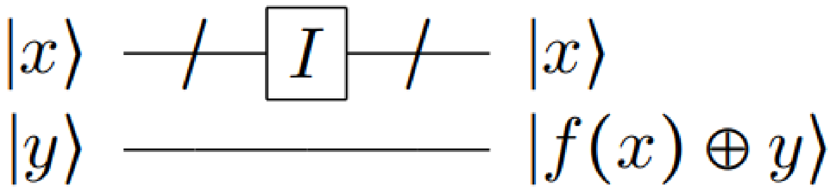
Lab 3 Report

Quantum Computing Algorithms

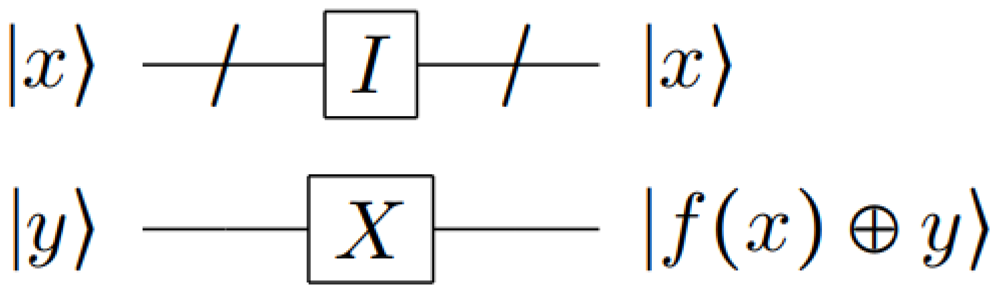
Author: B08901049 Yuan-Chia Chang
Instructor: Professor Hao-Chung Cheng
TA: Chia-Yi Chou
Created: 11/3/2021(Wed)
Last edited: 11/3/2021(Wed)
Contact: b08901049@ntu.edu.tw
Collaborator: B08901002 Chen-Han Lin, B08901209 Yu-Hsiang Lin

Q1.
(a)



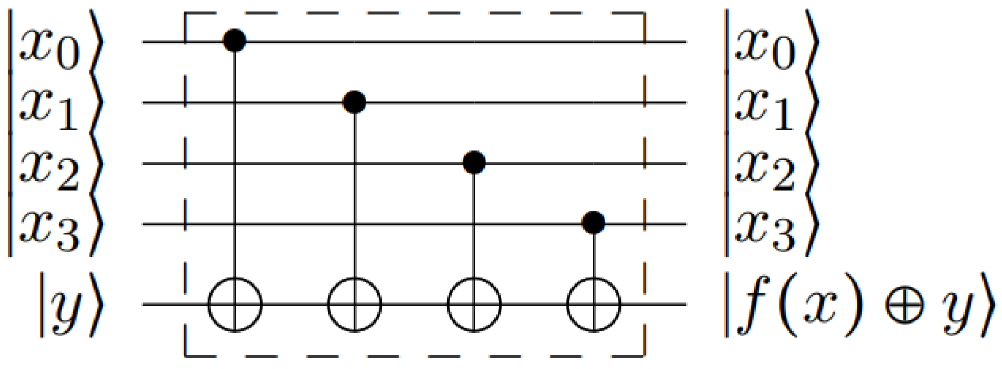
x0	x1	x2	x3	y	result
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	1	0	0

0	1	1	1	1	0
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	0
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0



x0	x1	x2	x3	y	result
0	0	0	0	0	1
0	0	0	0	1	1
0	0	0	1	0	1
0	0	0	1	1	1
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	1
0	0	1	1	1	1
0	1	0	0	0	1

0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
0	1	1	0	1	1
0	1	1	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	1	1
1	0	0	1	0	1
1	0	0	1	1	1
1	0	1	0	0	1
1	0	1	0	1	1
1	0	1	1	0	1
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	1
1	1	1	0	1	1
1	1	1	1	0	1
1	1	1	1	1	1

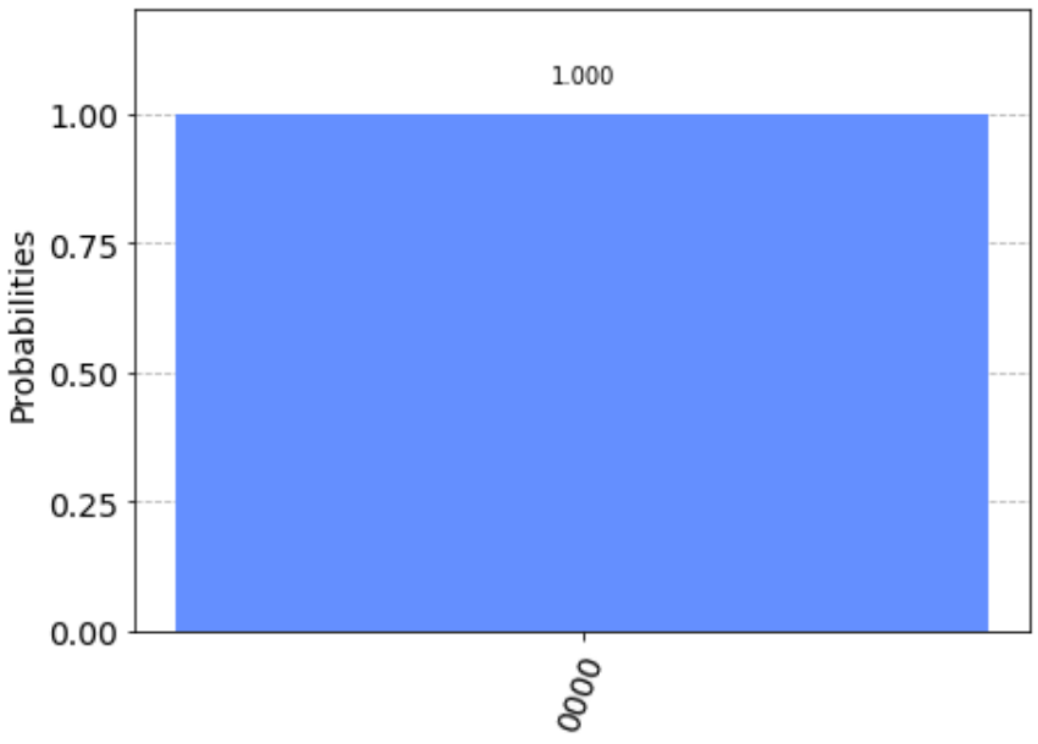


x0	x1	x2	x3	y	result
0	0	0	0	0	0

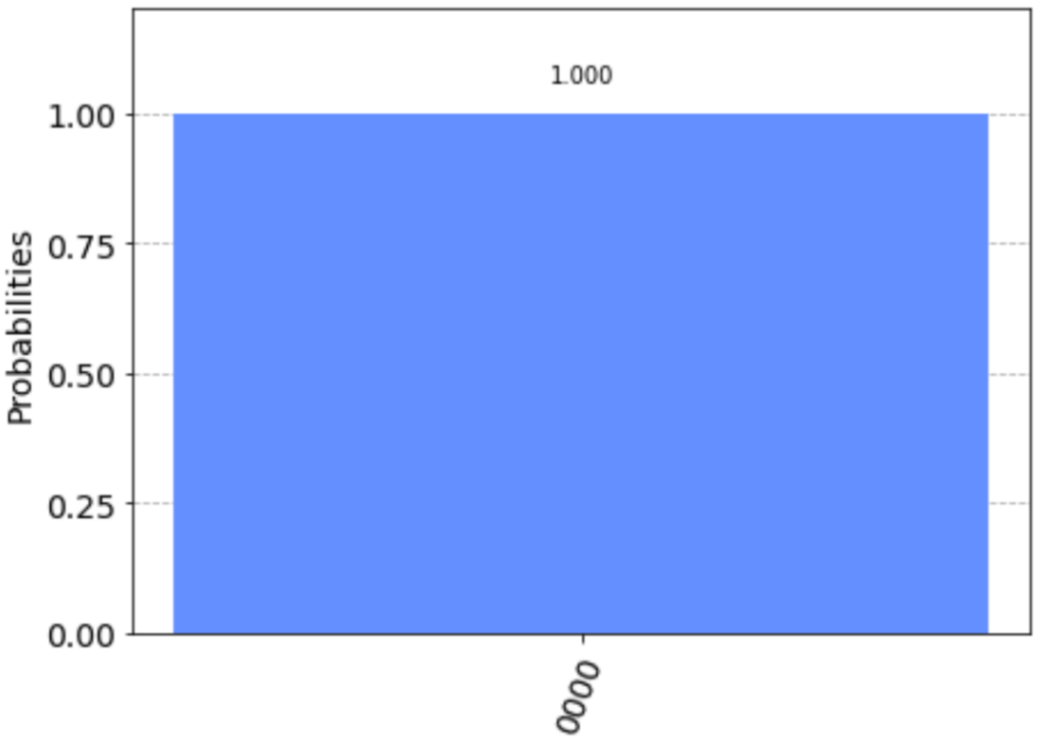
0	0	0	0	1	0
0	0	0	1	0	1
0	0	0	1	1	1
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	1
0	1	0	0	1	1
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	1	1
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	1
1	1	1	0	1	1
1	1	1	1	0	0
1	1	1	1	1	0

(b)

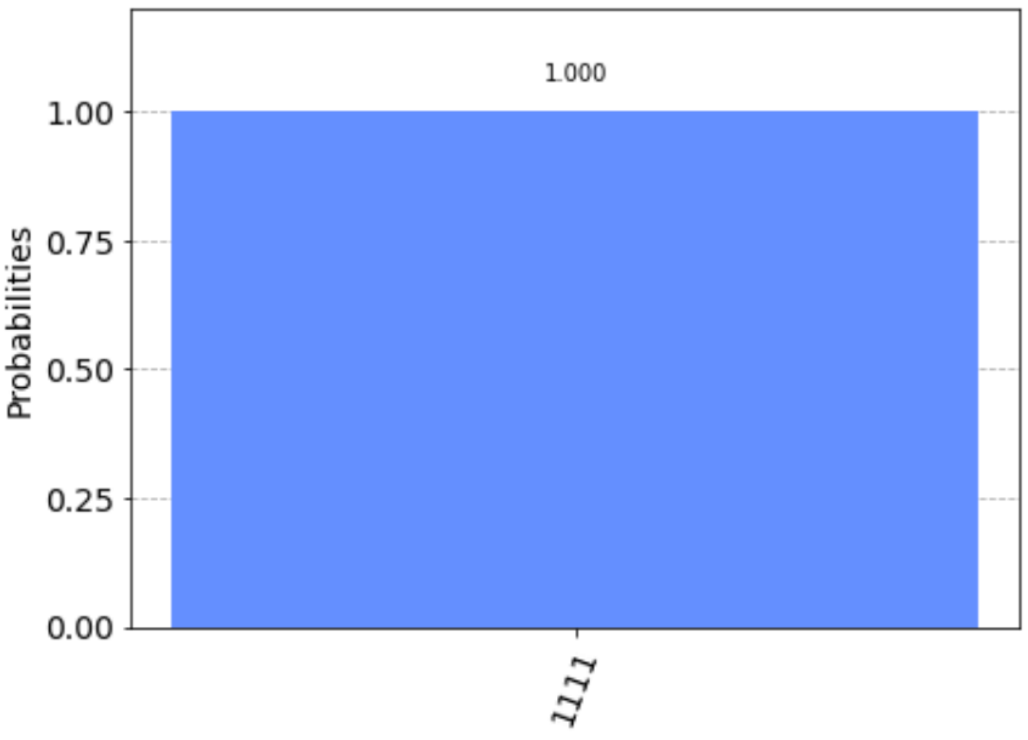
All 0



All 1



Balanced



Q2

(a)

Before the gate [y,x2,x1,x0]

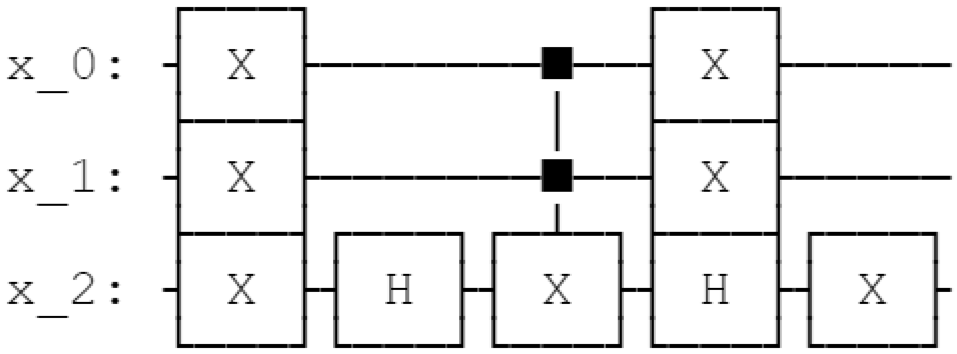
```
[ 0.08767274-0.30224868j -0.09285521-0.07818406j -0.30971463-0.04088062j
 -0.05439344-0.16916707j -0.07445979+0.12033078j -0.04777409-0.0877172j
  0.05042443-0.01429904j  0.47082412+0.05129328j -0.08767274+0.30224868j
  0.09285521+0.07818406j  0.30971463+0.04088062j  0.05439344+0.16916707j
  0.07445979-0.12033078j  0.04777409+0.0877172j  -0.05042443+0.01429904j
 -0.47082412-0.05129328j]
```

After the gate [y,x2,x1,x0]

```
[ 0.08767274-0.30224868j -0.09285521-0.07818406j -0.30971463-0.04088062j
 -0.05439344-0.16916707j -0.07445979+0.12033078j -0.04777409-0.0877172j
 -0.05042443+0.01429904j  0.47082412+0.05129328j -0.08767274+0.30224868j
  0.09285521+0.07818406j  0.30971463+0.04088062j  0.05439344+0.16916707j
  0.07445979-0.12033078j  0.04777409+0.0877172j   0.05042443-0.01429904j
 -0.47082412-0.05129328j]
```

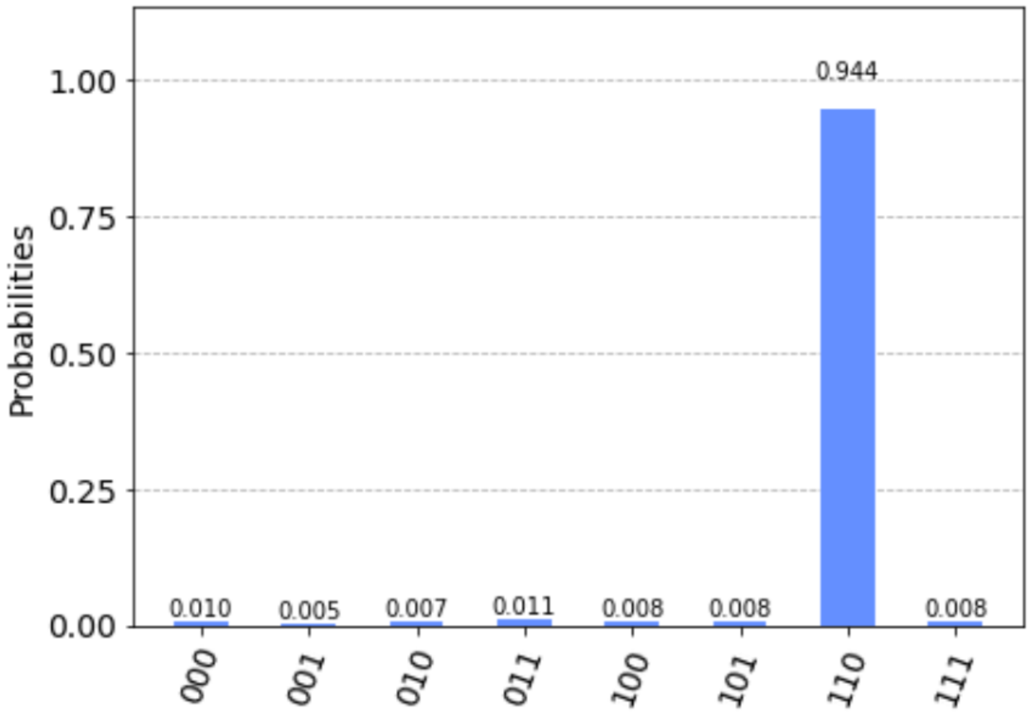
We find that number 6 and 14 number are the minus sign, which corresponds to [x2,x1,x0] = [1,1,0], which is [x0,x1,x2] = [0,1,1].

(b)

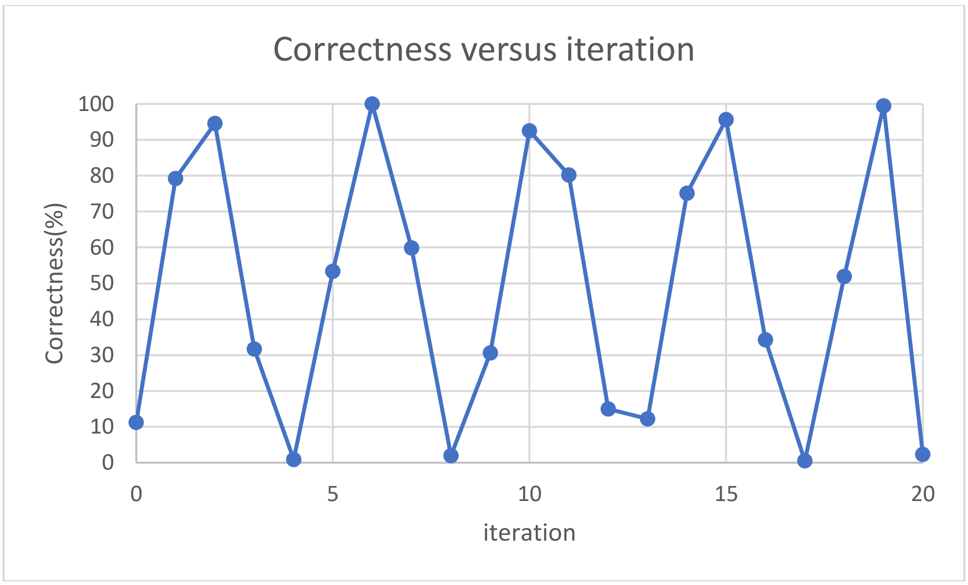


(c)

After $\sqrt{N} = \sqrt{8} \approx 2$ iterations, the result is shown below

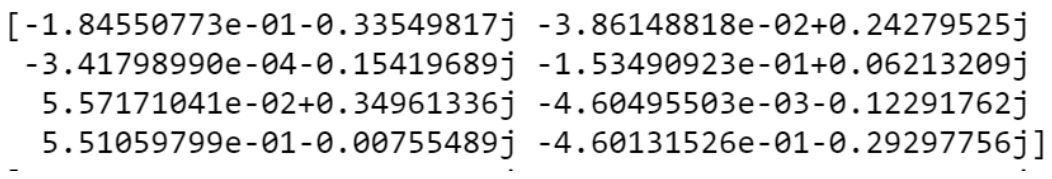


Which we get 011 with high probability, but not always.



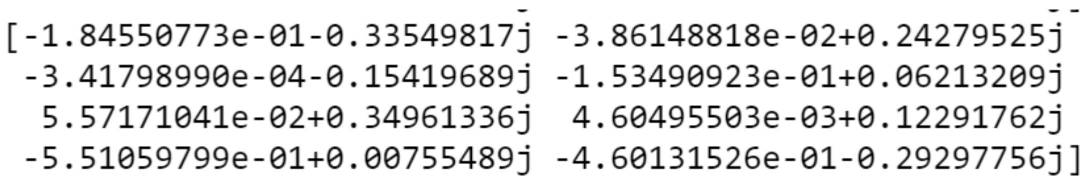
(d)

Before the gate [x2,x1,x0]



```
[-1.84550773e-01-0.33549817j -3.86148818e-02+0.24279525j
  -3.41798990e-04-0.15419689j -1.53490923e-01+0.06213209j
   5.57171041e-02+0.34961336j -4.60495503e-03-0.12291762j
   5.51059799e-01-0.00755489j -4.60131526e-01-0.29297756j]
```

After the gate [x2,x1,x0]

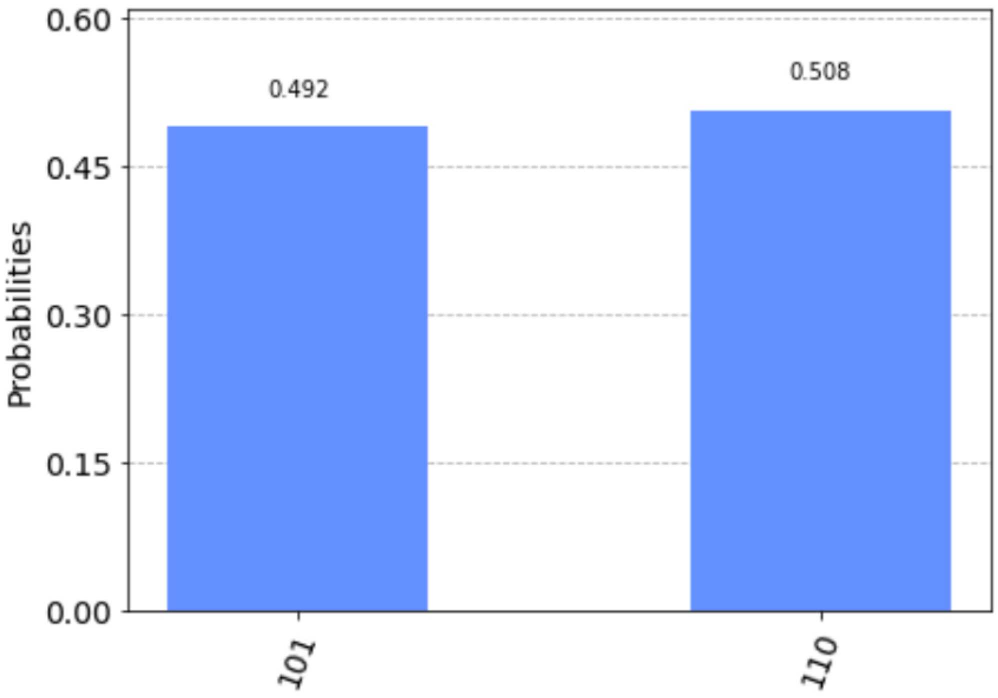


```
[-1.84550773e-01-0.33549817j -3.86148818e-02+0.24279525j
 -3.41798990e-04-0.15419689j -1.53490923e-01+0.06213209j
  5.57171041e-02+0.34961336j  4.60495503e-03+0.12291762j
 -5.51059799e-01+0.00755489j -4.60131526e-01-0.29297756j]
```

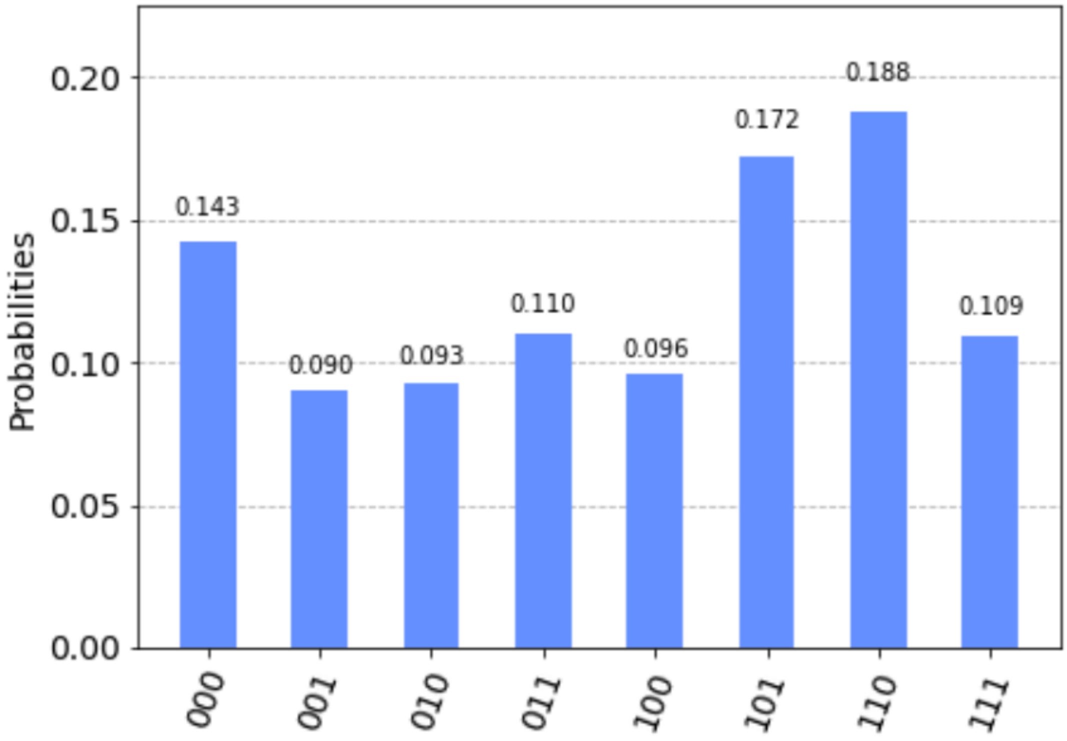
We find that number 5 and 6 number are the minus sign, which corresponds to [x2,x1,x0] = [1,0,1] and [1,1,0], which is [x0,x1,x2] = [1,0,1] and [0,1,1].

(e)

On qasm_simulator, we need 4 iterations of queries.

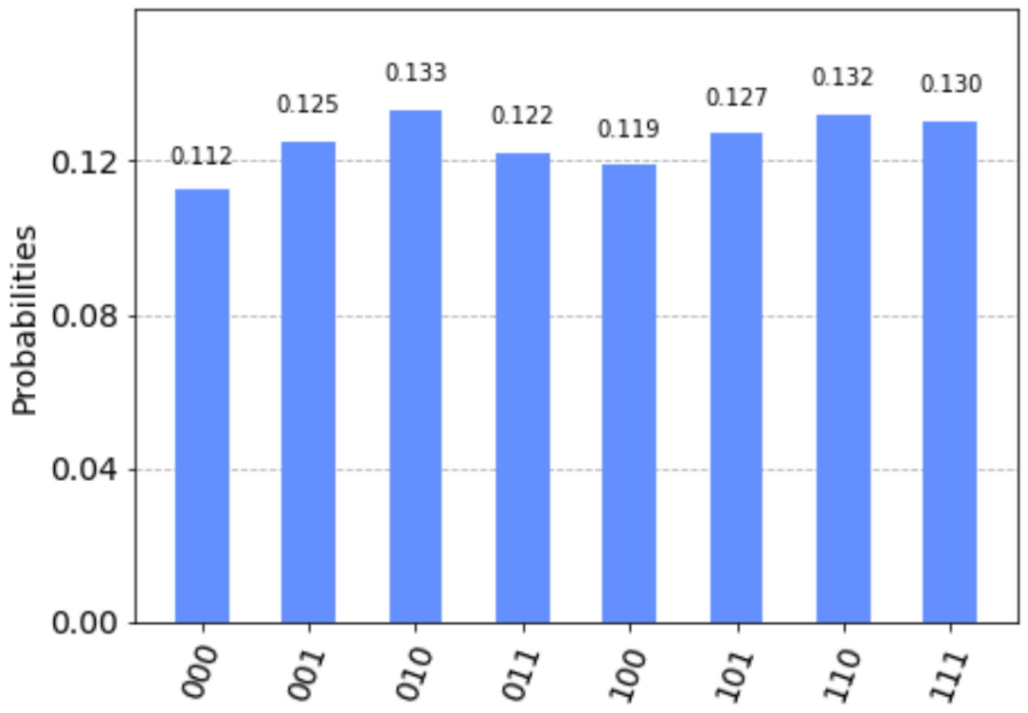


On real device,



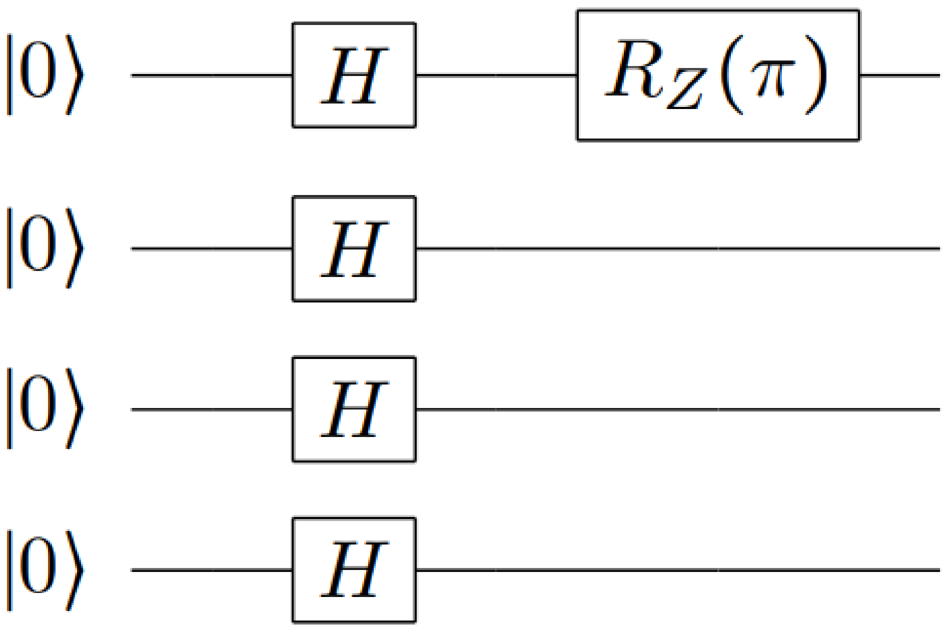
We get the highest probability on 101 and 011, but the error is really huge.

(f)

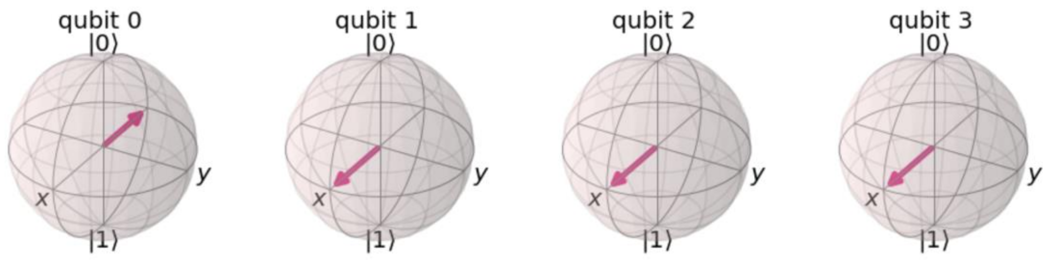


We will always get the same probability of all qubit states. The reason is we flip the half amount of qubits rather than one, which cause nothing difference. In other words, Grover search cannot search too much targets.

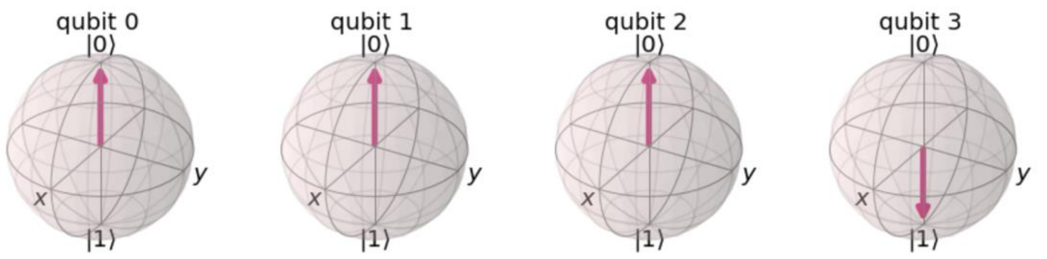
Q3.



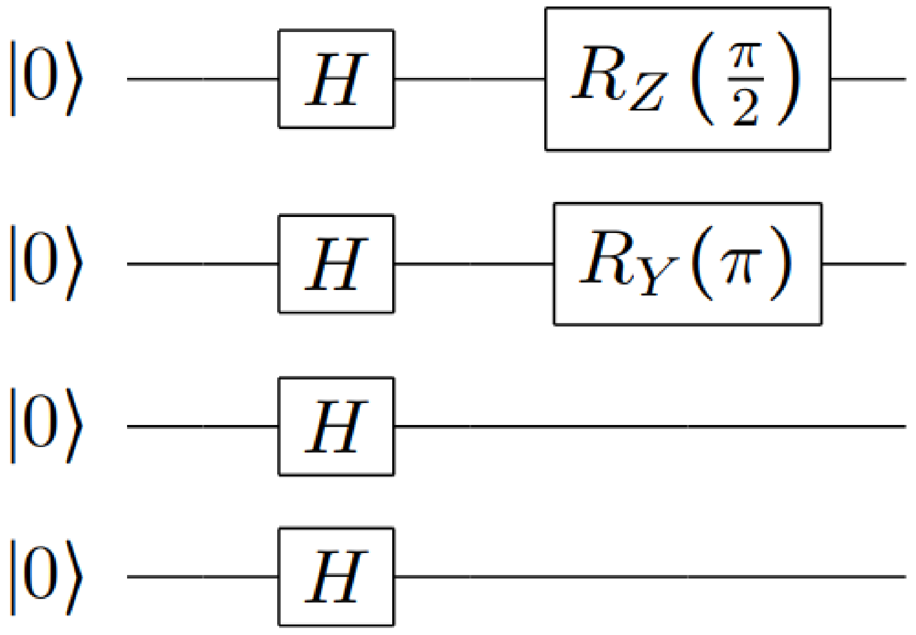
State in Fourier Basis



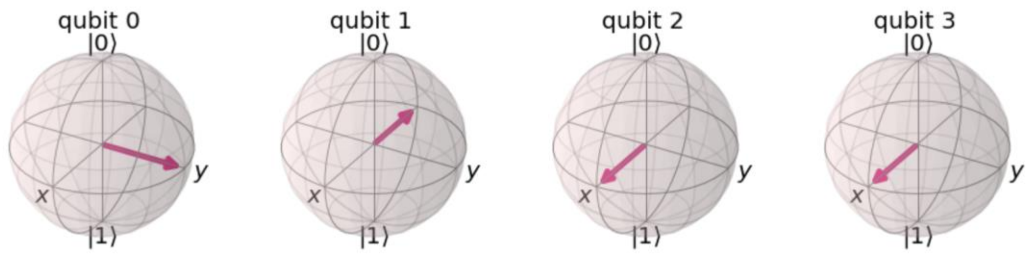
State in Computational Basis



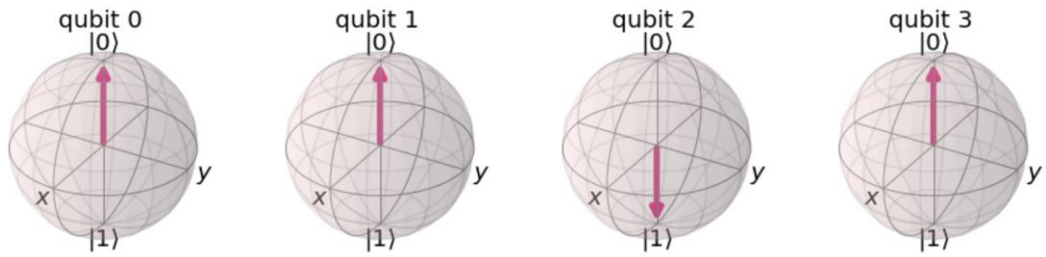
For the Fourier basis, we find the qubit 0 rotate to -x, also the qubit 0 rotate $\frac{\pi}{8}$ each flip in computational basis. Hence, there are 8 flips in computational basis. The representation <q3,q2,q1,q0> will be 1000. Hence <q0,q1,q2,q3> will be 0001.



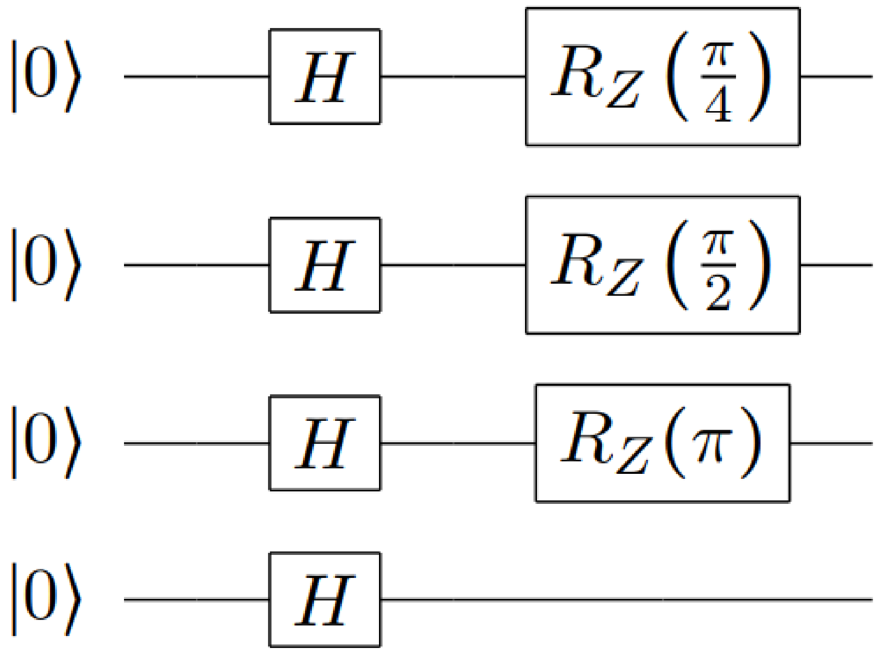
State in Fourier Basis



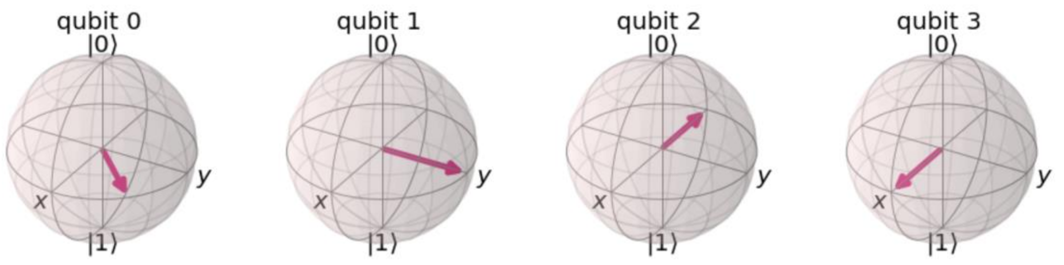
State in Computational Basis



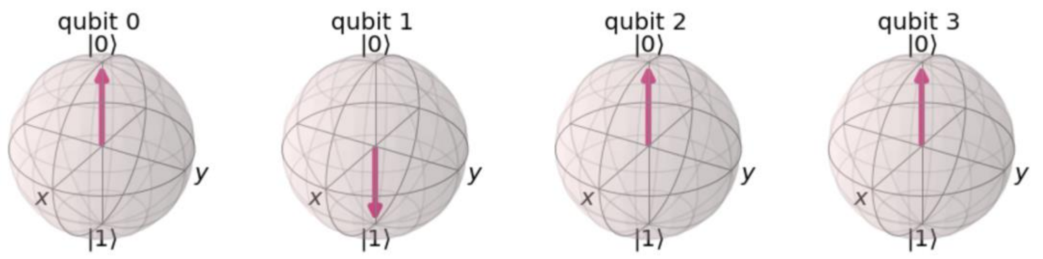
For the Fourier basis, we find the qubit 0 rotate to y, also the qubit 0 rotate $\frac{\pi}{8}$ each flip in computational basis. Hence, there are 4 flips in computational basis. The representation <q3,q2,q1,q0> will be 0100. Hence <q0,q1,q2,q3> will be 0010.



State in Fourier Basis



State in Computational Basis



For the Fourier basis, we find the qubit 0 rotate exactly between x and y, also the qubit 0 rotate $\frac{\pi}{8}$ each flip in computational basis. Hence, there are 2 flips in computational basis. The representation <q3,q2,q1,q0> will be 0010. Hence <q0,q1,q2,q3> will be 0100.

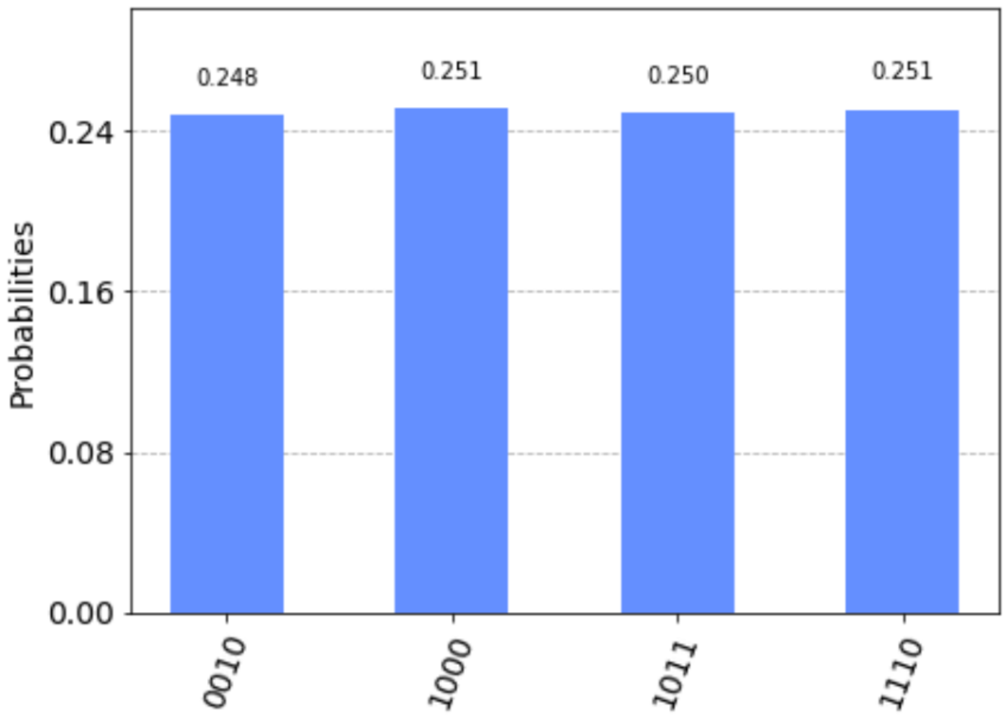
Q4.

To implement this. Let us consider the following example: $N = 2^n = 2^8$, $M = 2^4$, and $f(x) := a^x$ mod 15 with $a = 7$. Show that f is periodic (using a calculator or `Matlab`).

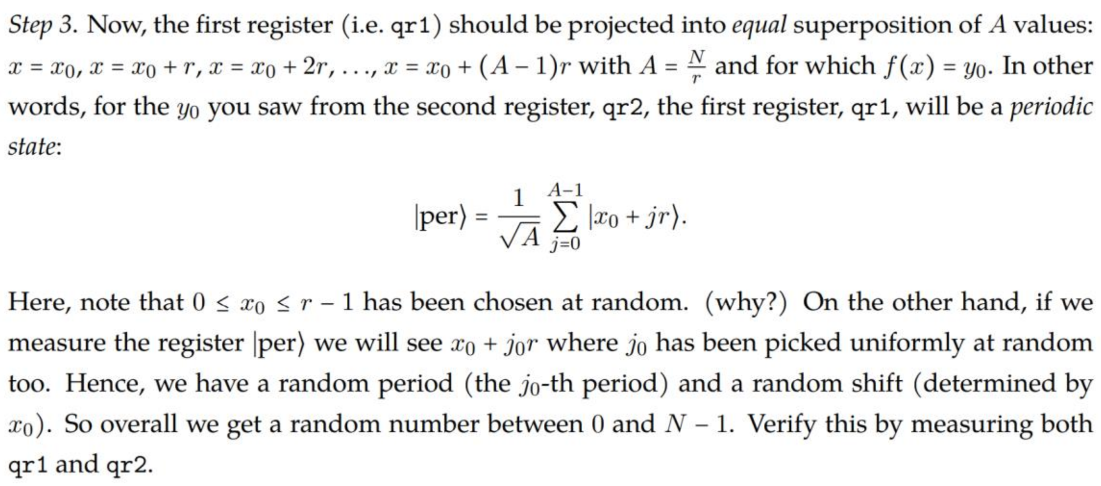
f(1) = 7, f(2) = 4, f(3) = 13, f(4) = 1, f(5) = 7... Hence f is periodic with period 4.

Step 2. Measure the second register (i.e. qr2). Use `plot_histogram()` (and `qasm_simulator`) to see what you get (in decimal form). How likely are them? Why is that? (You can answer

this in the end when you completely understand the whole procedure.) Ideally, you should see some $y_0 = f(x_0)$ for some x_0 is the least of x having $f(x) = y_0$.



We get 2,8,11,14 as equal probability 0.25. It is easily find that these numbers are twice modulus of the power of 7 mod 15(7,4,13,1).(for 11, the half modulus 11 mod 15 is (11+15)/2=13). The reason why these numbers are twice modulus of the power of 7 mod 15 is still a mystery for me.



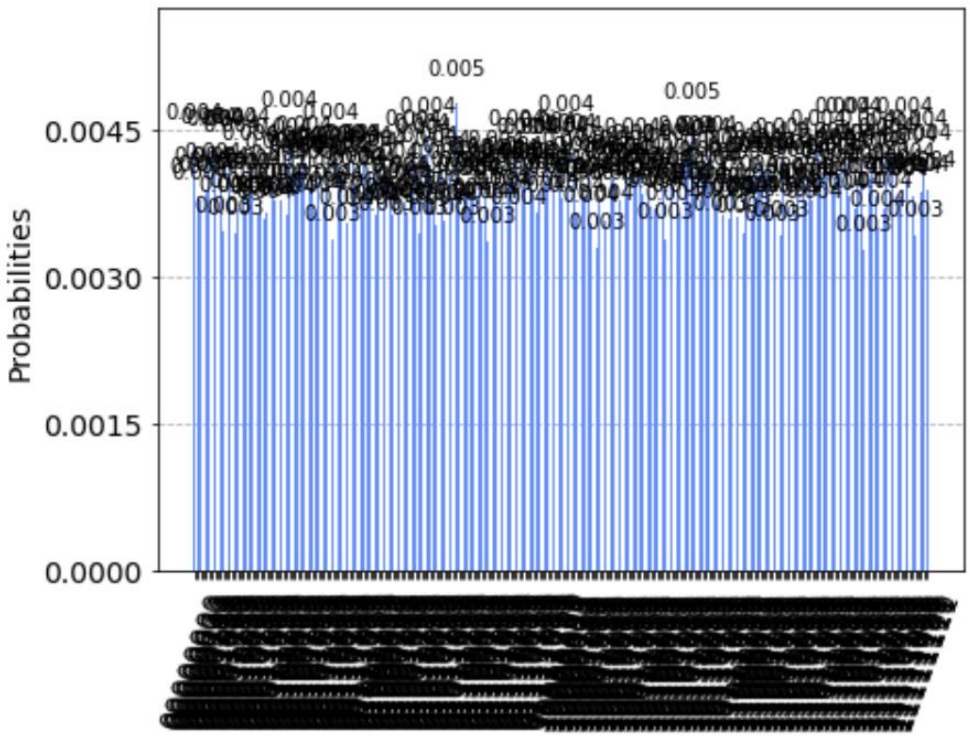
Step 3. Now, the first register (i.e. `qr1`) should be projected into *equal* superposition of A values: $x = x_0$, $x = x_0 + r$, $x = x_0 + 2r$, …, $x = x_0 + (A-1)r$ with $A = \frac{N}{r}$ and for which $f(x) = y_0$. In other words, for the y_0 you saw from the second register, `qr2`, the first register, `qr1`, will be a *periodic state*:

$$|\text{per}\rangle = \frac{1}{\sqrt{A}} \sum_{j=0}^{A-1} |x_0 + jr\rangle.$$

Here, note that $0 \le x_0 \le r-1$ has been chosen at random. (why?) On the other hand, if we measure the register $|\text{per}\rangle$ we will see $x_0 + j_0 r$ where j_0 has been picked uniformly at random too. Hence, we have a random period (the j_0-th period) and a random shift (determined by x_0). So overall we get a random number between 0 and $N-1$. Verify this by measuring both `qr1` and `qr2`.

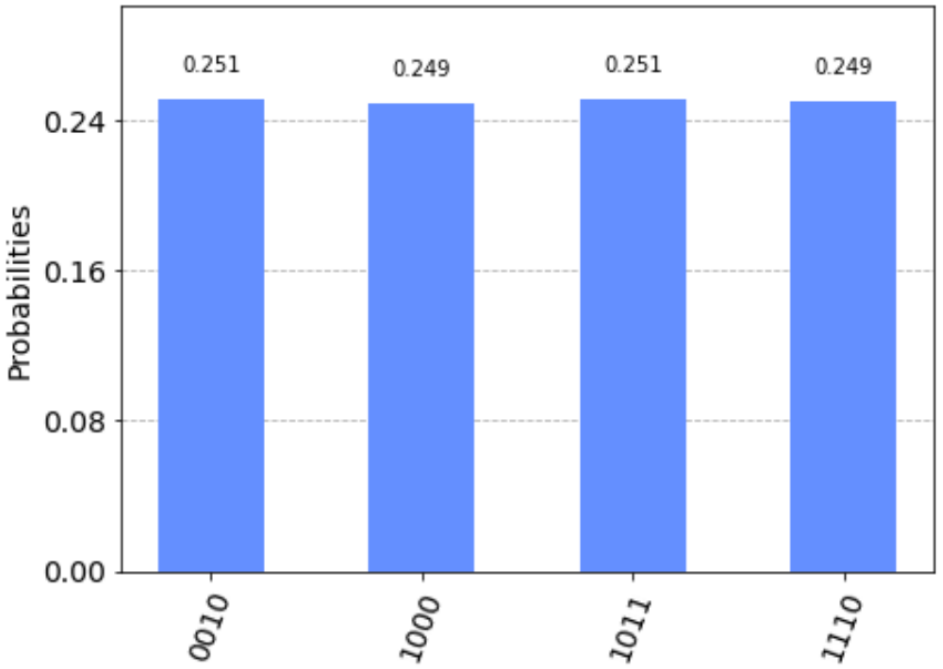
If we didn't randomize chosen x0, x0+jr may lead all the state be the same. And the per state may be A times the same state, which is not preferred.

qr1

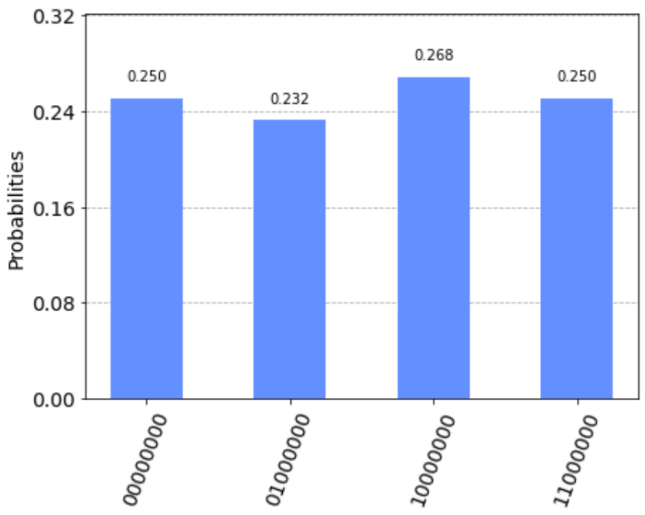


Uniform distribution between 0 to N-1

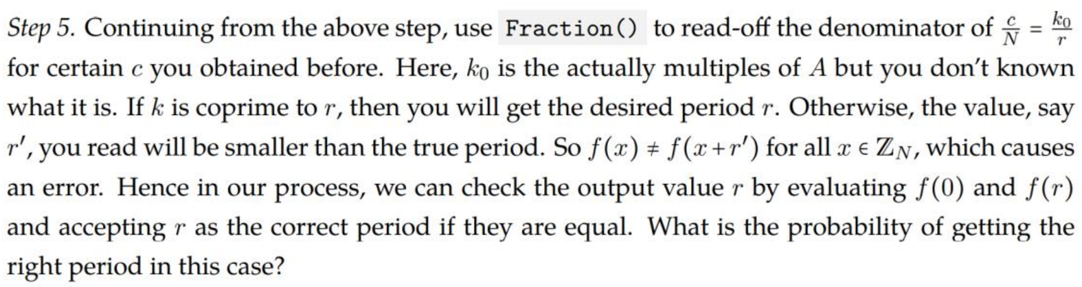
qr2



Step 4. Apply $\mathbf{QFT}_N^\dagger$ on $|\text{per}\rangle$. Show that all terms have zero amplitude except for the multiples of $A = \frac{N}{r}$, i.e. you should get $c = k \cdot A$ for $k = 0, 1, 2, \ldots, r-1$. What did you really get (in decimal form)? (You may need the following sample codes.)



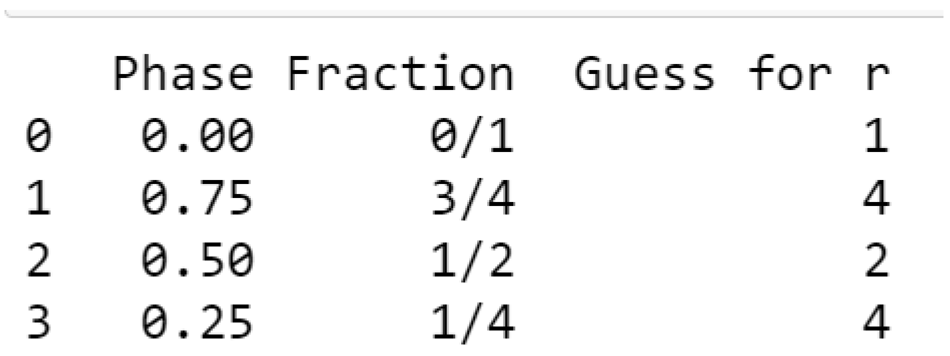
N = 256, r = 4, A = 256/4 = 64. We get c = 0,64,128,192.



Step 5. Continuing from the above step, use `Fraction()` to read-off the denominator of $\frac{c}{N} = \frac{k_0}{r}$ for certain c you obtained before. Here, k_0 is the actually multiples of A but you don't known what it is. If k is coprime to r, then you will get the desired period r. Otherwise, the value, say r', you read will be smaller than the true period. So $f(x) \neq f(x+r')$ for all $x \in \mathbb{Z}_N$, which causes an error. Hence in our process, we can check the output value r by evaluating $f(0)$ and $f(r)$ and accepting r as the correct period if they are equal. What is the probability of getting the right period in this case?

There are 4 cases in a = 7, where two cases represent right period 4. Hence the probability of getting the right period in this case is 50%.

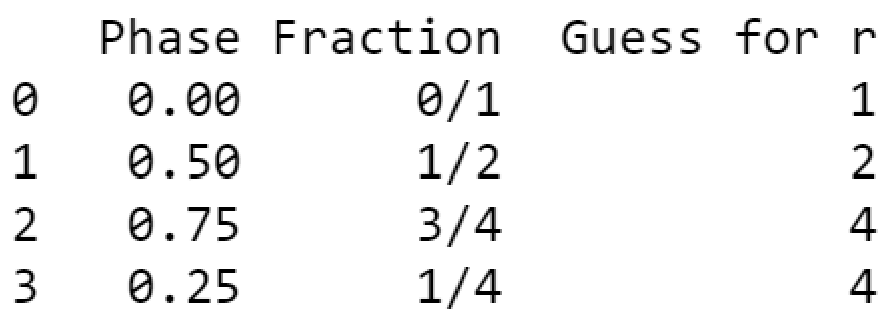
a=7



```
   Phase Fraction  Guess for r
0   0.00      0/1            1
1   0.75      3/4            4
2   0.50      1/2            2
3   0.25      1/4            4
```

After validating, the period is 4.

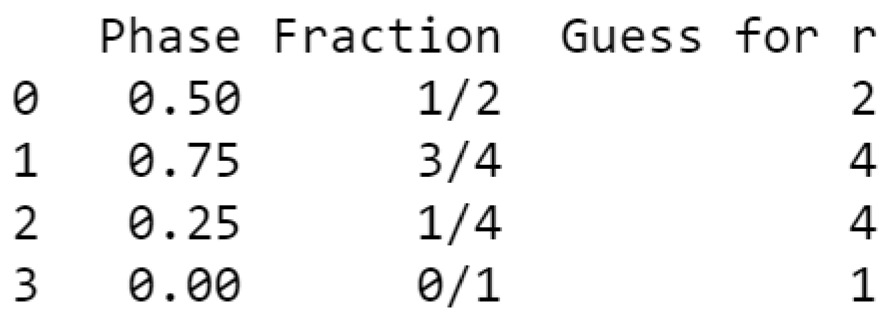
a= 2



```
   Phase Fraction  Guess for r
0   0.00      0/1            1
1   0.50      1/2            2
2   0.75      3/4            4
3   0.25      1/4            4
```

After validating, the period is 4.

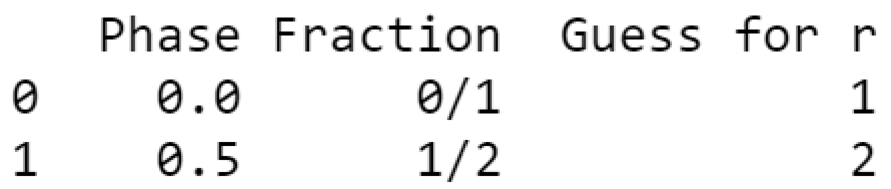
a=8



```
   Phase Fraction  Guess for r
0   0.50      1/2            2
1   0.75      3/4            4
2   0.25      1/4            4
3   0.00      0/1            1
```

After validating, the period is 4.

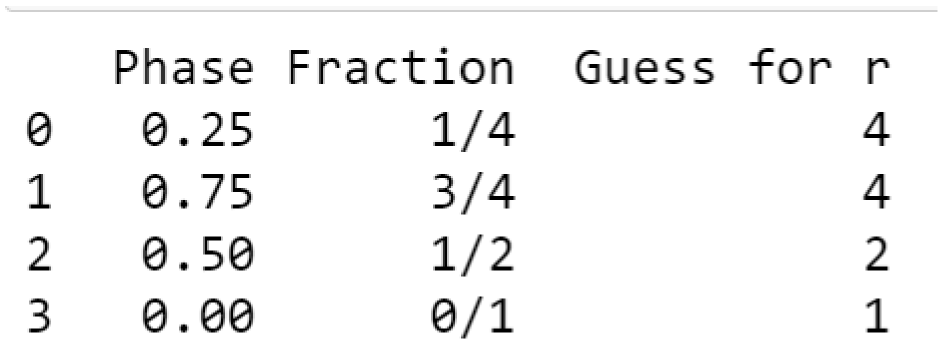
a=11



```
   Phase Fraction  Guess for r
0    0.0      0/1            1
1    0.5      1/2            2
```

After validating, the period is 2.

a=13



```
   Phase  Fraction  Guess for r
0   0.25       1/4            4
1   0.75       3/4            4
2   0.50       1/2            2
3   0.00       0/1            1
```

After validating, the period is 4.

Appendix

Code: https://github.com/yuanchiachang/CommLab/blob/main/Lab3/src

Reference :

[1] https://qiskit.org/textbook/ch-algorithms/deutsch-jozsa.html

[2] https://qiskit.org/textbook/ch-algorithms/grover.html

[3] https://qiskit.org/textbook/ch-algorithms/quantum-fourier-transform.html

[4] https://qiskit.org/textbook/ch-algorithms/shor.html