

# Exercício 2 - Física Moderna (ANP)

Aluno: João Vitor Pereira Amorim

---

## Questões:

1. Gere o estado  $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$  e meça o qubit 20 vezes. Quantas vezes o resultado da medida indicou que o estado medido era  $|0\rangle$  e quantas vezes o estado medido foi  $|1\rangle$  ?
  2. Você consegue gerar essa estatística de forma mais eficiente? Mostre como e implemente sua solução.
- 

## Soluções:

### QUESTÃO 1)

Importação das bibliotecas:

In [154...

```
%matplotlib inline
import matplotlib
from math import pi
from qiskit import *
from qiskit.tools.visualization import plot_histogram
from qiskit.visualization import plot_bloch_multivector
from qiskit.extensions import Initialize
qiskit.__qiskit_version__
```

Out[154...

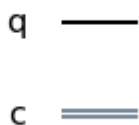
```
{'qiskit-terra': '0.15.2',
 'qiskit-aer': '0.6.1',
 'qiskit-ignis': '0.4.0',
 'qiskit-ibmq-provider': '0.9.0',
 'qiskit-aqua': '0.7.5',
 'qiskit': '0.21.0'}
```

Inicialização do circuito com um qubit e um registrador:

In [155...

```
circuit = QuantumCircuit(1, 1)
circuit.draw(output='mpl')
```

Out[155...

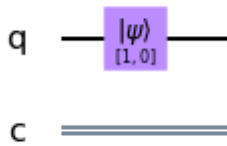


Inicialização do qubit no estado  $|0\rangle$ :

In [156...

```
initializer = Initialize([1,0])
circuit.append(initializer, [0])
circuit.draw(output='mpl')
```

Out[156...

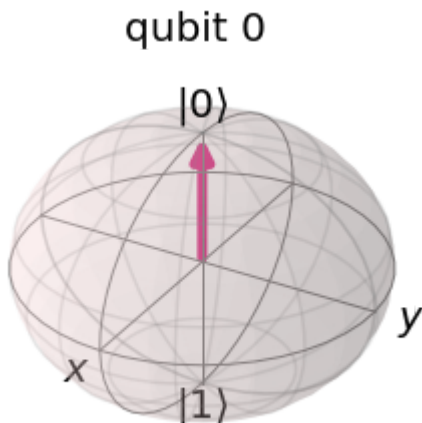


Podemos importar o simulador de vetor de estado quântico que vem incluído no Qiskit, para dessa forma podermos plotar o  $|0\rangle$  na Esfera de Bloch (com o `plot_bloch_multivector()`, importado no início):

In [157...

```
statevector_simulator = Aer.get_backend('statevector_simulator')
output = execute(circuit, backend = statevector_simulator).result()
vector = output.get_statevector()
plot_bloch_multivector(vector)
```

Out[157...



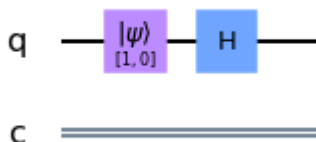
Na questão solicita-se que as medições sejam realizadas com o estado  $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ , ou seja, um dos estados de superposição. Para isso, é necessário gerá-lo a partir desse qubit que está no estado  $|0\rangle$ . Podemos fazer isso utilizando uma porta Hadamard, já que o efeito dela é justamente o de criação do estado de superposição:  $|+\rangle$ , se aplicada no qubit com estado  $|0\rangle$ , e  $|-\rangle$ , se aplicada no qubit com estado  $|1\rangle$ .

Portanto, abaixo é aplicada a porta Hadamard ao único qubit do circuito:

In [158...

```
circuit.h(0)
circuit.draw(output='mpl')
```

Out[158...



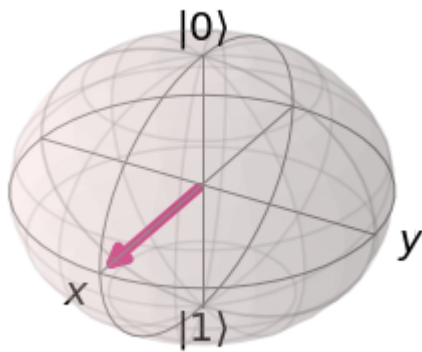
Conferindo se o qubit está no estado  $|+\rangle$ , através da plotagem na Esfera de Bloch:

In [159...

```
output = execute(circuit, backend = statevector_simulator).result()
vector = output.get_statevector()
plot_bloch_multivector(vector)
```

Out[159...

qubit 0

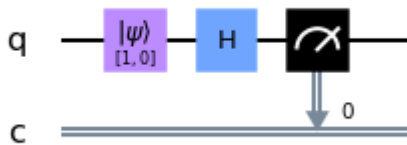


Antes de realizarmos as medições, adicionamos a operação de medida e salvamos em nosso único registrador:

In [160...

```
circuit.measure(0,0)
circuit.draw(output='mpl')
```

Out[160...



Agora que temos nosso qubit no estado de superposição  $|+\rangle$ , já podemos realizar as 20 medições nele. Para isso, primeiro importamos o "simulador de computador quântico", que permite executarmos medições em nosso circuito:

In [161...

```
qasm_simulator = Aer.get_backend('qasm_simulator')
```

Agora realizamos as medições 20 vezes:

In [180...

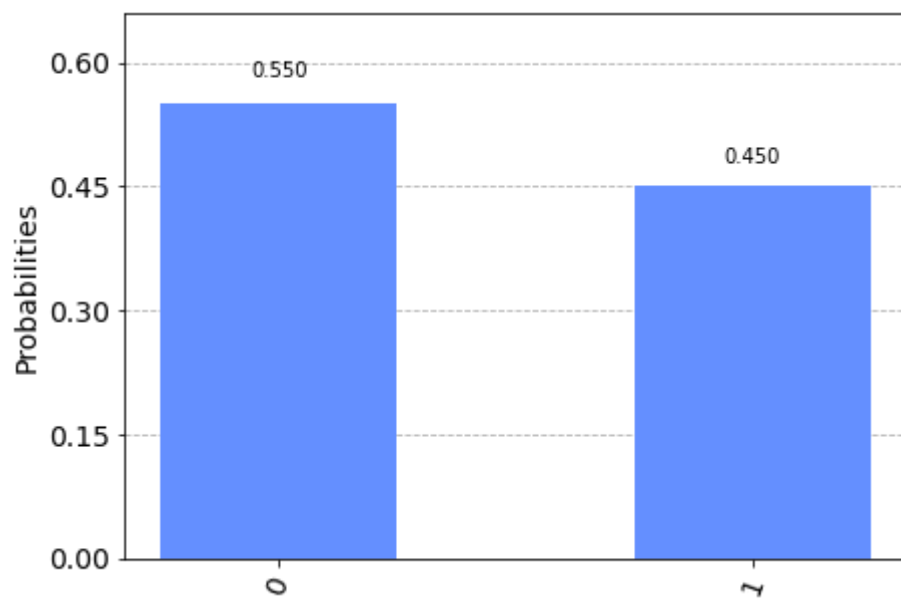
```
result = execute(circuit, backend = qasm_simulator, shots = 20).result()
```

E, por fim, plotamos o nosso resultado em um histograma:

In [181...

```
counts = result.get_counts()
plot_histogram(counts)
```

Out[181...

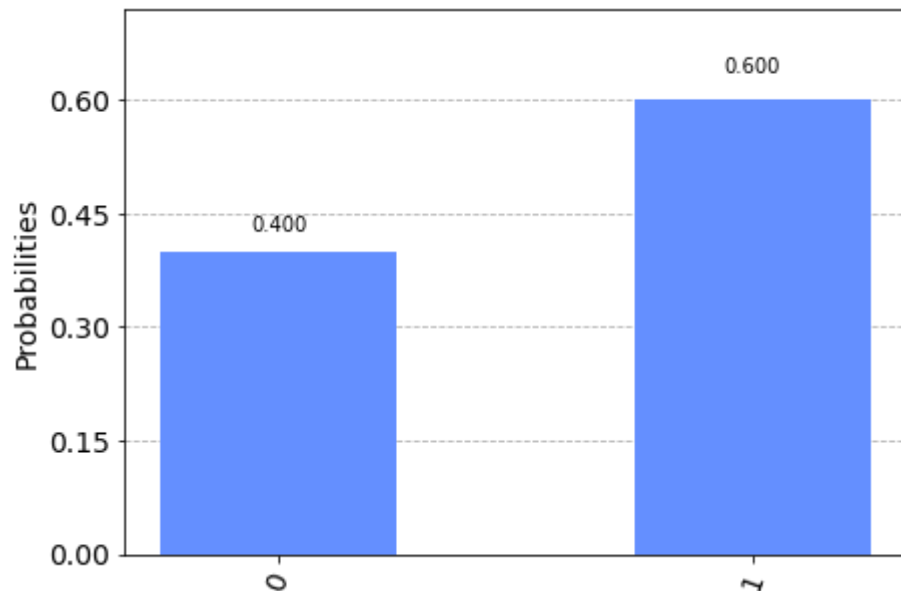


Podemos realizar as 20 medições novamente para vermos se obteremos o mesmo resultado estatístico:

In [174...

```
result = execute(circuit, backend = qasm_simulator, shots = 20).result()
counts = result.get_counts()
plot_histogram(counts)
```

Out[174...



Como podemos ver, o nosso resultado estatístico não é muito confiável, o que nos leva a pensar em como melhorar isso. Apresento uma solução abaixo na questão 2.

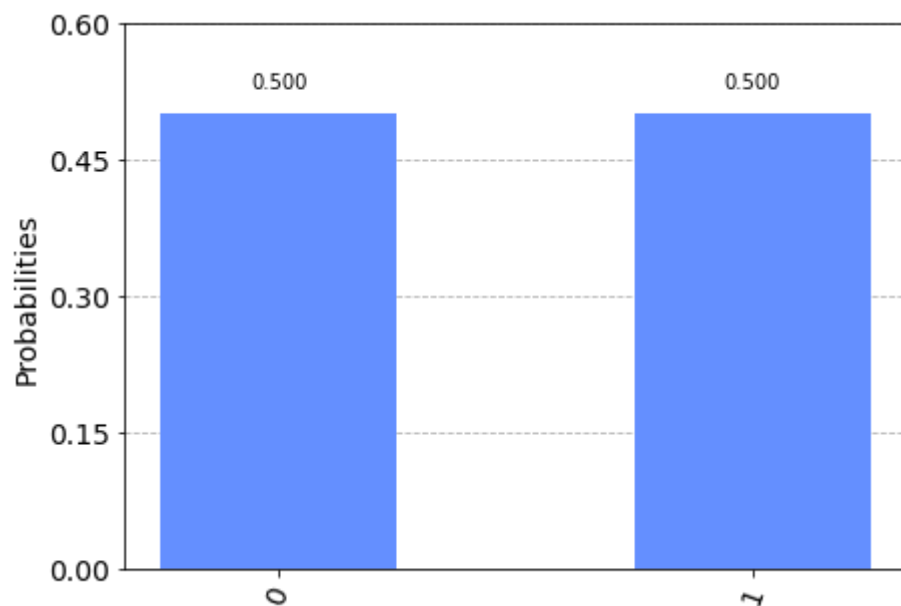
## QUESTÃO 2)

A minha solução para gerar o nosso resultado estatístico de forma mais eficiente, seria realizando um número bem maior de medidas, pois, quanto mais amostras, mais perto estaremos de um resultado mais confiável na estatística:

In [168...

```
result = execute(circuit, backend = qasm_simulator, shots = 1000000).result()
counts = result.get_counts()
plot_histogram(counts)
```

Out[168...



Foram realizadas 1 milhão de medições (máximo permitido pelo simulador).

Como podemos ver, agora temos um valor muito mais correto para a estatística, e que é a porcentagem esperada para a porta de superposição (50% para medidas do estado 0 e 50% para medidas do estado 1).

Podemos repetir a mesma quantidade de medições novamente, apenas para vermos que obtemos um resultado muito próximo ou igual ao acima:

In [169...

```
result = execute(circuit, backend = qasm_simulator, shots = 1000000).result()
counts = result.get_counts()
plot_histogram(counts)
```

Out[169...

