Python Warm-Up Studio

Studio Handout

Session Overview

Build and refine your Lab Handout 1 warm-up notebook. Work in **pairs** while the instructor circulates to assist.

Session Objectives:

- Complete M1 homework with peer support
- Practice NumPy arrays for quantum shot data
- Generate/interpret probability visualizations
- Build notebook habits (markdown + code structure)

Timed Agenda (75 minutes)

1. **0:00–0:10** Review M1 + demo solutions
2. **0:10–1:05** Paired workshop (55 min work time)
3. **1:05–1:13** Group share-out (2-3 volunteers)
4. **1:13–1:15** Collect + Week 2 preview

1 Quick M1 Review (Instructor Demo)

Run these cells to refresh basic syntax before pairs begin.

```python
# M1 REFRESHER: Basic qubit probability calculation
import numpy as np

shots = 1024
outcomes = np.random.choice([0, 1], size=shots, p=[0.7, 0.3])
counts = np.bincount(outcomes)

print(f"Theoretical: P(|0>)=70%, P(|1>)=30%")
print(f"Observed: P(|0>)={counts[0]/shots:.1%}, P(|1>)={counts[1]/shots:.1%}")
print(f"Counts: |0>={counts[0]}, |1>={counts[1]}")
```

Expected: Observed probabilities close to theoretical values.

2 Paired Workshop Tasks (55 minutes total)

Instructions: Work together! One types, one reviews. Switch roles halfway.
Submit: Export PDF → Upload on Canvas by start of Lab 2 M1

Task 1: Arrays for Qubit Probabilities (10 min)

Complete the code below. Expected: probs ≈ 70%/30%

```
import numpy as np

# Task 1: Simulate biased qubit (70% |0>, 30% |1>)
shots = 1024
outcomes = np.random.choice([0, 1], size=shots, p=[0.7, 0.3])

# Compute empirical probabilities
counts = np.bincount(outcomes)
prob_0 = counts[0] / shots
prob_1 = counts[1] / shots

print(f"P(|0>) = {prob_0:.1%}, P(|1>) = {prob_1:.1%}")
print(f"Counts: |0>={counts[0]}, |1>={counts[1]}")

# CHECK: Probs should be close to 70%/30%
```

Task 2: Visualize Quantum Data (15 min)

Create a bar plot like Qiskit histograms. Use `counts` from Task 1.

```
import matplotlib.pyplot as plt
%matplotlib inline

# Task 2: Bar plot of shot counts
states = ['|0>', '|1>']
plt.figure(figsize=(8, 5))
plt.bar(states, counts, alpha=0.7, color=['skyblue', 'salmon'])
plt.xlabel('Qubit States')
plt.ylabel('Shot Counts')
plt.title('Mock Qubit Measurement (1024 shots)')
plt.grid(axis='y', alpha=0.3)
plt.show()

# BONUS: Rerun Task 1 with p=[0.5, 0.5] (equal superposition)
# How does the plot change?
```

Task 3: Multi-Qubit Preview (15 min)

Simulate 2-qubit Bell state: Only $|00\rangle$ (0) or $|11\rangle$ (3) outcomes.

```
# Task 3: 2-qubit correlated measurements
shots = 1024
measurements_2q = np.random.choice([0, 3], size=shots, p=[0.5, 0.5])  # |00> or |11>
unique, counts_2q = np.unique(measurements_2q, return_counts=True)

plt.figure(figsize=(8, 5))
plt.bar(unique.astype(str), counts_2q / shots * 100, alpha=0.7, color=['gold', 'purple'])
plt.xlabel('2-Qubit State (decimal)')
plt.ylabel('Probability (%)')
plt.title('Bell State: |00> + |11> (1024 shots)')
plt.show()

print("Observed probabilities:")
for state, count in zip(unique, counts_2q/shots):
    print(f" State {int(state)}: {count:.1%}")
```

Task 4: Documentation (10 min)

Add these markdown sections (double-click cells above to edit):

Methods

Describe your simulation approach here.

- Generated 1024 shots using `np.random.choice()`
- Theoretical probabilities: P($|0\rangle$)=70%, P($|1\rangle$)=30%
- Used `np.bincount()` for efficient counting

Key Question: How does increasing `shots` affect plot smoothness?

Expected vs Observed

State	Expected	Observed	Notes
$\|0\rangle$	70.0%	______%	
$\|1\rangle$	30.0%	______%	

One-sentence observation: *Fill this in!*

3 Submission Checklist

Task 1: Probabilities print ($\approx$ 70%/30%)

Task 2: Bar plot displays

Task 3: 2-qubit plot shows correlations

Task 4: "Methods" + "Expected vs Observed" sections complete

Export: File → Download as → PDF via LaTeX

Grade Rubric: Code runs (40%) | Plot present (30%) | Markdown structure (30%)

4 Next Steps: Lab 2

- IBM Quantum account signup
- Real quantum hardware access
- Your notebook = template for future labs!