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” (to be honest, some turned out to be rather complicated, but we hope not too much for you!)

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. Bonus questions provide just some ideas how to expand given topic.

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

Design a quantum circuit that considers as input the following vector of integers numbers:

[1,5,7,10]

returns a quantum state which is a superposition of indices of the target solution, obtaining in the output the indices of the inputs where two adjacent bits will always have different values. In this case the output should be: 1/sqrt(2) * (|01> + |11>), as the correct indices are 1 and 3.

1 = 0001
5 = 0101
7 = 0111
10 = 1010

Context

If you're struggling to find a proper way to solve this task, you can find some suggestions for a possible solution below. This is one way to approach the problem, but other solutions may be feasible as well, so feel free to also investigate different strategies if you see fit!

The key to this task is to use the superposition offered by quantum computing to load all the values of the input array on a single quantum state, and then locate the values that meet the target condition. So, how can we use a quantum computer to store multiple values? A possible solution is using the QRAM (some references: https://arxiv.org/pdf/0708.1879.pdf, https://github.com/qsharp-community/qram/blob/master/docs/primer.pdf).

As with classical computers, in the QRAM information is accessed using a set of bits indicating the address of the memory cell, and another set for the actual data stored in the array.
For example, if you want to use a QRAM to store 2 numbers that have most have 3 bits, it can be achieved with 1 qubit of address and 3 qubits of data.

Suppose you have the vector input_2 = [2,7].
In a properly constructed circuit, when the value of the address qubit is |0> the data qubits have value 010 (binary representation of 2) and when it is |1> in the data qubits have value 111 (binary representation of 7).

Given such a structure, you should be able to use Grover's algorithm in order to obtain the solution to the task.

You can assume that the input always contains at least two numbers that have alternating bitstrings.

Bonus:

Design a general circuit that accepts vectors with random values of size 2^n with m bits in length for each element and finds the state(s) indicated above from an oracle.

Task 2

- Prepare 4 random 4-qubit quantum states of your choice.
- Create and train a variational circuit that transforms input states into predefined output states. Namely
    - if random state 1 is provided, it returns state |0011>
    - if random state 2 is provided, it returns state |0101>
    - if random state 3 is provided, it returns state |1010>
    - if random state 4 is provided, it returns state |1100>
- What would happen if you provided a different state?

Analyze and discuss the results.

Feel free to use existing frameworks (e.g. PennyLane, Qiskit) for creating and training the circuits.
This PennyLane demo can be useful: Training a quantum circuit with Pytorch,
This Quantum Tensorflow tutorial can be useful: Training a quantum circuit with Tensorflow .

For the variational circuit, you can try any circuit you want. You can start from one with a layer of RX, RY and CNOTs, repeated a couple of times (though there are certainly better circuits to achieve this goal).

Context:
This challenge has been inspired by the following papers "A generative modeling approach for benchmarking and training shallow quantum circuits" and "Generation of High-Resolution Handwritten Digits with an Ion-Trap Quantum Computer". The target states of this task can be interpreted as the 2x2 "bars and stripes" patterns used in the first paper.

Task 3

Implement an interpreter of the qasm 3.0 code that can convert it to a quantum circuit (in the framework of your choice) and calculate a conjugate of a circuit. Provide examples showing that it works.

The gate list that you need to consider are X, Y, Z, RX, RY, RZ, H, S, $S^\dagger$, T, $T^\dagger$, CX, CCX, SWAP & CSWAP.

Some algorithms, such as Grover's algorithm (https://arxiv.org/pdf/2005.06468) or Quantum Autoencoders (https://arxiv.org/pdf/1612.02806) need the transpose conjugate of matrix U. In some frameworks, there is already a way to generate the conjugate of a gate or even a circuit, to help in such situations. In this challenge, you should do this yourself.

The transpose conjugate of a matrix U is denoted with $U^\dagger$, and it is obtained by taking the transpose of the matrix U and then taking the complex conjugate of each element.
Note that the transpose conjugate $U^\dagger$ of a unitary matrix U has the following properties:
$U^\dagger = U^{-1}$ and $U^{-1} U = I$.
https://en.wikipedia.org/wiki/Conjugate_transpose
https://en.wikipedia.org/wiki/Complex_conjugate

Idea for expanding:

- Try writing an interpreter that works also with symbolic parameters, i.e. “RX(theta)” instead of just “RX(0.2)”.

Task 4 – runtime estimation

Write a program that estimates how long running a variational quantum algorithm might take.
It should take the following data as input:

- Circuit (might be a circuit created in some popular framework, QASM file or some other format),
- Number of circuit evaluations per iteration,
- Number of iterations of the optimization loop,
- Device information – information about the device being used (e.g. execution times of the gates),
- Any additional information that you think is relevant.

An example of a simple, but not very accurate formula would be:

Total runtime = $(N_1 * t_1 + N_2 * t_2) * n_s * n_i$

Where:

- N_1, N_2 – number of 1-qubit (or 2) gates
- t_1, t_2 – time of execution of 1-qubit (or 2) gates
- n_s – number of samples per iteration
- n_i – number of iterations

Note that this doesn't take into account that certain gates can be executed in parallel.

This task is pretty open-ended – please try to make your formula as realistic as possible. It will require some investigation and review of existing literature or technical documentation on your own, which might turn out to be much more challenging than it seems, but we hope also much more rewarding :)

Once this is done, you can try analyzing some numerical data from the existing research in order and see how long running such a circuit took (if done on a real device) or could take (if data comes from a simulation).

Some papers with data about the quantum computing devices:

- N. Lacroix et al. (ETH), see Table I
- Arute et al., supplementary information (Google)
- Superconducting Qubits: current state of play (Review)
- Materials challenges and opportunities for quantum computing hardware (behind paywall :( )
- You can often find specific information about quantum devices on the website of the companies building quantum hardware/software.
- (Please send us other references if you know them, we'll add them here!)

Review paper on Variational Quantum Algorithms to look for factors that may contribute to longer runtimes:

- 1st review paper
- 2nd review paper

Deadline
2 weeks from when you've submitted your application in your timezone.
This means that if you submitted your application on September 15th, you can send your solution by midnight of September 29th.

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! (TODO: check if link is correct).

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 (TODO: check if link is correct)

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?
A: 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.

Q: Can I please get the slack link? I think the link has expired ?

A: try this: https://qosf.slack.com/archives/C019UEZRCM9
Another one to try:
https://join.slack.com/t/qosf/shared_invite/zt-bw59w8b9-WJ~k0~FAMHukTZov4AnLfA

Q: It is saying that my email has no account in the workspace. This is my first time trying to login. What should I do?
No idea, try this link perhaps:
https://join.slack.com/t/qosf/shared_invite/zt-bw59w8b9-WJ~k0~FAMHukTZov4AnLfA
?