Hi!

Once again, thank you for your interest in the QC Mentorship program!

We decided to select participants based on how they will manage to do some simple “screening tasks”.

These tasks have been designed to:
- find out if you have the skills necessary to succeed in our program.
- be doable with basic QC knowledge - nothing should be too hard for you to quickly learn.
- allow you to learn some interesting concepts of QC.
- give you some choices depending on your interests.

What we mean by skills is not knowledge and expertise in QC. It’s the ability to code, learn new concepts and to meet deadlines.

What are we looking for in these applications?
- Coding skills – clear, readable, well-structured code
- Communication – well-described results, easy to understand, tidy.
- Reliability – submitted on time, all the points from the task description are met
- Research skills – asking good questions and answering them methodically

Also, feel free to be creative – once you finish the basic version of the task, you can expand it.

Choose tasks based on your interests, don’t try to pick the easiest one.

You need to do only 1 task. Feel free to do all of them, it might be a good learning opportunity, but it won’t affect admissions to the program :)

So here are the tasks:

Task 1

The Swap test is a simple quantum circuit which, given two states, allows to compute how much do they differ from each other.

1) Provide a variational (also called parametric) circuit which is able to generate the most general 1 qubit state. By most general 1 qubit state we mean that there exists a set of the parameters in the circuit such that any point in the Bloch sphere can be reached. Check that the circuit works correctly by showing that by varying randomly the parameters of your circuit you can reproduce correctly the Bloch sphere.
2) Use the circuit built in step 1) and, using the SWAP test, find the best choice of your parameters to reproduce a randomly generated quantum state made with 1 qubit.

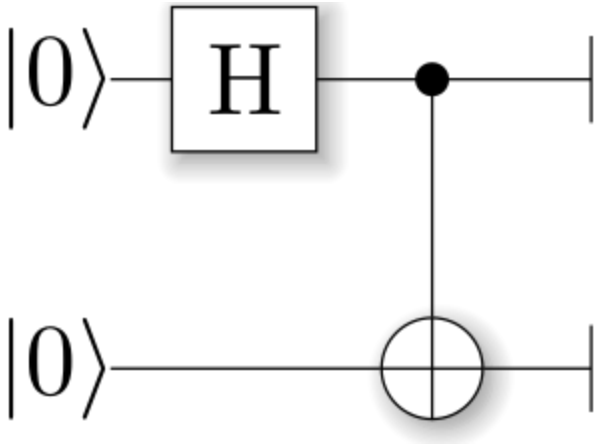
3) Suppose you are given with a random state, made by N qubits, for which you only know that it is a product state and each of the qubits are in the state | 0 > or | 1>. By product state we mean that it can be written as the product of single qubit states, without the need to do any summation. For example, the state
|a> = |01>
Is a product state, while the state
|b> = |00> + |11>
Is not.

Perform a qubit by qubit SWAP test to reconstruct the state. This part of the problem can be solved via a simple grid search.

Task 2

The bit-flip code and the sign-flip code (you can find a description of both here) are two very simple circuits able to detect and fix the bit-flip and the sign-flip errors, respectively.

1) Build the following simple circuit to prepare the Bell state:



2) Now add, right before the CNOT gate and for each of the two qubits, an arbitrary “error gate”. By error gate we mean that with a certain probability (that you can decide but must be non-zero for all the choices) you have a 1 qubit unitary which can be either the identity, or the X gate (bit-flip error) or the Z gate (sign-flip error).
3) Encode each of the two qubits with a sign-flip or a bit-flip code, in such a way that **all the possible** choices for the error gates described in 2), occurring on the logical qubits, can be detected and fixed. Motivate your choice. This is the most non-trivial part of the problem, so do it with a lot of care!
4) Test your solution by making many measurements over the final state and testing that the results are in line with the expectations.

Task 3

Learning by doing: the best way to understand the basics of quantum computation is to implement a quantum circuit simulator. This task is suitable both for people from computer sciences who want to learn about quantum computing, and for people from math/physics who want to exercise coding.

Detailed description of the task with some learning resources and examples can be found in this jupyter notebook

It is expected that simulator can perform following:

- initialize state
- read program, and for each gate:
    - calculate matrix operator
    - apply operator (modify state)
- perform multi-shot measurement of all qubits using weighted random technique

Task 4

The MaxCut problem is a well-known optimization problem in which the nodes of a given undirected graph have to be divided in two sets (often referred as the set of “white” and “black” nodes) such that the number of edges connecting a white node with a black node are maximized. The MaxCut problem is a problem on which the QAOA algorithm has proved to be useful (for an explanation of the QAOA algorithm you can read this blogpost).

At this link you can find an explicit implementation of the QAOA algorithm to solve the MaxCut problem for the simpler case of an *unweighted* graph. We ask you to generalize the above code to include also the solution for the case of *weighted* graphs. You can use the same code or you can also do an alternative implementation using, for example, qiskit. The important point is that you do not make use of any built-in QAOA functionalities.

Deadline

2 weeks from when you’ve submitted your application in your timezone.

Once you have finished a screening task, please submit your GitHub repository containing the code to this google form: https://forms.gle/akrQzeyciRzEfLgYA -- other forms of submission will not be accepted!

If you have any questions - please add comments to this document, or ask it in the QOSF slack workspace (invitation_link) in the #mentorship-applicants channel. We will be updating this document with more details and/FAQ to avoid confusion, so make sure to check it before asking :)

Have a nice day!
QOSF team

FAQ

Q: Can we use any quantum libraries or are we restricted to a particular set of tools?

A: Feel free to use whatever you like, just make sure that the tool doesn't solve the whole problem for you.
Regarding the language of choice, Python is definitely the preferred one, since this is the language that most of the mentors use.
You can do the task first in the language of your preference and then translate it to Python if that's more convenient for you.

Q: I am applying as a member of a team. How many tasks do we submit?

A: Each member of a team must submit their own screening task. This will help us judge the skill level of each individual team member and help us pair folks up with the right mentor.

Q: How should I submit the solution?

A: All the materials for the submission should be inside a GitHub repository. Please do not send us any loose files as attachments or in any other format. Please submit your GitHub repository to this google form once you've finished: https://forms.gle/akrQzeyciRzEfLgYA

Q: My team-mate wants to leave the team because he/she/they can't manage these along with exams. So will this affect our team status or anything like that?

A: Well, just let us know and you can continue as an individual/smaller team.

Q: Is it possible to make more than one task and send everything together?

Yes, you can. But you should specify which task you want to be evaluated. In other words, do it as an exercise but it does not affect your chances to enter the program.