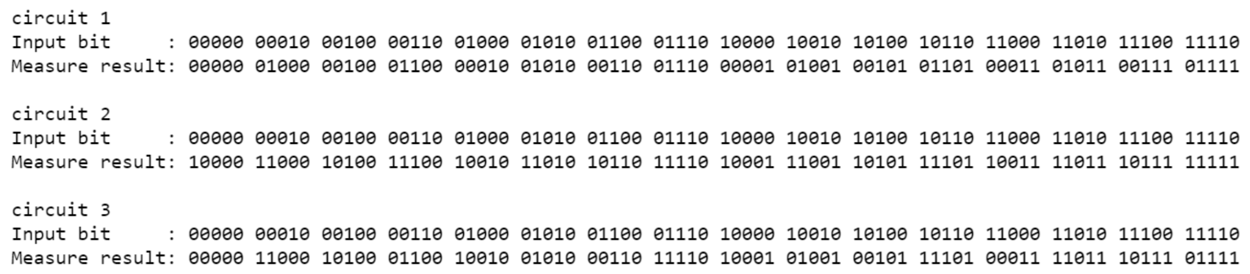
陳志偉 B06901126 電機五

Electrical Engineering Lab（topics on Communication System）
Lab3 Report

1 a) By measuring the corresponding output, we observe that the first bit of circuit 1 is all 0 and the first bit of circuit 2 is all 1. So, quantum oracles 1 and 2 are constant functions. But the first bit of circuit 3 has two results, 1 and 0 which is a balanced function.

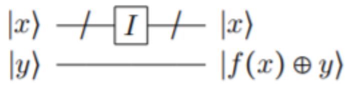


```
circuit 1
Input bit     : 00000 00010 00100 00110 01000 01010 01100 01110 10000 10010 10100 10110 11000 11010 11100 11110
Measure result: 00000 01000 00100 01100 00010 01010 00110 01110 00001 01001 00101 01101 00011 01011 00111 01111

circuit 2
Input bit     : 00000 00010 00100 00110 01000 01010 01100 01110 10000 10010 10100 10110 11000 11010 11100 11110
Measure result: 10000 11000 10100 11100 10010 11010 10110 11110 10001 11001 10101 11101 10011 11011 10111 11111

circuit 3
Input bit     : 00000 00010 00100 00110 01000 01010 01100 01110 10000 10010 10100 10110 11000 11010 11100 11110
Measure result: 00000 11000 10100 01100 10010 01010 00110 11110 10001 01001 00101 11101 00011 11011 10111 01111
```

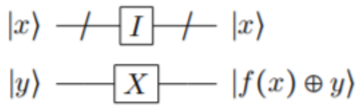
Figure 1a

1 b) By the result showing as *Figure 1b*, we confirm the following conclusions.

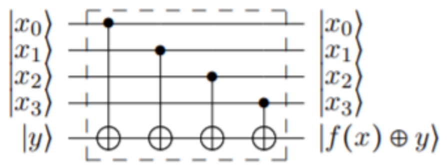
Quantum oracles 1 and 2 are constant functions, quantum oracle 3 is a balanced function.



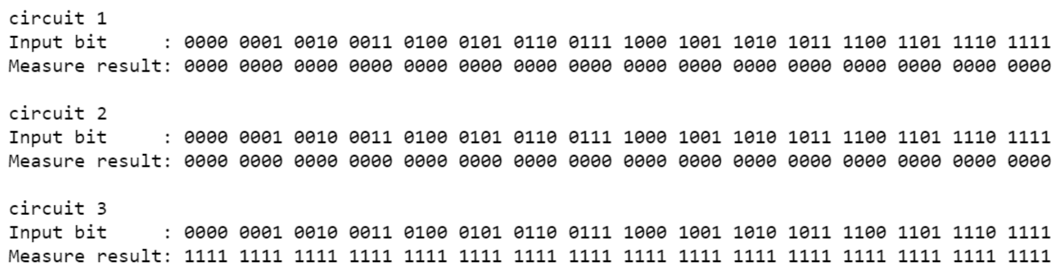
Quantum oracle 1



Quantum oracle 2



Quantum oracle 3



```
circuit 1
Input bit     : 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
Measure result: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

circuit 2
Input bit     : 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
Measure result: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

circuit 3
Input bit     : 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
Measure result: 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111
```

Figure 1b

2 a) Before pass thought U_f we have the arbitrary state show as Figure 2a, after pass though U_f we have the arbitrary state show as Figure 2b.

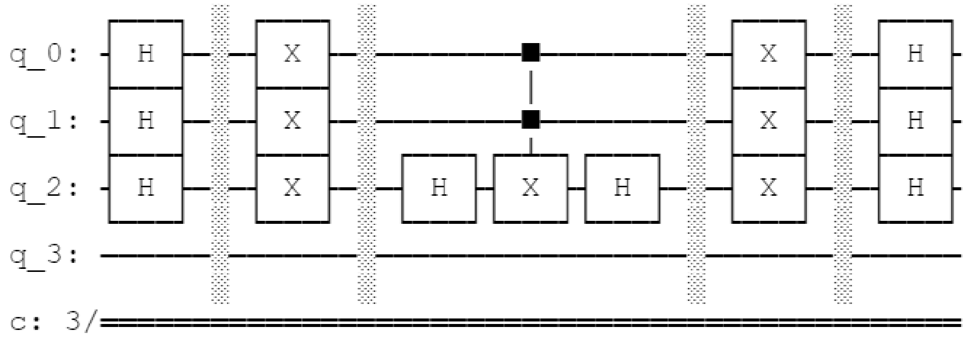
```
[ 0.25+0.000000e+00j  0.25+0.000000e+00j  0.25+0.000000e+00j
  0.25+0.000000e+00j  0.25+0.000000e+00j  0.25+0.000000e+00j
  0.25+0.000000e+00j  0.25+0.000000e+00j -0.25-3.061617e-17j
 -0.25-3.061617e-17j -0.25-3.061617e-17j -0.25-3.061617e-17j
 -0.25-3.061617e-17j -0.25-3.061617e-17j -0.25-3.061617e-17j
 -0.25-3.061617e-17j]
```

Figure 2a

```
[ 0.53033009-9.74200563e-17j -0.1767767 +7.57711549e-17j
  0.1767767 +1.08244507e-17j  0.1767767 -3.24733521e-17j
  0.1767767 +1.08244507e-17j  0.1767767 -3.24733521e-17j
 -0.1767767 +3.24733521e-17j -0.1767767 +3.24733521e-17j
 -0.53033009+1.08244507e-17j  0.1767767 -7.57711549e-17j
 -0.1767767 -1.08244507e-17j -0.1767767 +3.24733521e-17j
 -0.1767767 -1.08244507e-17j -0.1767767 +3.24733521e-17j
  0.1767767 -3.24733521e-17j  0.1767767 -3.24733521e-17j]
```

Figure 2b

2 b)



2 c) After apply $\lfloor\sqrt{N}\rfloor$ times, where N = 8 and $\lfloor\sqrt{N}\rfloor = 2$. We have the result of Figure 2c to measure ‘011’. Figure 2d show the result after apply 20 times.

```
{'110': 1927, '101': 11, '000': 15, '001': 14, '111': 23, '100': 16, '011': 25, '010': 17}
```

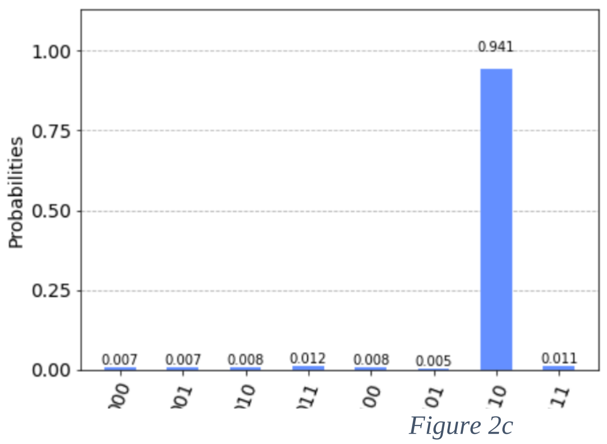


Figure 2c

```
{'110': 1252, '100': 111, '010': 127, '011': 123, '000': 119, '111': 104, '101': 119, '001': 93}
```

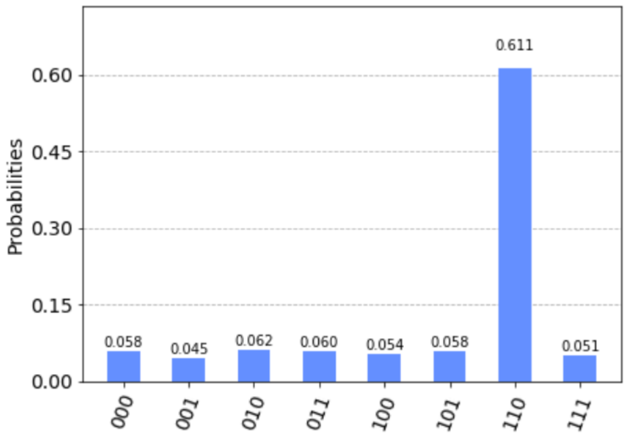


Figure 2d

2 d) By using 'statevector_simulator', we get the following result. The sixth and seventh element of the vector is negative, so we sure that '101' and '110' have been flipped.

```
[ 0.35355339+0.j  0.35355339+0.j  0.35355339+0.j  0.35355339+0.j
  0.35355339+0.j -0.35355339+0.j -0.35355339+0.j  0.35355339-0.j]
```

2 e) We need only one query to solve problem and we have the 50% of probability to get both '011' and '101' (Figure 2e). By using IBM's real device, '101' and '110' also have a higher chance to measure.

```
{'101': 1054, '110': 994}
```

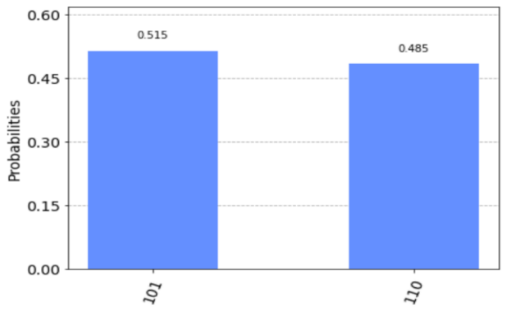


Figure 2e

```
Job Status: job has successfully run
{'000': 5, '001': 41, '010': 33, '011': 21, '100': 43, '101': 432, '110': 431, '111': 18}
```

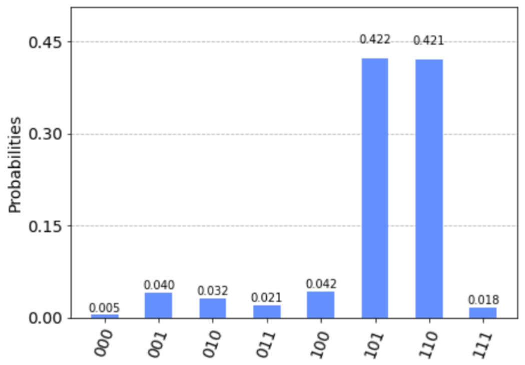


Figure 2f

2 f) From Figure 2g, ‘011’, ‘101’, ‘110’, ‘111’ have been flipped. But after measuring, we have the almost same probability to get all state.

```
[ 0.35355339+0.j  0.35355339+0.j  0.35355339+0.j -0.35355339+0.j
  0.35355339+0.j -0.35355339+0.j -0.35355339+0.j -0.35355339+0.j]
```

Figure 2g

```
{'011': 276, '111': 273, '010': 256, '110': 258, '000': 255, '101': 254, '100': 253, '001': 223}
```

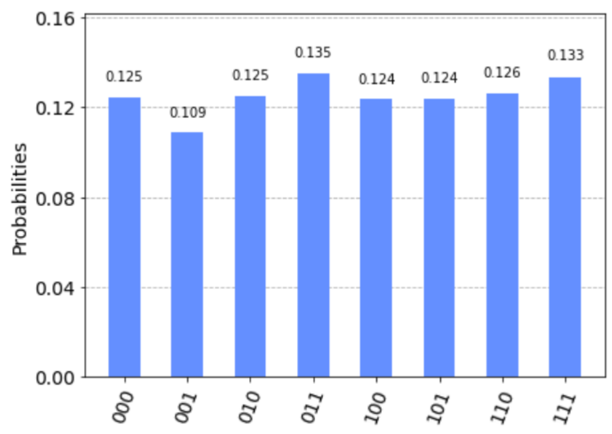
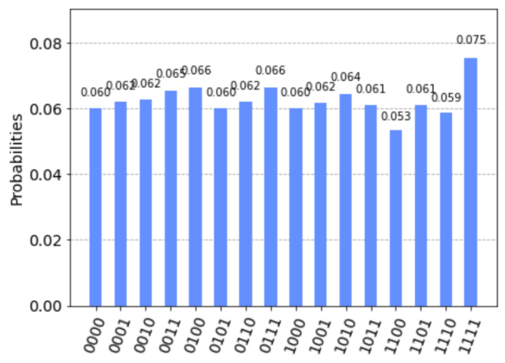


Figure 3

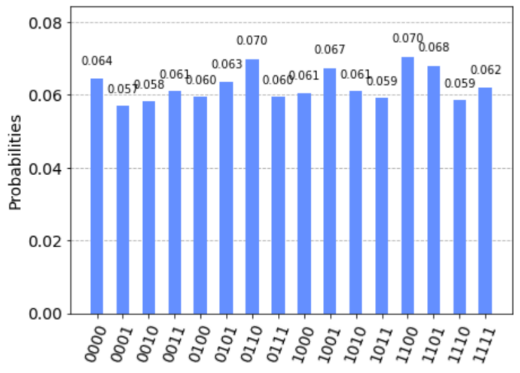
3) The following 3 Figures show the result of directly measuring the 3 circuits.

```
{'0000': 123, '1001': 126, '1010': 132, '0010': 128, '1111': 154, '0111': 136, '1101': 125, '1100': 109, '1011': 125, '010
0': 136, '0001': 127, '1110': 120, '0110': 127, '1000': 123, '0101': 123, '0011': 134}
```



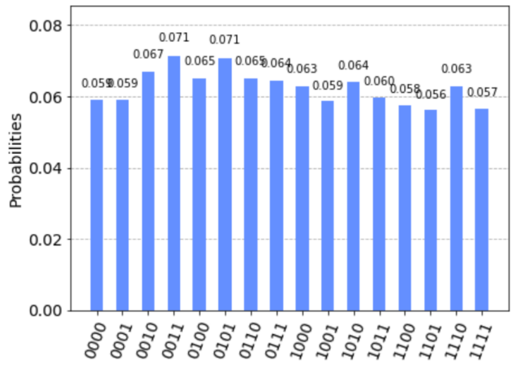
Result of measuring circuit 1

```
{'0001': 117, '1001': 138, '0101': 130, '0110': 143, '1011': 121, '1100': 144, '1000': 124, '0000': 132, '1111': 127, '010
0': 122, '1010': 125, '1101': 139, '0010': 119, '0011': 125, '1110': 120, '0111': 122}
```



Result of measuring circuit 2

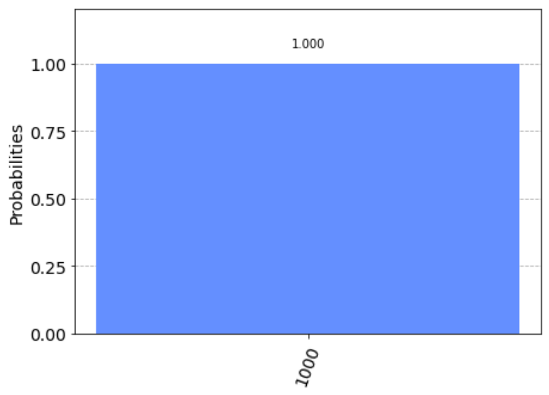
```
{'0011': 146, '1111': 116, '0110': 133, '1110': 129, '0101': 145, '1000': 129, '1011': 122, '1010': 131, '1100': 118, '010
0': 133, '0001': 121, '1101': 115, '0000': 121, '0010': 137, '0111': 132, '1001': 120}
```



Result of measuring circuit 3

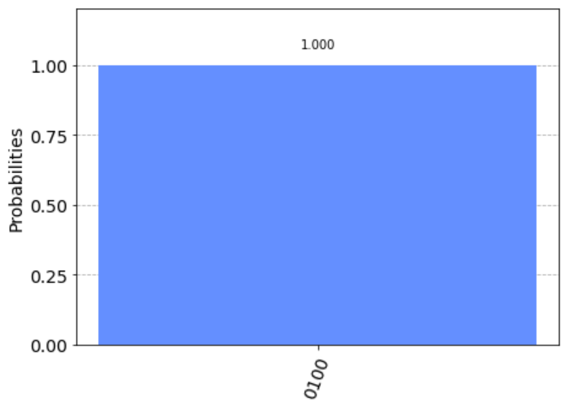
After apply QFT$^{+}{}_{N}$ to 3 circuit, we have the following result.

```
{'1000': 2048}
```



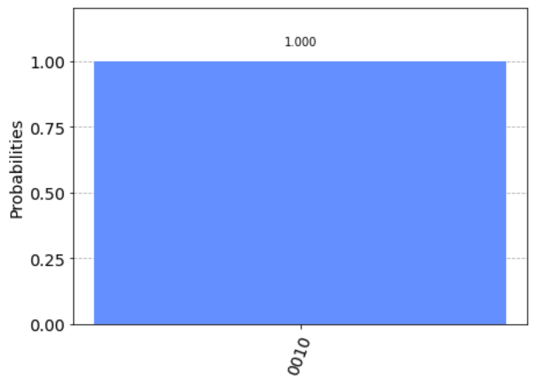
After apply QFT$^{+}{}_{N}$ to circuit 1

```
{'0100': 2048}
```



After apply QFT$^{+}{}_{N}$ to circuit 2

```
{'0010': 2048}
```



After apply QFT$^{+}{}_{N}$ to circuit 3

4) Step 1: To show that f(x) $= a^x mod\ 15\ \ where\ a = 7$ is periodic, I write a python code and plot the result. From the result of Figure 4a, show that f(x) is periodic.

```
def f(a, x):
    return a**x % 15

a = 7

y = []
for i in range(25):
    y.append(f(a, i))

plt.plot(y, marker = '*')
plt.xticks(range(0, 25))
plt.xlabel('x')
plt.ylabel('a$^x$ mod 15')
plt.show()
```

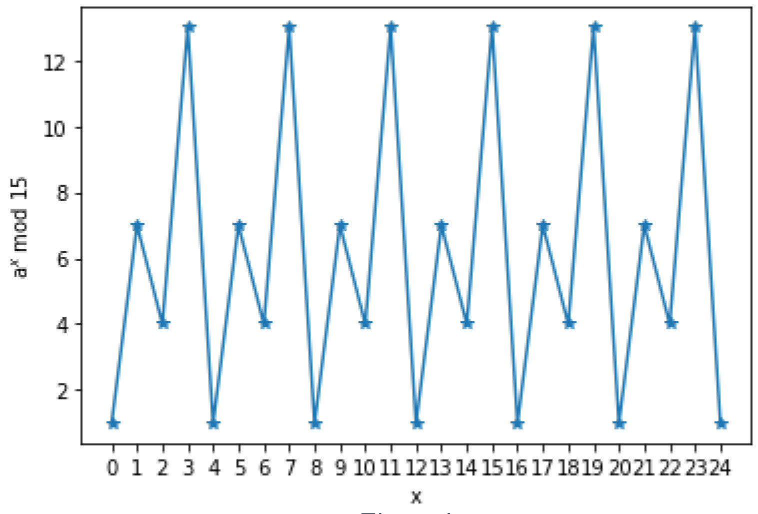


Figure 4a

Step 2: After measuring the second register, I get the result as Figure 4b match with ‘4’, ‘1’, ‘13’, ‘7’ in decimal. It actually same as the Figure 4a.

```
{'1011 00000000': 518, '0010 00000000': 507, '1000 00000000': 516, '1110 000000
00': 507}
```

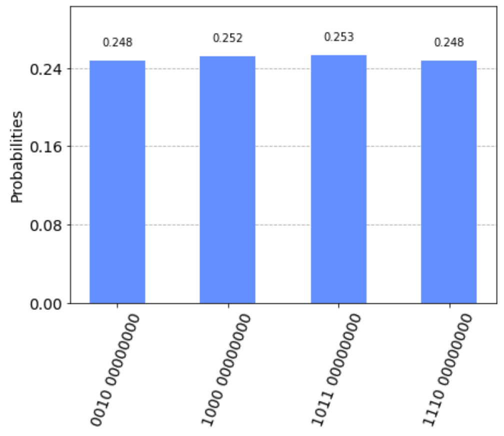


Figure 4b

Step 3: Figure 4c show the result of measuring both qr1 and qr2. There are 256 elements in the result, which is 2^8.

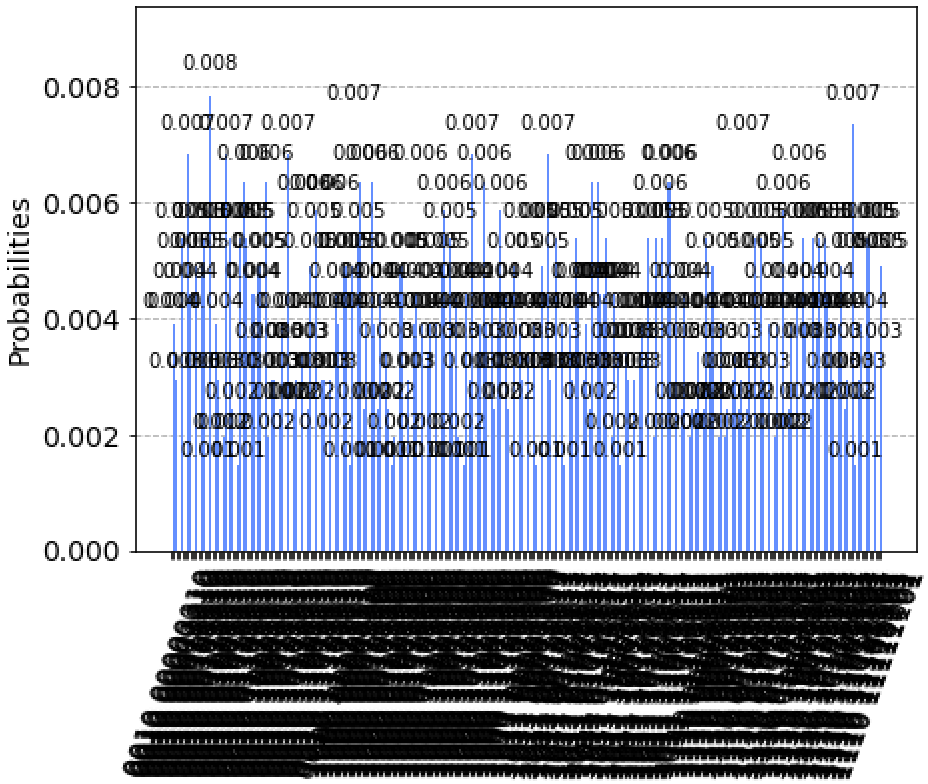


Figure 4c

Step 4: Figure 4d show the measure result after apply QFT^{+}_{N} and the given sample code.

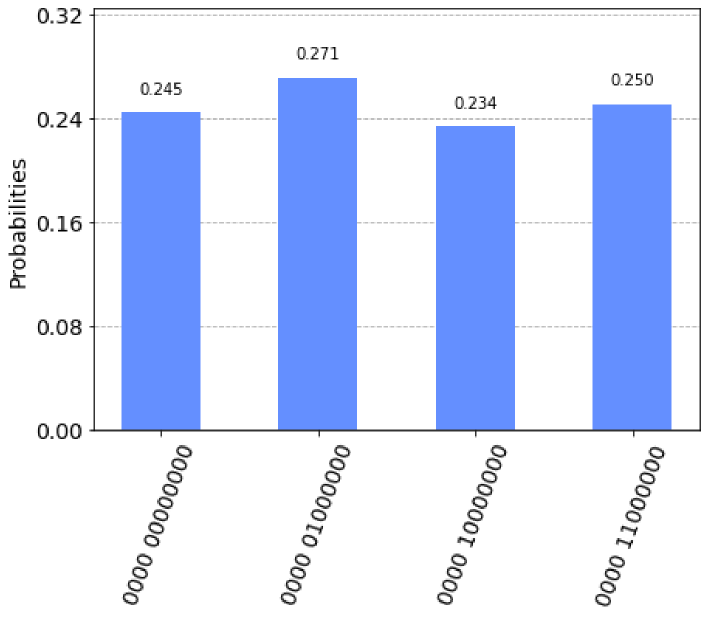
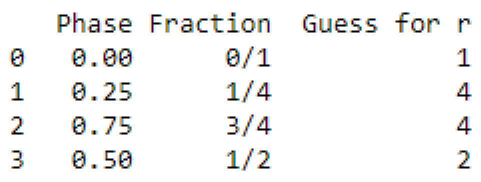


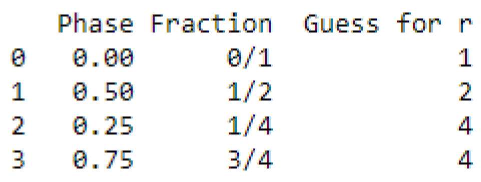
Figure 4d

Step 5:



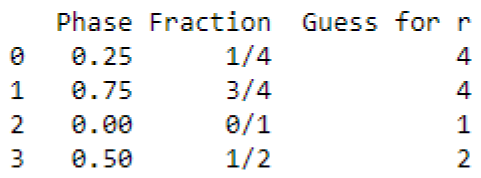
```
   Phase Fraction  Guess for r
0   0.00      0/1            1
1   0.25      1/4            4
2   0.75      3/4            4
3   0.50      1/2            2
```

a = 7



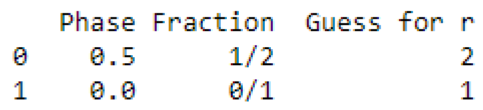
```
   Phase Fraction  Guess for r
0   0.00      0/1            1
1   0.50      1/2            2
2   0.25      1/4            4
3   0.75      3/4            4
```

a = 2



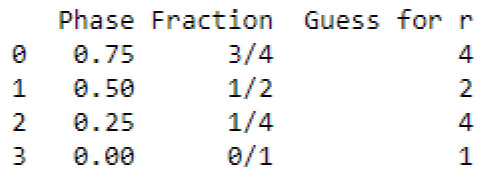
```
   Phase Fraction  Guess for r
0   0.25      1/4            4
1   0.75      3/4            4
2   0.00      0/1            1
3   0.50      1/2            2
```

a = 8



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

a = 11



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

a = 13

Appendix

All my code will update to my github's repo: https://github.com/finalwee/CommLab