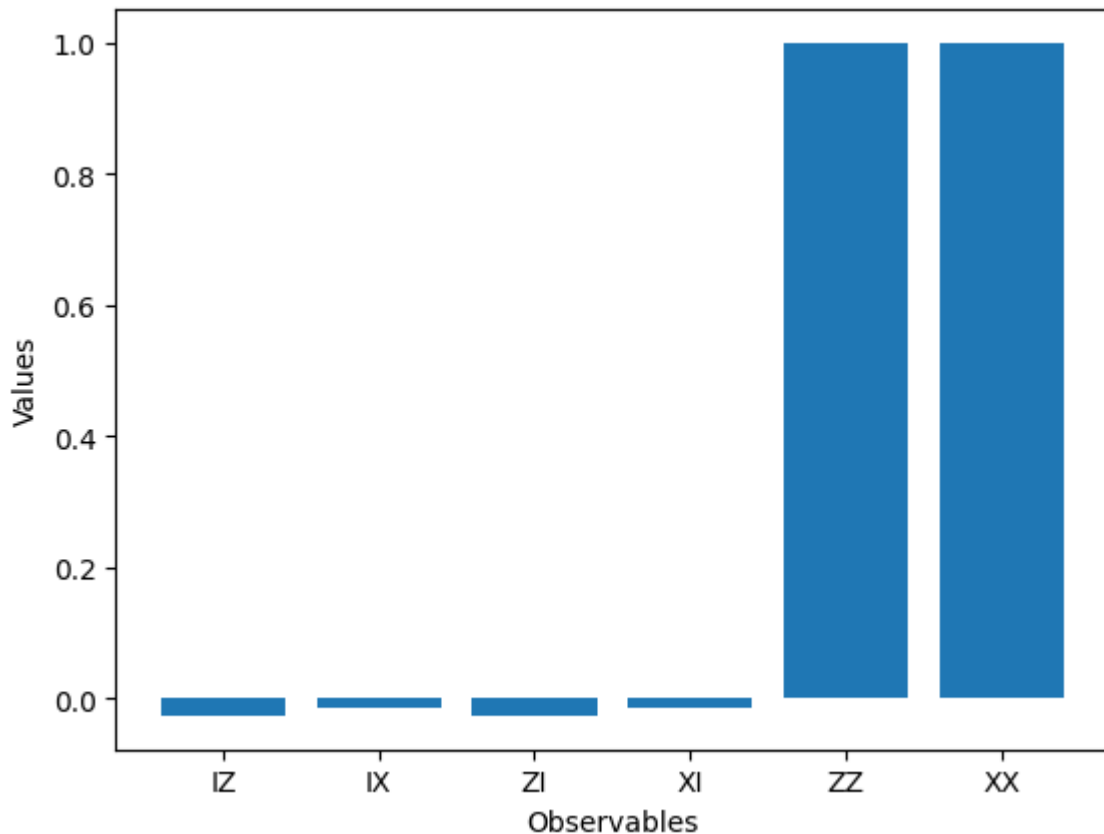

Congratulations!

Congratulations on making it to the end of the first Qiskit Fall Fest notebook!

You learned how to follow the workflow of Qiskit patterns to set up, optimize, run, and process your circuit. You successfully created a two-qubit Bell state, and showed that it was properly entangled by visualizing the operators. You learned how to use the auto-grader as well. Rock on!

With the basics down, you're now ready to begin thinking like a quantum developer. It's a little different than thinking like a quantum physicist, though. We're not just making discoveries about how the world works—we're trying to use quantum mechanics as a tool to solve meaningful problems and enact positive change in the world.

We like to say we've entered the era of quantum utility—for the first time, quantum computers can do things beyond the abilities of brute-force classical computing. Now, we're looking for quantum advantage, where quantum computers are the best way to solve the problem. We need you, the quantum community for that—we need to uncover and implement quantum algorithms and apply them to real-world use cases. The rest of the challenge will be devoted to that—getting you to start thinking like a quantum developer.

And of course, solving problems with real-world impact means thinking about who

you're impacting and how. At IBM we hold that we must research and develop technology responsibly. To that end, quantum computing and ethics experts at IBM have been researching the societal implications of quantum computing, and our role as a quantum computing provider in mitigating potentially undesired consequences of the technology. We encourage you to learn more by reading about our Responsible Quantum Computing effort.

Enjoy the rest of your Qiskit Fall Fest event, and good luck!

Additional information

Created by: Brian Ingmanson

Advised by: Junye Huang, Va Barbosa, Sophy Shin

Version: 1.1.0

Summary

I learned a Bell state can be created with a Hadamrd gate followed by a CNOT gate. Furthermore I learned/relized that measuring an entangled state with different observables will result in a value of 0