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

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

<u>You need to do only 1 task.</u> 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 Multiplier
To make a multiplier, for this we design the input of two positive integers to a function and this function will process a quantum algorithm that makes the multiplier (see Draper adder) and returns the result in an integer.

you cannot use any implementation already designed by the framework

```
def multiplier(int:number_1, int ,number_2):
    “””
 number_1 : integer positive value that is the first parameter to the multiplier function,
number_2 : integer positive value that is the second parameter to the multiplier function.
Return the positive integer value of the multiplication between number_1 and number_2
    “””

    # use a framework that works with quantum circuits, qiskit, cirq, pennylane, etc.


     # define a quantum circuit to convert the integer values in qubits, example bases
encoding
     # basis encoding: n bits are equals to a state of n qubits, example
     # The integer value 3 convert to a binary string that is 11, the basis encoding is |11⟩

     # use the state of the art to check the possibles ways to design a multiplier

  Return # the result of the quantum circuit into an integer

# consider print your quantum circuit,
```

Example:

```
A = multiplier(5,6)
print(A)

30
```

Bonus:

Use your proposal to design different inputs, and check the limitations of your simulator and framework, consider number of qubits, time of execution, the depth of the quantum circuit and number of the gates.

References:

[1] Addition on a Quantum Computer
https://arxiv.org/pdf/quant-ph/0008033.pdf

[2] T-count Optimized Design of Quantum Integer Multiplication
https://arxiv.org/pdf/1706.05113.pdf

[3] Quantum arithmetic with the Quantum Fourier Transform
https://arxiv.org/pdf/1411.5949.pdf

Task 2 Missing Number
From a function that has as a parameter a vector of positive integers of size 2^n -1, which is missing a number, this vector can be disordered, to search for the missing number from a quantum circuit.

```
def missing_number(Optional[list,array, str]: input_vector):
    “””
 input_vector: List, array or string that contain integer values of size 2^n -1, where are
missing a number to obtain all the number 2^n

Return the positive integer value that is missing in the input.
    “””


    # use a framework that works with quantum circuits, qiskit, cirq, pennylane, etc.

     # define a quantum circuit to convert the vector in a quantum circuit
    # define an oracle to find the missing value
   # encoding the output value in an integer value

# consider print your quantum circuit,
```

Example:

```
A =  missing_number([2,0,1])
print(A)
3
```

Bonus:

Which is the largest list that can be implemented? Identify it and describe the result

References:

[1] Deutsch, David, and Richard Jozsa. "Rapid solution of problems by quantum computation." Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences 439.1907 (1992): 553-558.

[2] Bernstein, Ethan, and Umesh Vazirani. "Quantum complexity theory." SIAM Journal on computing 26.5 (1997): 1411-1473.

[3] Grover, Lov K. , "A fast quantum mechanical algorithm for database search", Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (1996), arXiv:quant-ph/9605043

Task 3 Mottonen state preparation
Implement the Mottonen state preparation of any dataset you have for at most one 8-element vector.

you cannot use any implementation already designed by the framework

```
defstate_prep(Optional[list,array]: input_vector):
    “””
 input_vector: List, array that contain float values of size 2^n
Return the  mottomen state preparation of the input_vector
    “””

    # use a framework that works with quantum circuits, qiskit, cirq, pennylane, etc.

     # define a quantum circuit to convert the vector in a quantum circuit
    # define the Mottonen state


# consider print your quantum circuit
```

Bonus:
Consider a state vector of size 5,7,10 how you can implement a vector of size different to 2^n.

References:

[1] Transformation of quantum states using uniformly controlled rotations
https://arxiv.org/pdf/quant-ph/0407010.pdf

Task 4 sending files
It uses the BB84 protocol to generate a key with which you are going to send a multimedia file (image) or video and this can be transferred from the quantum teleportation and deciphered with the same key. send the QOSF logo from folder mentee to folder mentor

you cannot use any implementation already designed by the framework

Download the QOSF logo, create a folder mentee and a folder mentor, in the mentee folder save the image of the QOSF logo(grey scale). Design a function called send_file where you send the QOSF logo to the folder mentor.

```
def send_file(str: path, str:name,str:path_destionation):
    “””
  path: String value where is the local folder (mentee folder)
  name: String for the file to send to the mentor folder, (the name of the QOSF logo)
 Path_destionation: String value where is the destination folder (mentor folder)
Return is success or not

    “””
    # use a framework that works with quantum circuits, qiskit, cirq, pennylane, etc.
    # define a quantum circuit to convert the image in a quantum circuit, you can convert
in a gray scale the logo.
   # define the protocol BB84 and the quantum teleportation
   # save the result in the destination folder.

# consider show the result in a folder and explain the process
```

Describe the process and explain the limitations, advantages of this process for security.

Bonus:
Try to use it on 2 different computers.

References
[1] Simple Proof of Security of the BB84 Quantum Key Distribution Protocol

https://arxiv.org/pdf/quant-ph/0003004.pdf

Deadline
2 weeks from when you've submitted your application in your timezone.
This means that if you submitted your application on September 25th, you can send your solution by midnight of October 9th.

Once you have finished a screening task, please submit your GitHub repository containing the code to this google form: https://forms.gle/nM9jxFD8e7tnScUX7 -- 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/nM9jxFD8e7tnScUX7

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-vb1ftjp1-D2gpVKEfl6Ifv9oXvk_xDw