Quantum Computing Results

Dirac Notation:

```
myState2=[
  (numpy.sqrt(0.1)*1.j, '101'),
  (numpy.sqrt(0.5), '000') ,
  (-numpy.sqrt(0.4), '010' )
]
PrettyPrintBinary(myState2)
PrettyPrintInteger(myState2)
```

Paste the result of running your code on the above output:

```
print(DiracToVec(myState2))
print(VecToDirac(DiracToVec(myState2)))
```

Paste the result of running your code on the above output:

Quantum Simulator S

Paste the result from **rand.circuit** for Simulator S

Quantum Simulator M

My simulator II results for the three circuit tests (should agree with previous results):

- ❏ Check that simulator M-b gives the same results as Ia for the example.circuit:
- ❏ Check that simulator M-c gives the same results as S for the example.circuit (extra credit)
- ❏ Check that simulator M-d gives the same results as S for the example.circuit (extra credit)

Paste the output of your time and RAM tests for the simulators you have

Paste the histogram from doing **measure.circuit** for simulator Ia

Paste the output from **input.circuit** for Simulator Ia

Non-Atomic Gates

Circuit Description for Not:

Result of Not on |1>:

Circuit Description for Rz:

Circuit Description for (short-range) Control-Rz:

Circuit Description for (short-range) Control-Phase:

Circuit Description for Swap(2,5) (using short-range gates):

```
9
H 0
CPHASE 0 5 0.3
P 1 0.3
CNOT 4 7
SWAP 2 8
```

Result of running your circuit (after precompilation) on the above input:

Phase Estimation

Run your simulator for 100 evenly chosen values of ϕ between 0 and 2pi and make the following graph: On the x-axis put $\phi/(2\pi)$ and on the y-axis put the θ_j maximally predicted by your circuit

As a separate graph, let $\phi/(2\pi)=0.1432394487827058$ and graph a histogram of the probability your circuit gives back the result θ_j (as a function of θ_j). Paste your histogram and mark on your histogram 0.1432:

Produce the maximally predicted θ_j plot and measured θ_j histogram for the circuit with **2 wires** on top:

Produce the maximally predicted θ_j plot and measured θ_j histogram for the circuit with **6 wires** on top. Also paste the circuit description for this phase estimate circuit.

Using ϕ=0.5 and the given initial state, run the phase estimate circuit with 6 wires on top. Make a graph which histograms how often you get all 2^6 outputs for the top wires.

Paste a circuit description for the ϕ=0.5, 6 top wire phase estimation circuit that uses fewer gates to represent the Quantum Fourier Transform:

Come up with your own circuit description for phase estimation with a U on the bottom wire that is made of NOTs and P gates, rather than just a single phase gate as we have been doing. Run your phase estimation circuit with this U gate and generate a histogram of the possible outputs for the top wires.

Quantum Fourier Transform

Paste the circuit description for the three-qubit Quantum Fourier Transform (QFT). Demonstrate that it produces the correct output when run through your simulator:

Paste the circuit description for the five-qubit QFT. Demonstrate that it produces the correct output when run through your simulator. Show the output of the five-qubit QFT when run with the myInputState input file:

Understanding the QFT (extra credit)

Work through the three approaches for building the QFT.

Classical Shors

Show that your code successfully factors numbers up to 10 bits:

Factor the number 33 and give the x and r you find:

Put a list of ten N, x, r where N is less than 5 bits and x and r are not trivial:

How fast is classical Shor's? (extra credit)

Plot the execution time versus k:

Plot the frequency of the two failure modes as a function of k and show that they do not scale linearly with k:

Period Finding Unitary Matrix

Write code to produce a period finding unitary matrix for a given co-prime (x,N). Give an example of an output unitary matrix:

For a few different examples of x, N, generate the matrix U and find its eigenvalues e. Also, compute the period r using your Classical Shor's algorithm. For each (x,N), paste the vector of eigenvalues e, the period r, and the vector e*r, which should be integers:

Show that you can find the period r given a random eigenvalue of the matrix U for a particular (x,N).

Using the above algorithm for factoring using only the eigenvalues of the U matrix (without the help of the Classical Shor's algorithm), factor some numbers. Paste the output here:

Adding classical gates to your simulator

Paste in your circuit descriptions that use the xyModN and control-xyModN gates and show the input and output that verifies that they work:

Shor's Algorithm

Show that your quantum computing simulator running the Quantum Shor's circuit can successfully factor numbers. Try to factor 21.

Show that your simulator runs faster with the speed-up trick.