THE QUANTA CODE:

```
from qiskit import QuantumCircuit, transpile
from qiskit_aer import Aer
import numpy as np

def quantum_walk_circuit():
    qc = QuantumCircuit(3, 3)  # 3 qubits, 3 classical bits

    # Apply Hadamard gates to create superposition
    qc.h(0)
    qc.h(1)
    qc.h(2)

    # Apply CZ (Controlled-Z) gates to create entanglement
    qc.cz(0, 1)
    qc.cz(1, 2)
    qc.cz(0, 2)

    # Measure all qubits
    qc.measure([0, 1, 2], [0, 1, 2])

    return qc

# Create the quantum circuit
qc = quantum_walk_circuit()

# Use AerSimulator for execution
simulator = Aer.get_backend('aer_simulator')
```

```
compiled_circuit = transpile(qc, simulator)
result = simulator.run(compiled_circuit, shots=1024).result()

# Get measurement results
counts = result.get_counts()
print("Quantum Walk Measurement Results:")
print(counts)

# Verifier function to check results
def verify_quantum_walk(counts):
    threshold = 50
    valid_states = ["000", "111"]  # Expected valid states
    success = sum(counts.get(state, 0) for state in valid_states) > threshold
    return "Accepted" if success else "Rejected"

print("Verifier's Decision:", verify_quantum_walk(counts))
```

THE OUTPUT:

```
Quantum Walk Measurement Results:
{'100': 130, '101': 133, '110': 104, '000': 139, '010': 141, '001': 108, '111': 129, '011': 140}
Verifier's Decision: Accepted
```

```python
from qiskit import QuantumCircuit, transpile
from qiskit_aer import Aer
import numpy as np

def quantum_walk_circuit():
    qc = QuantumCircuit(3, 3)  # 3 qubits, 3 classical bits

    # Apply Hadamard gates to create superposition
    qc.h(0)
    qc.h(1)
    qc.h(2)

    # Apply CZ (Controlled-Z) gates to create entanglement
    qc.cz(0, 1)
    qc.cz(1, 2)
    qc.cz(0, 2)

    # Measure all qubits
    qc.measure([0, 1, 2], [0, 1, 2])

    return qc

# Create the quantum circuit
qc = quantum_walk_circuit()

# Use AerSimulator for execution
simulator = Aer.get_backend('aer_simulator')
compiled_circuit = transpile(qc, simulator)
result = simulator.run(compiled_circuit, shots=1024).result()

# Get measurement results
counts = result.get_counts()
print("Quantum Walk Measurement Results:")
```

```python
# Create the quantum circuit
qc = quantum_walk_circuit()

# Use AerSimulator for execution
simulator = Aer.get_backend('aer_simulator')
compiled_circuit = transpile(qc, simulator)
result = simulator.run(compiled_circuit, shots=1024).result()

# Get measurement results
counts = result.get_counts()
print("Quantum Walk Measurement Results:")
print(counts)

# Verifier function to check results
def verify_quantum_walk(counts):
    threshold = 50
    valid_states = ["000", "111"]  # Expected valid states
    success = sum(counts.get(state, 0) for state in valid_states) > threshold
    return "Accepted" if success else "Rejected"

print("Verifier's Decision:", verify_quantum_walk(counts))
```

```
Quantum Walk Measurement Results:
{'100': 130, '101': 133, '110': 104, '000': 139, '010': 141, '001': 108, '111': 129, '011': 140}
Verifier's Decision: Accepted
```