Self-Supervised Test

# Objective

Please read and provide the answer to the questions below before your in-person interview.

Candidates should expect more questions of this nature to be asked in in-person interviews.

This questionnaire usually should take less than ~2 hours to complete.

# Coding

Although some examples below are in Python, the candidate can select any other programming language of their choice.

## Question 1. Histogram

Q1a: Given a list of floats and histogram ranges, we need a histogram of the number of data items falling in that range.

### Example Usage

hist = Histogram([0, 5, 10])

# The input [0, 5, 10] defines the following ranges -

# (-inf, 0), [0, 5), [5, 10), [10, +inf).

hist.add\_numbers([1, 9, 12, 5, 6])

print(hist.histogram())  # Should print [0, 1, 3, 1]

hist.add\_numbers([4, 11])

print(hist.histogram())  # Should print [0, 2, 3, 2]

### Template with Public Methods

Please fill in the following template. Please add any method, variables, class etc. that can be helpful.

Note: Do *not* use pre-existing implementation of histogram (such as numpy.histogram).

Answer (should fill in the signature provided below) –

def Histogram:

  def \_\_init\_\_(self, boundaries: List[float]) -> None:

    pass

  def add\_numbers(self, numbers: List[float]) -> None:

    pass

  def histogram(self) -> List[float]:

    pass

Q1b: (Bonus Coding): How would you write tests for the Histogram class?

Answer (one example is provided, your answer should contain any new test cases) -

def test1():  # Example Test Case

  hist = Histogram([0, 5, 10])

  hist.add\_numbers([1, 9, 12, 5, 6])

  self.assertEqual(hist.histogram(), [0, 1, 3, 1])

def test2():  # Your Test Case(s)

  ...

## Question 2. Sudoku

Suppose we want to create a Sudoku Solving Robot, which with pen-and-paper solves puzzles given to it.

Q2a: Please design an API, with implementation, to take a completed Sudoku grid, and produce a boolean result indicating if the solution is correct or not.

In particular -

* What input data structure would you use to represent the Sudoku board?
* What should be the public methods for a client?

Q2b (Bonus Coding): How would you extend the API to not only return False for incorrect solutions, but also indicate why it is false?