Preparation for Qiskit certification exam

Presenter: **Prateek Jain,** IBM Quantum Educator

Lead Researcher Quantum AI @ Fractal Analytics

Outline

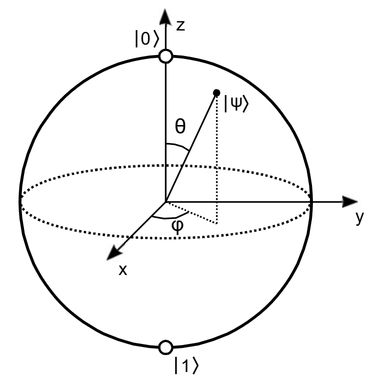
- **Introduction of quantum computing and Qiskit**
- Brief Details About the Exam
- Sample Exam go through & Answers with explanation

Introduction to Quantum computing

- Qubit (quantum bit)
  - Superposition

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad |\alpha|^2 + |\beta|^2 = 1.$$

  - Entanglement
    - Bell state $|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$



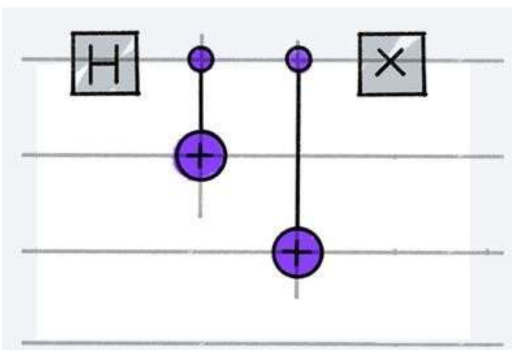
Bloch sphere. Source: Wiki

- Quantum operations
  - Single-qubit gate: X, Z, H, etc.
  - Two-qubit gate: CNOT gate, CZ gate, SWAP gate etc.
  - Measurement

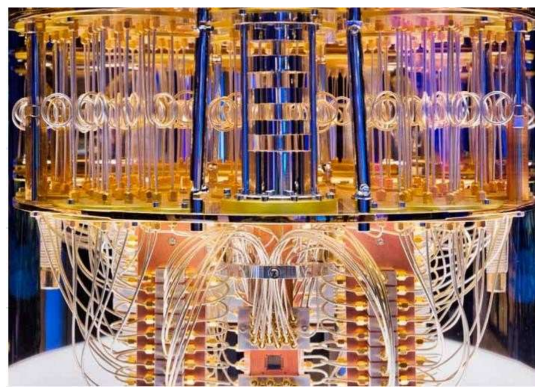
- References
  - [Quantum computation and quantum information]

Introduction to IBMQ and Qiskit

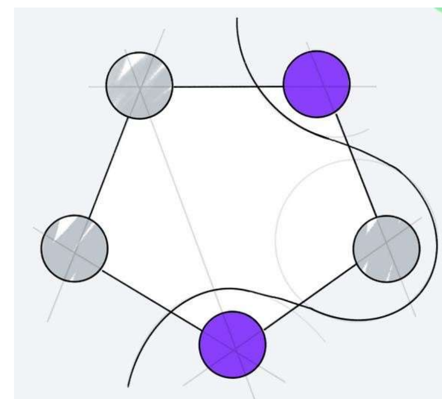
- An open-source toolkit for working with quantum computers at the level of pulses, circuits, and application modules.



Quantum circuits



Quantum hardware



Quantum algorithm

Source: https://qiskit.org/

Outline

- Introduction of quantum computing and Qiskit
- **Brief Details About the Exam**
- Sample Exam go through & Answers with explanation

Brief Details about the Exam

- Test information
  - Number of questions: 60
  - Time allowed: 90 mins
  - Number of questions to pass: 44
  - Status: Live
  - Official Study guide
  - Use this **form** to Get a free Exam Coupon
- Tips
  - Only laptop is allowed (mac/windows)
  - Prepare a mouse (use provided white board for calculation)
  - Multiple choice questions (how many choices is indicated)
- Resources
  - For any doubts refer to Qiskit slack channel #qiskit-cert-exam
  - Add certi page link, certi sample link, add other imp resources links like My own notebook, Bartu notebook etc

Outline

- Introduction of quantum computing and Qiskit
- Brief Details About the Exam
- **Sample Exam go through with Answers & explanation**

Create a quantum circuit

1. **Which statement will create a quantum circuit with four quantum bits and four classical bits?**

   A. `QuantumCircuit(4, 4)`
   B. `QuantumCircuit(4)`
   C. `QuantumCircuit(QuantumRegister(4, 'qr0'), QuantumRegister(4, 'cr1'))`
   D. `QuantumCircuit([4, 4])`

Create a quantum circuit

qiskit.circuit.QuantumCircuit ¶

CLASS `QuantumCircuit(*regs, name=None, global_phase=0, metadata=None)` [SOURCE] ¶

Create a new circuit.

A circuit is a list of instructions bound to some registers.

Parameters

- **regs** (list(`Register`) or list(`int`) or list(list(`Bit`))) –
  The registers to be included in the circuit.
  - If a list of `Register` objects, represents the `QuantumRegister` and/or `ClassicalRegister` objects to include in the circuit.
    For example:
    - `QuantumCircuit(QuantumRegister(4))`
    - `QuantumCircuit(QuantumRegister(4), ClassicalRegister(3))`
    - `QuantumCircuit(QuantumRegister(4, 'qr0'), QuantumRegister(2, 'qr1'))`
  - If a list of `int`, the amount of qubits and/or classical bits to include in the circuit. It can either be a single int for just the number of quantum bits, or 2 ints for the number of quantum bits and classical bits, respectively.
    For example:
    - `QuantumCircuit(4) # A QuantumCircuit with 4 qubits`
    - `QuantumCircuit(4, 3) # A QuantumCircuit with 4 qubits and 3 classical bits`

Source: https://qiskit.org/documentation/

Answer

Create a quantum circuit

1. Which statement will create a quantum circuit with four quantum bits and four classical bits?

A. `QuantumCircuit(4, 4)` ✓

B. `QuantumCircuit(4)`

C. `QuantumCircuit(QuantumRegister(4, 'qr0'), QuantumRegister(4, 'cr1'))`

D. `QuantumCircuit([4, 4])`

Measurement probability

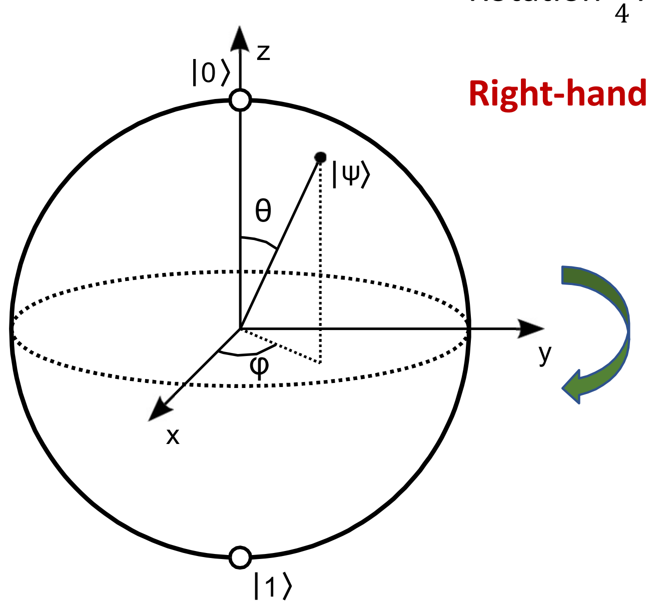
2. Given this code fragment, what is the probability that a measurement would result in |0> ?

```
qc = QuantumCircuit(1)
qc.ry(3 * math.pi/4, 0)
```

A. 0.8536
B. 0.5
C. 0.1464
D. 1.0

Rotation $\frac{3}{4}\pi$ around Y axis

Right-hand rule!



Answer

Measurement probability

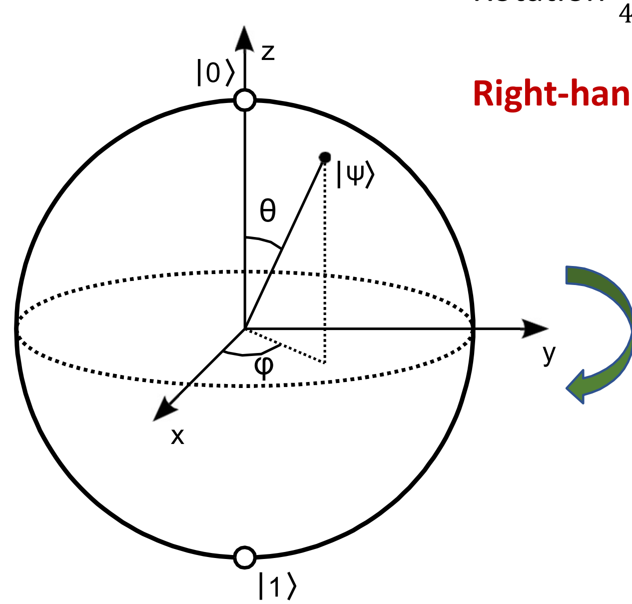
2. Given this code fragment, what is the probability that a measurement would result in |0> ?

```
qc = QuantumCircuit(1)
qc.ry(3 * math.pi/4, 0)
```

A. 0.8536
B. 0.5
C. ✓ 0.1464
D. 1.0

Rotation $\frac{3}{4}\pi$ around Y axis

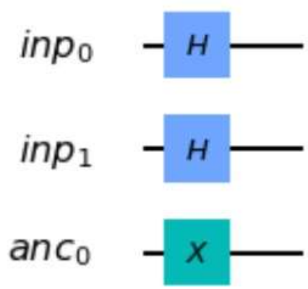
Right-hand rule!



Create a given circuit

3. Assuming the fragment below, which three code fragments would produce the circuit illustrated?

```
inp_reg = QuantumRegister(2, name='inp')
ancilla = QuantumRegister(1, name='anc')
qc = QuantumCircuit(inp_reg, ancilla)

# Insert code here
```



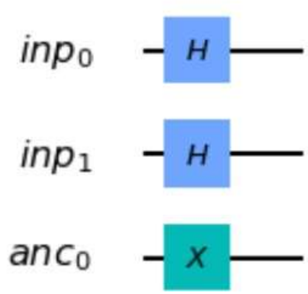
A.
```
qc.h(inp_reg)
qc.x(ancilla)
qc.draw()
```
B.
```
qc.h(inp_reg[0:2])
qc.x(ancilla[0])
qc.draw()
```
C.
```
qc.h(inp_reg[0:1])
qc.x(ancilla[0])
qc.draw()
```
D.
```
qc.h(inp_reg[0])
qc.h(inp_reg[1])
qc.x(ancilla[0])
qc.draw()
```
E.
```
qc.h(inp_reg[1])
qc.h(inp_reg[2])
qc.x(ancilla[1])
qc.draw()
```
F.
```
qc.h(inp_reg)
qc.h(inp_reg)
qc.x(ancilla)
qc.draw()
```

Answer

Create a given circuit

3. Assuming the fragment below, which three code fragments would produce the circuit illustrated?

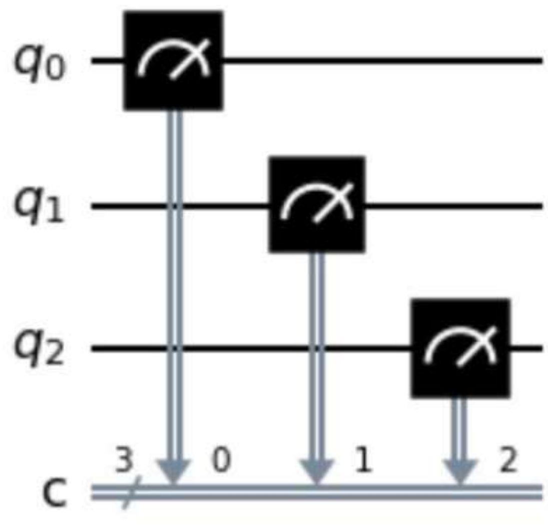
```
inp_reg = QuantumRegister(2, name='inp')
ancilla = QuantumRegister(1, name='anc')
qc = QuantumCircuit(inp_reg, ancilla)

# Insert code here
```



A.
```
qc.h(inp_reg)
qc.x(ancilla)
qc.draw()
```
B.
```
qc.h(inp_reg[0:2])
qc.x(ancilla[0])
qc.draw()
```
C.
```
qc.h(inp_reg[0:1])
qc.x(ancilla[0])
qc.draw()
```
D.
```
qc.h(inp_reg[0])
qc.h(inp_reg[1])
qc.x(ancilla[0])
qc.draw()
```
E.
```
qc.h(inp_reg[1])
qc.h(inp_reg[2])
qc.x(ancilla[1])
qc.draw()
```
F.
```
qc.h(inp_reg)
qc.h(inp_reg)
qc.x(ancilla)
qc.draw()
```

Measurement

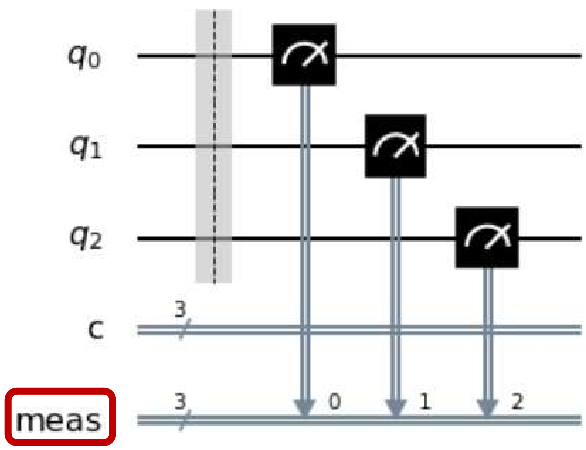
4. Given an empty QuantumCircuit object, qc, with three qubits and three classical bits, which one of these code fragments would create this circuit?



A. `qc.measure([0,1,2], [0,1,2])`
B. `qc.measure([0,0], [1,1], [2,2])`
C. `qc.measure_all()`
D. `qc.measure(0,1,2)`

```
def measure(self, qubit, cbit):
    """Measure quantum bit into classical bit (tuples).

    Args:
        qubit (QuantumRegister|list|tuple): quantum register
        cbit (ClassicalRegister|list|tuple): classical register
```

measure_all

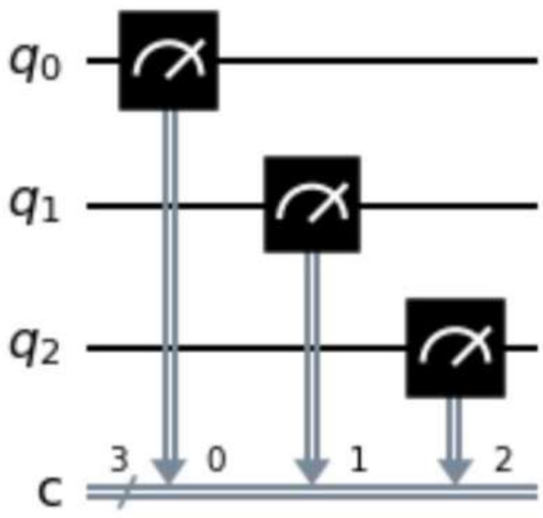


Source: https://qiskit.org/documentation/

Answer

Measurement

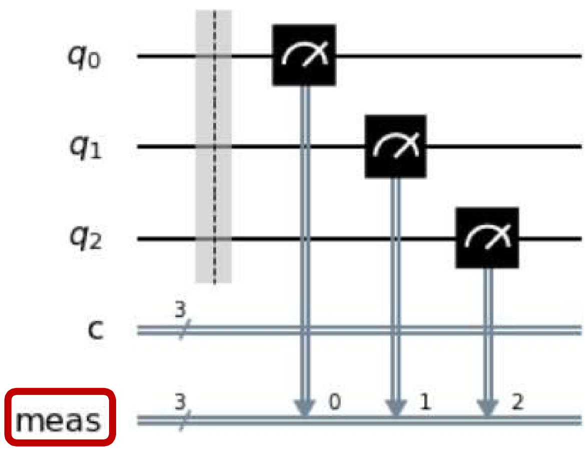
4. Given an empty QuantumCircuit object, qc, with three qubits and three classical bits, which one of these code fragments would create this circuit?



A. `qc.measure([0,1,2], [0,1,2])`
B. `qc.measure([0,0], [1,1], [2,2])`
C. `qc.measure_all()`
D. `qc.measure(0,1,2)`

```
def measure(self, qubit, cbit):
    """Measure quantum bit into classical bit (tuples).

    Args:
        qubit (QuantumRegister|list|tuple): quantum register
        cbit (ClassicalRegister|list|tuple): classical register
```

measure_all



Source: https://qiskit.org/documentation/

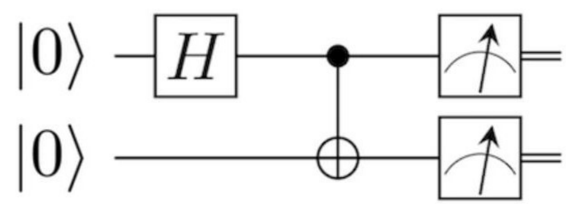
Bell state

5. Which code fragment will produce a maximally entangled, or Bell, state?

A.
```
bell = QuantumCircuit(2)
bell.h(0)
bell.x(1)
bell.cx(0, 1)
```
B.
```
bell = QuantumCircuit(2)
bell.cx(0, 1)
bell.h(0)
bell.x(1)
```
C.
```
bell = QuantumCircuit(2)
bell.h(0)
bell.x(1)
bell.cz(0, 1)
```
D.
```
bell = QuantumCircuit(2)
bell.h(0)
bell.h(0)
```

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$
$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$$
$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$
$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

Bell states



Answer

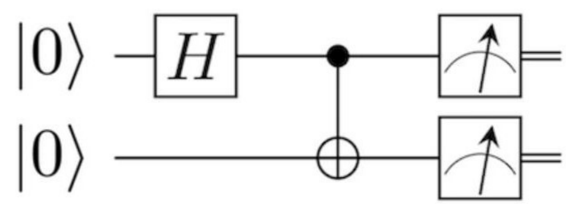
Bell state

5. Which code fragment will produce a maximally entangled, or Bell, state?

A.
```
bell = QuantumCircuit(2)
bell.h(0)
bell.x(1)
bell.cx(0, 1)
```
B.
```
bell = QuantumCircuit(2)
bell.cx(0, 1)
bell.h(0)
bell.x(1)
```
C.
```
bell = QuantumCircuit(2)
bell.h(0)
bell.x(1)
bell.cz(0, 1)
```
D.
```
bell = QuantumCircuit(2)
bell.h(0)
bell.h(0)
```

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$
$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$$
$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$
$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

Bell states

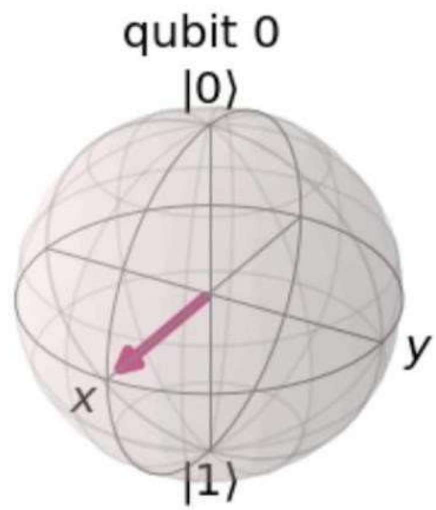


Quantum operation

6. Given this code, which two inserted code fragments result in the state vector represented by this Bloch sphere?

```
qc = QuantumCircuit(1,1)
# Insert code fragment here

simulator = Aer.get_backend('statevector_simulator')
job = execute(qc, simulator)
result = job.result()
outputstate = result.get_statevector(qc)
plot_bloch_multivector(outputstate)
```

A. `qc.h(0)`
B. `qc.rx(math.pi / 2, 0)`
C. `qc.ry(math.pi / 2, 0)`
D. `qc.rx(math.pi / 2, 0)`
`qc.rz(-math.pi / 2, 0)`
E. `qc.ry(math.pi, 0)`



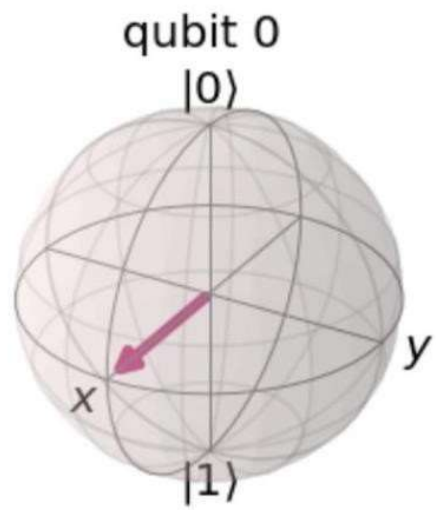
Answer

Quantum operation

6. Given this code, which two inserted code fragments result in the state vector represented by this Bloch sphere?

```
qc = QuantumCircuit(1,1)
# Insert code fragment here

simulator = Aer.get_backend('statevector_simulator')
job = execute(qc, simulator)
result = job.result()
outputstate = result.get_statevector(qc)
plot_bloch_multivector(outputstate)
```

A. ✓ `qc.h(0)`
B. `qc.rx(math.pi / 2, 0)`
C. ✓ `qc.ry(math.pi / 2, 0)`
D. `qc.rx(math.pi / 2, 0)`
`qc.rz(-math.pi / 2, 0)`
E. `qc.ry(math.pi, 0)`



Qiskit phase gate

7. S-gate is a Qiskit phase gate with what value of the phase parameter?

A. $\pi/4$
B. $\pi/2$
C. $\pi/8$
D. π

S gate: square root of Pauli-Z gate (Clifford gate)
T gate: fourth root of Pauli-Z gate (non-Clifford gate)

Answer

Qiskit phase gate

7. S-gate is a Qiskit phase gate with what value of the phase parameter?

A. π/4
B. π/2 ✓
C. π/8
D. π

S gate: square root of Pauli-Z gate (Clifford gate)
T gate: fourth root of Pauli-Z gate (non-Clifford gate)

State construction

8. Which two code fragments, when inserted into the code below, will produce the statevector shown in the output?

```
from qiskit import QuantumCircuit, Aer, execute
from math import sqrt

qc = QuantumCircuit(2)

# Insert fragment here

simulator = Aer.get_backend('statevector_simulator')
result = execute(qc, simulator).result()
statevector = result.get_statevector()
print(statevector)
```

Output:

```
[0.707+0.j  0.+0.j  0.+0.j  0.707+0.j]
```

$$\frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\1\end{pmatrix} => \frac{1}{\sqrt{2}}(|00> \ +|11>)$$

A.
```
v = [1/sqrt(2), 0, 0, 1/sqrt(2)]
qc.initialize(v,[0,1])
```
B.
```
qc.h(0)
qc.cx(0,1)
```
C.
```
v1, v2 = [1,0], [0,1]
qc.initialize(v1,0)
qc.initialize(v2,1)
```
D.
```
qc.cx(0,1)
qc.measure_all()
```
E.
```
qc.h(0)
qc.h(1)
qc.measure_all()
```

CLASS **`Initialize`**(*params*, *num_qubits=None*)

Complex amplitude initialization.

params (str, list, int or Statevector):

- Statevector: Statevector to initialize to.
- list: vector of complex amplitudes to initialize to.

Answer

State construction

8. Which two code fragments, when inserted into the code below, will produce the statevector shown in the output?

```
from qiskit import QuantumCircuit, Aer, execute
from math import sqrt

qc = QuantumCircuit(2)

# Insert fragment here

simulator = Aer.get_backend('statevector_simulator')
result = execute(qc, simulator).result()
statevector = result.get_statevector()
print(statevector)
```

Output:

```
[0.707+0.j  0.+0.j  0.+0.j  0.707+0.j]
```

$$\frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\1\end{pmatrix} => \frac{1}{\sqrt{2}}(|00> \ +|11>)$$

A. `v = [1/sqrt(2), 0, 0, 1/sqrt(2)]`
`qc.initialize(v,[0,1])`

B. `qc.h(0)`
`qc.cx(0,1)`

C. `v1, v2 = [1,0], [0,1]`
`qc.initialize(v1,0)`
`qc.initialize(v2,1)`

D. `qc.cx(0,1)`
`qc.measure_all()`

E. `qc.h(0)`
`qc.h(1)`
`qc.measure_all()`

CLASS **Initialize**(*params*, *num_qubits=None*)

Complex amplitude initialization.

params (str, list, int or Statevector):

- Statevector: Statevector to initialize to.
- list: vector of complex amplitudes to initialize to.

Source: https://qiskit.org/documentation/

Multi-qubit / CNOT gate

9. Which code fragment will produce a multi-qubit gate other than a CNOT ?

A. `qc.cx(0,1)`
B. `qc.cnot(0,1)`
C. `qc.mct([0],1)`
D. `qc.cz(0,1)`

`mct`*(self, q_controls, q_target, q_ancilla, mode='basic')* Multiple-Control Toffoli operation

$$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \qquad \mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Source: https://qiskit.org/documentation/

Answer

Multi-qubit / CNOT gate

9. **Which code fragment will produce a multi-qubit gate other than a CNOT ?**

A. `qc.cx(0,1)`
B. `qc.cnot(0,1)`
C. `qc.mct([0],1)`
D. `qc.cz(0,1)` ✓

`mct(self, q_controls, q_target, q_ancilla, mode='basic')` Multiple-Control Toffoli operation

$$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \qquad \mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Source: https://qiskit.org/documentation/

Toffoli gate

10. Which code fragment will produce a multi-qubit gate other than a Toffoli?

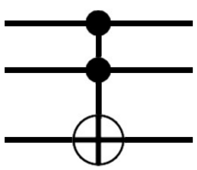
```
A. qc.ccx(0,1,2)
B. qc.mct([0,1], 2)
C. from qiskit.circuit.library import CXGate
ccx = CXGate().control()
qc.append(ccx, [0,1,2])
D. qc.cry(0,1,2)
```

```
CLASS CRYGate(theta, label=None, ctrl_state=None)
```

Controlled-RY gate.

Circuit symbol:

Toffoli gate



```
class CCXGate(ControlledGate):
""CCX gate, also known as Toffoli gate.
```

```
control(num_ctrl_qubits=1, label=None, ctrl_state=None)
```

Return a controlled-X gate with more control lines.

Source: https://qiskit.org/documentation/

Answer

Toffoli gate

10. Which code fragment will produce a multi-qubit gate other than a Toffoli?

A. `qc.ccx(0,1,2)`

B. `qc.mct([0,1], 2)`

C.
```
from qiskit.circuit.library import CXGate
ccx = CXGate().control()
qc.append(ccx, [0,1,2])
```

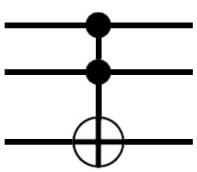
D. `qc.cry(0,1,2)`

```
CLASS CRYGate(theta, label=None, ctrl_state=None)
```

Controlled-RY gate.

Circuit symbol:

Toffoli gate



```
class CCXGate(ControlledGate):
""CCX gate, also known as Toffoli gate.
```

```
control(num_ctrl_qubits=1, label=None, ctrl_state=None)
```

Return a controlled-X gate with more control lines.

Source: https://qiskit.org/documentation/

Barriers

11. Which two options would place a barrier across all qubits to the QuantumCircuit below?

```
qc = QuantumCircuit(3,3)
```

A. `qc.barrier(qc)`
B. `qc.barrier([0,1,2])`
C. `qc.barrier()`
D. `qc.barrier(3)`
E. `qc.barrier_all()`

Answer

Barriers

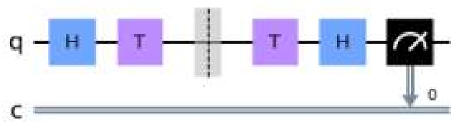
11. Which two options would place a barrier across all qubits to the QuantumCircuit below?

```
qc = QuantumCircuit(3,3)
```

A. `qc.barrier(qc)`
B. `qc.barrier([0,1,2])`
C. `qc.barrier()`
D. `qc.barrier(3)`
E. `qc.barrier_all()`

Barriers and circuit optimisation

12. What code fragment codes the equivalent circuit if you remove the barrier in the following QuantumCircuit?



A.
```
qc = QuantumCircuit(1,1)
qc.h(0)
qc.s(0)
qc.h(0)
qc.measure(0,0)
```
B.
```
qc = QuantumCircuit(1,1)
qc.measure(0,0)
```
C.
```
qc = QuantumCircuit(1,1)
qc.h(0)
qc.t(0)
qc.tdg(0)
qc.h(0)
qc.measure(0,0)
```
D.
```
qc = QuantumCircuit(1,1)
qc.h(0)
qc.z(0)
qc.h(0)
qc.measure(0,0)
```

The transpiler does not optimize across barriers.

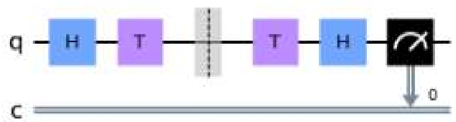
T gate: fourth root of Pauli-Z gate (non-Clifford gate)

T * T = S

$T^{\dagger} = tdg$

Barriers and circuit optimisation

12. What code fragment codes the equivalent circuit if you remove the barrier in the following QuantumCircuit?



A.
```
qc = QuantumCircuit(1,1)
qc.h(0)
qc.s(0)
qc.h(0)
qc.measure(0,0)
```
B.
```
qc = QuantumCircuit(1,1)
qc.measure(0,0)
```
C.
```
qc = QuantumCircuit(1,1)
qc.h(0)
qc.t(0)
qc.tdg(0)
qc.h(0)
qc.measure(0,0)
```
D.
```
qc = QuantumCircuit(1,1)
qc.h(0)
qc.z(0)
qc.h(0)
qc.measure(0,0)
```

The transpiler does not optimize across barriers.

T gate: fourth root of Pauli-Z gate (non-Clifford gate)

T * T = S

$T^{\dagger} = tdg$

Circuit depth

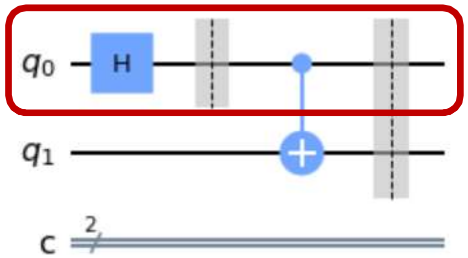
13. Given the following code, what is the depth of the circuit?

```
qc = QuantumCircuit(2, 2)

qc.h(0)
qc.barrier(0)
qc.cx(0,1)
qc.barrier([0,1])
```

A. 2
B. 3
C. 4
D. 5

Circuit depth: length of critical path (quantum gates and measurement)

Barrier is not a quantum operation!



Circuit depth

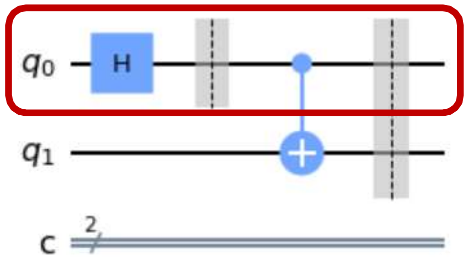
13. Given the following code, what is the depth of the circuit?

```
qc = QuantumCircuit(2, 2)

qc.h(0)
qc.barrier(0)
qc.cx(0,1)
qc.barrier([0,1])
```

A. 2
B. 3
C. 4
D. 5

Circuit depth: length of critical path (quantum gates and measurement)

Barrier is not a quantum operation!



Quantum simulator

```
[docs]def execute(
    experiments,
    backend,
    basis_gates=None,
    coupling_map=None,  # circuit transpile options
    backend_properties=None,
    initial_layout=None,
    seed_transpiler=None,
    optimization_level=None,
    pass_manager=None,
    qobj_id=None,
    qobj_header=None,
    shots=None,  # common run options
```

14. Which code snippet would execute a circuit given these parameters?

1) • Measure the circuit 1024 times,
2) • use the QASM simulator,
3) • and use a coupling map that connects three qubits linearly

```
qc = QuantumCircuit(3)

# Insert code fragment here
result = job.result()
```

A.
```
qasm_sim = Aer.get_backend('qasm_simulator')
couple_map = [[0, 1], [1, 2]]
job = execute(qc, backend=qasm_sim, shots=1024,
coupling_map=couple_map)
```
B.
```
qasm_sim = Aer.getBackend('ibmq_simulator')
couple_map = [[0, 1], [0, 2]]
job = execute(qc, loop=1024, coupling_map=couple_map)
```
C.
```
qasm_sim = Aer.get_backend('qasm_simulator')
couple_map = [[0, 1], [1, 2]]
job = execute(qc, backend=qasm_sim, repeat=1024,
coupling_map=couple_map)
```
D.
```
qasm_sim = Aer.get_backend('qasm_simulator')
couple_map = [[0, 1], [1, 2]]
job = execute(backend=qasm_sim, qc, shot=1024,
coupling_map=couple_map)
```

Source: https://qiskit.org/documentation/

Answer

Quantum simulator

```
[docs]def execute(
    experiments,
    backend,
    basis_gates=None,
    coupling_map=None,  # circuit transpile options
    backend_properties=None,
    initial_layout=None,
    seed_transpiler=None,
    optimization_level=None,
    pass_manager=None,
    qobj_id=None,
    qobj_header=None,
    shots=None,  # common run options
```

14. Which code snippet would execute a circuit given these parameters?

1) • Measure the circuit 1024 times,
2) • use the QASM simulator,
3) • and use a coupling map that connects three qubits linearly

```
qc = QuantumCircuit(3)

# Insert code fragment here
result = job.result()
```

A. `qasm_sim = Aer.get_backend('qasm_simulator')`
```
couple_map = [[0, 1], [1, 2]]
job = execute(qc, backend=qasm_sim, shots=1024,
coupling_map=couple_map)
```
B. `qasm_sim = Aer.getBackend('ibmq_simulator')`
```
couple_map = [[0, 1], [0, 2]]
job = execute(qc, loop=1024, coupling_map=couple_map)
```
C. `qasm_sim = Aer.get_backend('qasm_simulator')`
```
couple_map = [[0, 1], [1, 2]]
job = execute(qc, backend=qasm_sim, repeat=1024,
coupling_map=couple_map)
```
D. `qasm_sim = Aer.get_backend('qasm_simulator')`
```
couple_map = [[0, 1], [1, 2]]
job = execute(backend=qasm_sim, qc, shot=1024,
coupling_map=couple_map)
```

Source: https://qiskit.org/documentation/

Execute a circuit using qiskit

15. Which of these would execute a circuit on a set of qubits which are coupled in a custom way?

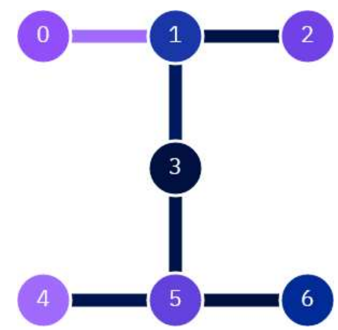
```
from qiskit import QuantumCircuit, execute, BasicAer
backend = BasicAer.get_backend('qasm_simulator')
qc = QuantumCircuit(3)

# insert code here
```

A. `execute(qc, backend, shots=1024, coupling_map=[[0,1], [1,2]])`

B. `execute(qc, backend, shots=1024, custom_topology=[[0,1],[2,3]]`

C. `execute(qc, backend, shots=1024, device="qasm_simulator", mode="custom")`

D. `execute(qc, backend, mode="custom")`



IBM Q 7 lagos

```
[docs]def execute(
    experiments,
    backend,
    basis_gates=None,
    coupling_map=None,  # circuit transpile options
```

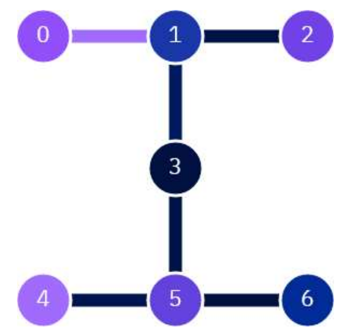
Answer

Execute a circuit using qiskit

15. Which of these would execute a circuit on a set of qubits which are coupled in a custom way?

```
from qiskit import QuantumCircuit, execute, BasicAer
backend = BasicAer.get_backend('qasm_simulator')
qc = QuantumCircuit(3)

# insert code here
```

A. ✓ execute(qc, backend, shots=1024, coupling_map=[[0,1], [1,2]])
B. execute(qc, backend, shots=1024, custom_topology=[[0,1],[2,3]]
C. execute(qc, backend, shots=1024, device="qasm_simulator", mode="custom")
D. execute(qc, backend, mode="custom")



IBM Q 7 lagos

```
[docs]def execute(
    experiments,
    backend,
    basis_gates=None,
    coupling_map=None,  # circuit transpile options
```

Source: https://qiskit.org/documentation/

BasicAer simulators

16. Which three simulators are available in BasicAer?

A. `qasm_simulator`
B. `basic_qasm_simulator`
C. `statevector_simulator`
D. `unitary_simulator`
E. `quantum_simulator`
F. `quantum_circuit_simulator`

Simulators

`QasmSimulatorPy` ([configuration, provider])	Python implementation of a qasm simulator.
`StatevectorSimulatorPy` ([configuration, provider])	Python statevector simulator.
`UnitarySimulatorPy` ([configuration, provider])	Python implementation of a unitary simulator.

BasicAer: A module of Python-based quantum simulators.

Aer: C++ based quantum simulators. Faster and support more features.

Source: https://qiskit.org/documentation/

Answer

BasicAer simulators

16. Which three simulators are available in BasicAer?

A. `qasm_simulator`
B. `basic_qasm_simulator`
C. `statevector_simulator`
D. `unitary_simulator`
E. `quantum_simulator`
F. `quantum_circuit_simulator`

Simulators ¶

`QasmSimulatorPy` ([configuration, provider])	Python implementation of a qasm simulator.
`StatevectorSimulatorPy` ([configuration, provider])	Python statevector simulator.
`UnitarySimulatorPy` ([configuration, provider])	Python implementation of a unitary simulator.

BasicAer: A module of Python-based quantum simulators.

Aer: C++ based quantum simulators. Faster and support more features.

Source: https://qiskit.org/documentation/

Statevector simulator

17. Which line of code would assign a statevector simulator object to the variable `backend` **?**

A. `backend = BasicAer.StatevectorSimulatorPy()`
B. `backend = BasicAer.get_backend('statevector_simulator')`
C. `backend = BasicAer.StatevectorSimulatorPy().name()`
D. `backend = BasicAer.get_back('statevector_simulator')`

Answer

Statevector simulator

17. Which line of code would assign a statevector simulator object to the variable `backend` **?**

A. `backend = BasicAer.StatevectorSimulatorPy()`
B. `backend = BasicAer.get_backend('statevector_simulator')`
C. `backend = BasicAer.StatevectorSimulatorPy().name()`
D. `backend = BasicAer.get_back('statevector_simulator')`

Quantum operator

18. Which code fragment would yield an operator that represents a single-qubit X gate?

A. `op = Operator.Xop(0)`
B. `op = Operator([[0,1]])`
C. `qc = QuantumCircuit(1)`
```
qc.x(0)
op = Operator(qc)
```
D. `op = Operator([[1,0,0,1]])`

- Operator is used to represent **matrix** operators acting on a quantum system.
- op = Operator ([[0,1], [1,0]])
- Converting classes to Operators
  - Pauli
  - Gate and Instruction
  - **QuantumCircuit**

Answer

Quantum operator

18. Which code fragment would yield an operator that represents a single-qubit X gate?

A. `op = Operator.Xop(0)`
B. `op = Operator([[0,1]])`
C.
```
qc = QuantumCircuit(1)
qc.x(0)
op = Operator(qc)
```
D. `op = Operator([[1,0,0,1]])`

- Operator is used to represent **matrix** operators acting on a quantum system.
- op = Operator ([[1,0], [0,1]])
- Converting classes to Operators
  - Pauli
  - Gate and Instruction
  - **QuantumCircuit**

Fidelity

19. What would be the fidelity result(s) for these two operators, which differ only by global phase?

```
op_a = Operator(XGate())
op_b = numpy.exp(1j * 0.5) * Operator(XGate())
```

A. `state_fidelity()` of 1.0
B. `state_fidelity()` and `average_gate_fidelity()` of 1.0
C. `average_gate_fidelity()` and `process_fidelity()` of 1.0
D. `state_fidelity()`, `average_gate_fidelity()` and `process_fidelity()` of 1.0

state_fidelity(*state1*, *state2*, *validate=True*) [SOURCE]

Return the state fidelity between two quantum states.

Source: https://qiskit.org/documentation/

Answer

Fidelity

19. What would be the fidelity result(s) for these two operators, which differ only by global phase?

```
op_a = Operator(XGate())
op_b = numpy.exp(1j * 0.5) * Operator(XGate())
```

A. `state_fidelity() of 1.0`
B. `state_fidelity() and average_gate_fidelity() of 1.0`
C. `average_gate_fidelity() and process_fidelity() of 1.0`
D. `state_fidelity(), average_gate_fidelity() and process_fidelity() of 1.0`

state_fidelity(*state1, state2, validate=True*) [SOURCE]

Return the state fidelity between two quantum states.

Source: https://qiskit.org/documentation/

Measurement

20. Given this code fragment, which output fits most closely with the measurement probability distribution?

```
qc = QuantumCircuit(2, 2)
qc.x(0)
qc.measure([0,1], [0,1])
simulator = Aer.get_backend('qasm_simulator')
result = execute(qc, simulator, shots=1000).result()
counts  = result.get_counts(qc)
print(counts)
```

A. {'00': 1000}
B. {'01': 1000}
C. {'10': 1000}
D. {'11': 1000}

$01 \Rightarrow q_1 q_0$

Answer

Measurement

20. Given this code fragment, which output fits most closely with the measurement probability distribution?

```
qc = QuantumCircuit(2, 2)
qc.x(0)
qc.measure([0,1], [0,1])
simulator = Aer.get_backend('qasm_simulator')
result = execute(qc, simulator, shots=1000).result()
counts  = result.get_counts(qc)
print(counts)
```

A. {'00': 1000}
B. {'01': 1000}
C. {'10': 1000}
D. {'11': 1000}

$01 \Rightarrow q_1 q_0$

Q & A