IBM Quantum Developer Certification Exam Cheat Sheet

qubit is represented in the below forms

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

where alpha and beta are the Prob. of qubit |0> and |1> respectively

$$|\psi\rangle = \begin{pmatrix}\alpha \\ \beta\end{pmatrix}.$$

The states $|0\rangle$ and $|1\rangle$ are usually represented as

$$|0\rangle = \begin{pmatrix}1 \\ 0\end{pmatrix}, \quad |1\rangle = \begin{pmatrix}0 \\ 1\end{pmatrix}.$$

$$|q\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle$$

Single qubit gates

X-gate (similar to classical not gate)

- Flips |0> qubit to |1> and |1> to |0> respectively
- It is mathematically represented as

$$X = \begin{bmatrix}0 & 1 \\ 1 & 0\end{bmatrix}:$$

Y-gate

- It converts |0> to i|1> and |1> to -i|0> respectively
- It is mathematically represented as

$$Y = \begin{bmatrix}0 & -i \\ i & 0\end{bmatrix}$$

Z-gate

- It does no operation on |0> and converts |1> to -|1>
- It is mathematically represented as

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

H-gate (Hadamard Gate)

- H-gate puts the |0> or |1> qubit in a superposition of |0> and |1> states
- It is mathematically represented as

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Phase gate

- It is mathematically represented as

$$P(\phi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}$$

- When phase = Pi/2, we represent the gate as S-gate and it is represented as

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{2}} \end{bmatrix}$$

- When phase = Pi/4, we represent the gate as T-gate and it is represented as

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix}$$

U-gate

- It is the most general of all quantum single qubit gates and it is represented as

$$U(\theta,\phi,\lambda) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -e^{i\lambda}\sin(\frac{\theta}{2}) \\ e^{i\phi}\sin(\frac{\theta}{2}) & e^{i(\phi+\lambda)}\cos(\frac{\theta}{2}) \end{bmatrix}$$

Rotation-gate

- Rotation gates are mathematically represented as

$$R_x(\theta) = \exp(-iX\theta/2) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix},$$
$$R_y(\theta) = \exp(-iY\theta/2) = \begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix},$$
$$R_z(\theta) = \exp(-iZ\theta/2) = \begin{bmatrix} \exp(-i\theta/2) & 0 \\ 0 & \exp(i\theta/2) \end{bmatrix}.$$

Mathematical Identities (both for the exam point of view and practical approach)

- HZH = X
- HXH = Z
- HZH = X
- ZXZ = -X
- ZYZ = -Y
- ZZZ = Z
- XX = I
- YY = I
- ZZ = I

Proof for these identities are explained in the separate document, please make sure to check out that.

Involutory Gates (Gates that have its own inverse)

Pauli gates X,Y and Z and Hadamard gates are Involutory Gates.

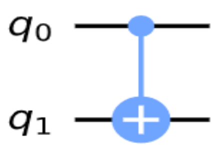
Hermitian Gates (Gates that have its own inverse)

Hadamard (H) and the Pauli gates (I, X, Y, Z) are Hermitian Gates.

Multi-qubit Gates

Controlled-NOT

- conditional not gate that performs an X-gate on the second qubit (target), if the state of the first qubit (control) is |1>. In the below circuit q0 is the control qubit and q1 is the target qubit



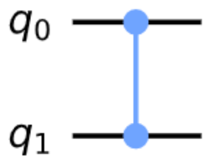
The classical truth table for the given circuit is

Input (q1,q0)	Output (q1,q0)
00	00
01	11
10	10
11	01

Note: In 3-qubit system,if we have qubit 0 is 1,qubit 1 is 0 and qubit 2 is 0,then we represent this as |001> (meaning higher order qubits will be first in the order followed by other qubits)

Controlled-Z

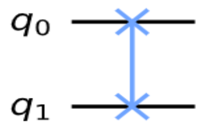
- conditional Z gate that performs an Z-gate on the second qubit (target), if the state of the first qubit (control) is |1>. In the below circuit q0 is the control qubit and q1 is the target qubit



Prepared by Simplify. Follow us on https://www.linkedin.com/company/simplify2

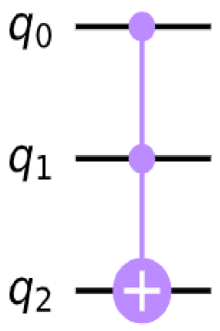
SWAP Gate

- It swaps the state of qubit q0 to q1 and q1 to q0 respectively



Toffoli Gate

- It is a three-qubit gate with two controls and one target. It performs an X gate operation on the target only if both control qubits are in |1> state.



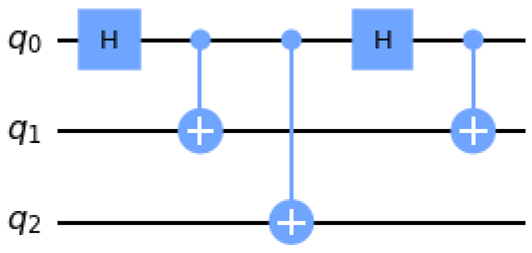
Qiskit 101

Quantum Circuits

- In Qiskit,we always represent the higher order qubit first followed by lower order qubits
  For instance:
- We can add two individual circuits using + operator or using the compose function in qiskit
  For example, we have two quantum circuits qc1 and qc2 to combine them
    - If we have two quantum circuit of same qubits, then new resultant quantum circuit
        - new_qc = qc1 + qc2
    - If we have two quantum circuit of different qubits, we can use compose function, then new resultant quantum circuit
        - new_qc = qc1.compose(qc2,[0,1])

Depth of the Quantum circuit

- Depth of the quantum circuit is given by QuantumCircuit.depth( ). In the below circuit,final output will depend on the H-gate1, C-NOT gate1, C-NOT gate2, H-gate1 and C-NOT gate3. Since the input has to propagate to these 5 levels, then depth of the circuit is 5.



Drawing the Quantum circuit

- QuantumCircuit.draw returns
  TextDrawing (output='text')
  matplotlib.figure (output='mpl')
  PIL.Image (output='latex')
  str (output='latex_source')

Barrier of the Quantum circuit

- To put barriers across all qubits after line of code, we can either use qc.barrier( ) or qc.barrier(*qargs)
  for example in 3 qubit circuit, if we want to apply barrier to all the qubits then use qc.barrier([0,1,2])
- To put barrier across one qubit after line of code, we use qc.barrier(particular_qubit)
  for example in 3 qubit circuit, if we want to apply barrier to the first qubits then use qc.barrier(0)

QuantumCircuit to QASM

- Convert quantum circuit to qasm string
  - qc.qasm(formatted='True')
- Convert quantum circuit to qasm string and saving to qasm file
  - qc.qasm(formatted='True',filename='new_qc.qasm')
- Convert qasm file to quantum circuit
  - QuantumCircuit.from_qasm_file('new_qc.qasm')

BasicAer: Python-based Simulators

Aer Provider

We have 3 Aer Provider in qiskit

- Qasm Simulator (Noisy quantum circuit simulator backend, It mimics the actual quantum computer)
- Statevector Simulator (Ideal quantum circuit state vector simulator)
- Unitary Simulator (Ideal quantum circuit unitary simulator)

QASM

- include "filename" continues parsing filename as if the contents of the file were inserted at the location of the include statement.

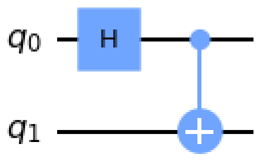
```
// First non-comment is a version string
OPENQASM 3.0;

include "stdgates.qasm";

// Rest of QASM program
```

Executing Quantum Circuits

- In Qiskit, always all qubits are initialized to 0 (when analysing the quantum circuit, please always keep this in mind)
- For example, we consider the below circuit, H-gate puts the qubit in superposition, then we have 1/sqrt(2) (|0> + |1>) in q0 and we have |0> in q1, applying tensorflow we get 1/sqrt(2) [cnot(|00>+|10>)] then we get the output will be 1/sqrt(2) (|00> + |11>)



- To put all qubits in equi-probable superposition, we apply H-gate to all qubits.
- Depending on the simulator, we finalize the output

Quantum Information

- In the quantum circuit, if we use the quantum registers and classical register, then we can measure the bit information using QuantumCircuit.measure( ) function
- In the quantum circuit, if we don't have classical registers. We cannot measure the bit information (which means we cannot use QuantumCircuit.measure( ) function), but we can use QuantumCircuit.measure_all( )
- Important Functions from qiskit.quantum_info

random_statevector(dims)	Generate a random state vector
random_unitary(dims)	Generate a random unitary Operator
random_quantum_channel( )	Generate a random CPTP quantum channel.

Fidelity

average_gate_fidelity( )	Return the average gate fidelity of a noisy quantum channel.
process_fidelity( )	Return the process fidelity of a noisy quantum channel.
gate_error( )	Return the gate error of a noisy quantum channel.

Operator

Operator(qc) is used to convert the QuantumCircuit to the operator.

For example to convert Z gate to operator, we can do it in two possible ways

```
qc  = QuantumCircuit(1)

qc.z(0)

op =Operator(qc)
```

(or)

```
op = Operator( [[1,0,0,-1]] )
```

Qiskit tools

job_monitor( )	used to monitor the job status
%qiskit_job_watcher	used to watch the status of jobs scheduled

Displaying and using system information

- Displaying the version of various qiskit libraries

    %qiskit_version_table
    (or)
    print(qiskit.__qiskit_version__)

- If we want to specifically view the qiskit terra version then use the below command

    qiskit.__version__

Coherence time refers to how long the qubits stay in the wave-like, quantum superposition state. But as the qubits talk to each other and to the wires in their environment, quantum information leaks out, resulting in decoherence. The coherence time and the time it takes to do a quantum gate "sets the time limit on how big a calculation you can do,"

Visualizations (please refer visualization Notebook for more information)

Plot a histogram of data.
plot_histogram(data[, figsize, color, ...])

Plot the Bloch sphere.
plot_bloch_vector(bloch[, title, ax, ...])

Plot the Bloch sphere.
plot_bloch_multivector(state[, title, ...])

Plot the cityscape of quantum state.
plot_state_city(state[, title, figsize, ...])

Plot a hinton diagram for the density matrix of a quantum state.
plot_state_hinton(state[, title, figsize, ...])

Plot the Pauli Vector representation of a quantum state.
plot_state_paulivec(state[, title, figsize, ...])

Plot the qsphere representation of a quantum state.

plot_state_qsphere(state[, figsize, ax, ...])

Note:

$|0\rangle$ the corresponding bloch vector is (0,0,1)

$|1\rangle$ the corresponding bloch vector is (0,0,-1)

$|+\rangle$ the corresponding bloch vector is (1,0,0)

$|-\rangle$ the corresponding bloch vector is (-1,0,0)

$|i\rangle$ the corresponding bloch vector is (0,1,0)

$|-i\rangle$ the corresponding bloch vector is (0,-1,0)

Prepared by Simplify. Follow us on https://www.linkedin.com/company/simplify2