Task 3 (3A, 3B, 3C) - Using Rotation to Obtain Probabilities

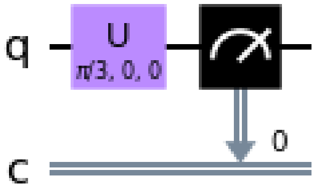
Task 3A Fill in the function below to return `qc_rot_a`, a single-qubit QuantumCircuit satisfying the following conditions: - it performs a measurement to a single classical bit - Pr(seeing $|0\rangle$ on measurement) = `0.75` - your circuit only uses gates from the following list: X, Y, Z, P, H, U

Plot your results using a histogram to verify your solution over `1024` trials.

```
In [22]: def qc_rot_a():
             # BEGIN SOLUTION
             # Find wave amplitudes for ket zero and ket one
             # Use amplitudes to determine the proportion of rotation needed
             qc = QuantumCircuit(1,1)
             theta = np.pi/3
             phi = 0
             lamda = 0
             qc.u(theta, phi, lamda, 0)
             qc.measure(0,0)
             # END SOLUTION
             return qc
```

```
In [23]: qc_rot_a().draw(output='mpl')
```

```
Out[23]:
```

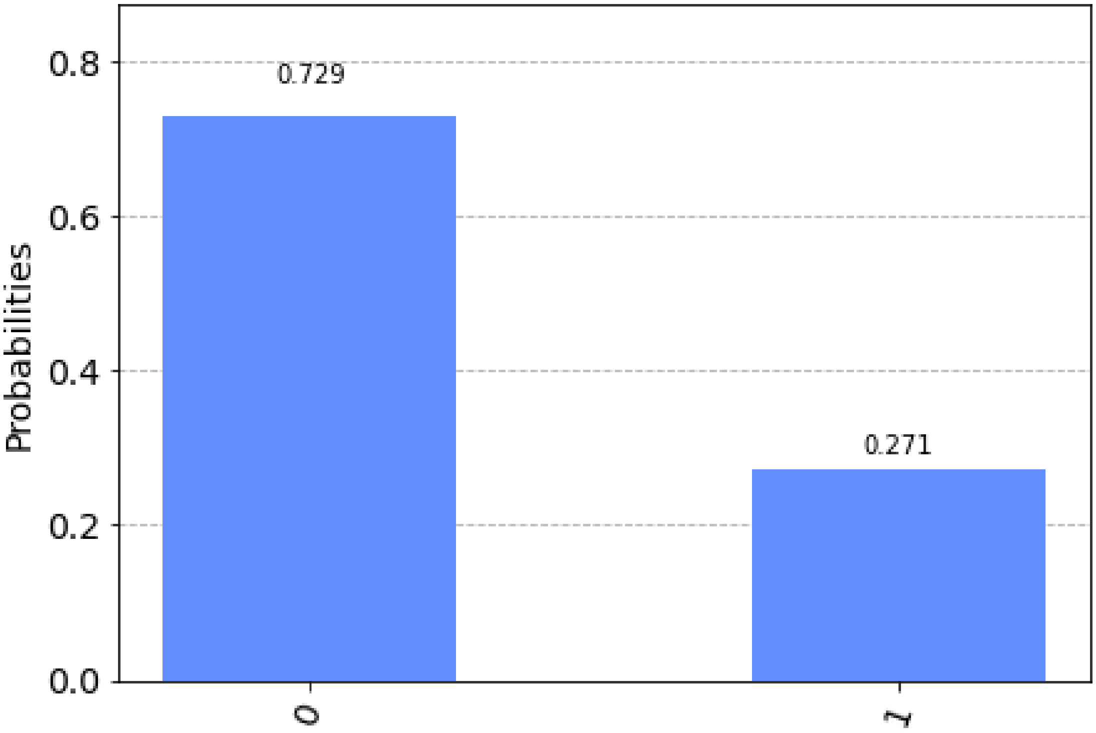


```
In [24]: # Plot your results in this cell!

         # BEGIN SOLUTION
         qc = qc_rot_a()
```

```
qasm_sim = BasicAer.get_backend("qasm_simulator")
job = execute(qc, qasm_sim)
counts = job.result().get_counts()
plot_histogram(counts)
# END SOLUTION
```

Out[24]:



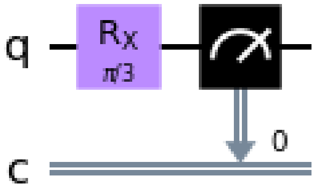
Task 3B - Rotation Operator Gates Again, fill in the function below to return `qc_rot_b`, a single-qubit QuantumCircuit satisfying the following conditions: - it performs a measurement to a single classical bit - Pr(seeing $|0\rangle$ on measurement) = `0.75` - your circuit only uses gates from the following list: RX, RY, RZ

Plot your results using a histogram to verify your solution over `1024` trials.

```
In [25]: def qc_rot_b():
             # BEGIN SOLUTION
             # Using the provided link, it can be seen that RX(pi) = -iX
             # From this info and part A, we need to use RX(pi/3) to get the desired state
             qc = QuantumCircuit(1,1)
             qc.rx(np.pi/3, 0)
             qc.measure(0,0)
             # END SOLUTION
             return qc
```

```
In [26]: qc_rot_b().draw(output='mpl')
```

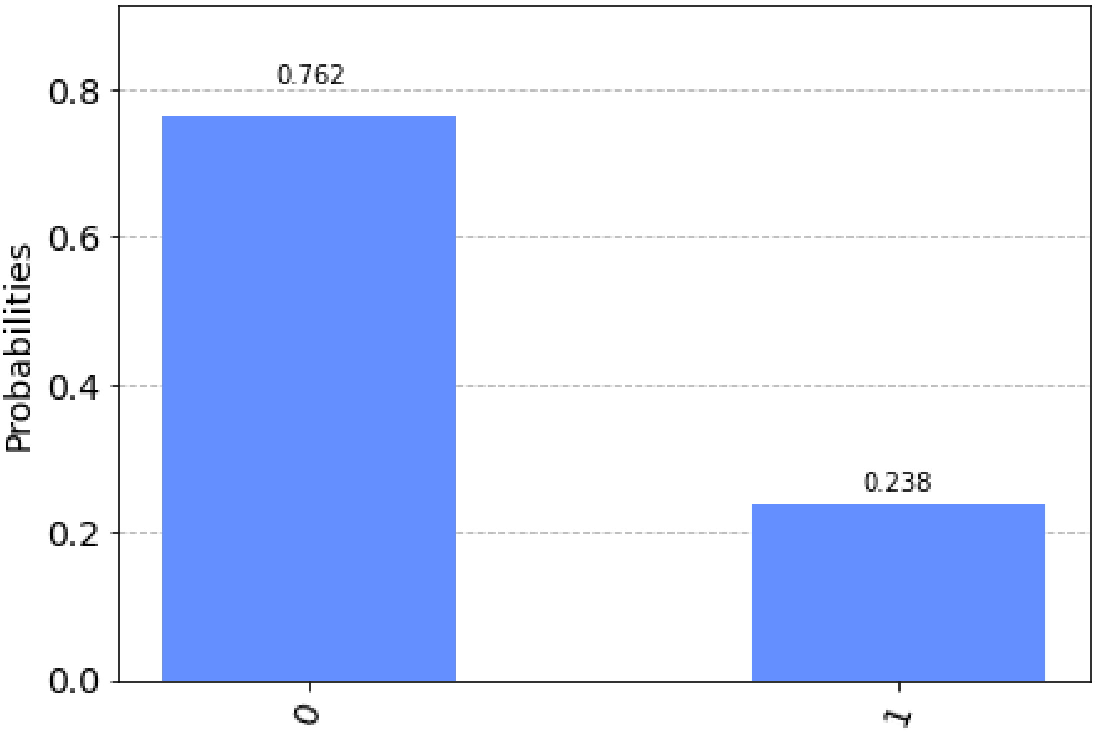
Out[26]:



```
In [27]: # Plot your results in this cell!

         # BEGIN SOLUTION
         qasm_sim = BasicAer.get_backend("qasm_simulator")
         job = execute(qc_rot_b(), qasm_sim)
         counts = job.result().get_counts()
         plot_histogram(counts)
         # END SOLUTION
```

Out[27]:


0.762
0.238
0.8
0.6
0.4
0.2
0.0
Probabilities
0
1

Task 3C Suppose we apply a Z gate to your circuit from task 3B just before measuring. How will the probability of measuring $|0\rangle$ change from that of the original circuit? Will measurement on the modified circuit yield a state equivalent to the original circuit up to a global phase?

Type your answer here, replacing this text.

The probability of seeing $|0\rangle$ won't change since Z is just a rotation about the z-axis. However, it's not equivalent up to a global phase since you can't pull out a factor.

Source: https://quantum-computing.ibm.com/composer/docs/iqx/operations_glossary#phase-gate

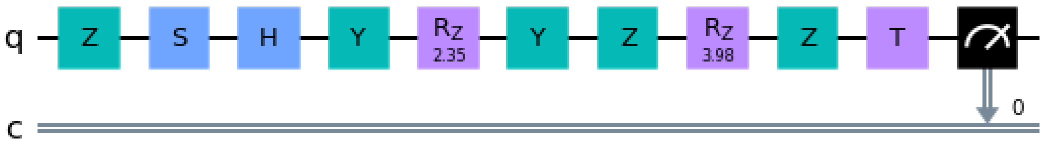
The P gate is equivalent up to a global phase with RZ. The P gate applies a phase to $|1\rangle$ of $e^{i\theta}$. Up to a global phase of $e^{i\theta/2}$, it is equivalent to $RZ(\theta)$

Task 5 (5A, 5B) - Transpiling Circuits When you submit a job to IBM, the quantum computer will most likely run a different circuit than you built. This is because the quantum computer can only do a very limited set of operations relative to the number of unitary gates. For IBM devices, the transpile step reduces all single-qubit operations to I, X, SX, and RZ (source).

Task 5A Choose one of the IBM backends (see the first assignment for a refresher on this). Use the transpile method to optimize the given circuit for the backend. Draw the transpiled circuit.

```
In [28]: qc_rand = random_circuit(1, 10, measure=True, seed=6)
         qc_rand.draw(output='mpl')
```

Out[28]:

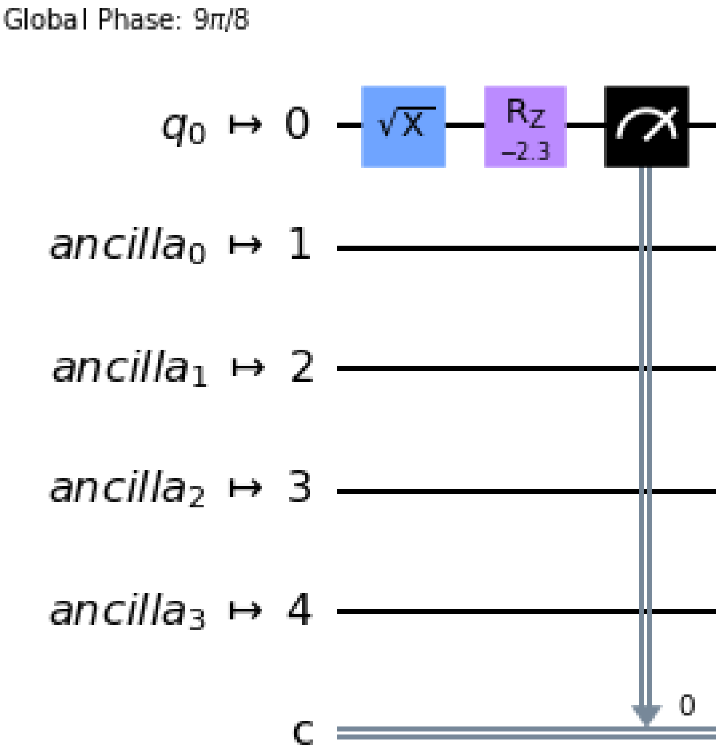


```
In [29]: IBMQ.load_account()
         # BEGIN SOLUTION
         provider = IBMQ.get_provider(hub='ibm-q')
         for backend in provider.backends():
             status = backend.status().to_dict()
             if status['operational'] and status['status_msg']=='active':
                 if 'simulator' not in status['backend_name']:
                     print(status['backend_name'])

         transpile(qc_rand, provider.get_backend('ibmq_lima')).draw(output='mpl')
         # END SOLUTION
```

```
ibmq_armonk
ibmq_bogota
ibmq_lima
ibmq_belem
ibmq_quito
ibmq_manila
```

Out[29]:


Global Phase: 9π/8
q₀ ↦ 0
ancilla₀ ↦ 1
ancilla₁ ↦ 2
ancilla₂ ↦ 3
ancilla₃ ↦ 4
c
√X
R_Z
−2.3
0

Task 5B Which gates from {I, X, SX, RZ}, and how many of each, are used in the transpiled circuit?

Type your answer here, replacing this text.

SX and RZ are used. There is one SX gate used and one RZ gate used.