Pseudocode

Input ⇒ number of countermeasures (n), budget (b)

Create 1 qubit and 2 qubit Registers -- O(n)

Defining Countermeasure Probabilities -- O(n)

For generation = 0 to 9: -- O(1)

- For quantum_register = 0 to n: -- O(n)
  - classical_register defined -- O(1)
  - Quantum circuit created -- O(1)
  - Quantum circuit initialized with probability amplitude -- O(1)
  - Quantum Circuit measured -- O(1)
  - job = results from quantum computer -- O(1)
- For CISO = 1 to 4: -- O(1)
  - binaryString = " "
  - For result in job: -- O(n)
    - binaryString += (digit that was measured most frequently) -- O(1)
  - For Countermeasure in binaryString: -- O(n)
    - add Countermeasure to the knapsack -- O(1)
    - Check if Knapsack is over the budget -- O(1) ✱ budget = b
    - If out of budget, remove last ~~person~~ CM in the Knapsack -- O(1)
  - profit = The amount saved by the countermeasures implemented -- O(1)

Find the best individual -- O(1)

- Congregate quantum registers in preperation for manipulation -- O(n)
- Split binary string into array of countermeasures -- O(n)
- For countermeasure in binaryString array: -- O(n)
  - check if countermeasure qubit value is the same as the best individual -- O(1)
  - IF qubit value is different: -- O(1)
    - manipulate probability amplitudes of that quantum register to increase probability of other qubit values -- O(1)
  - Repeat process for all 4 CISO's in the population -- O(1)

Final Complexity Class -- O(n)