Python Documentation – Questions 19 & 303

# NAME: ADEAGBO TIOLUWANIMI AYOMIDE MATRIC NO: CSC/2022/272

**DEPT: COMPUTER SCIENCE WITH MATHEMATICS**

**FACULTY: TECHNOLOGY**

# Question 19 – Print Only Even Numbers

IDE: PyCharm

## Algorithm

1. Create the list `numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`.  
2. Traverse the list with a `for` loop.  
3. Use `num % 2 == 0` to test evenness.  
4. Print the number if the test passes.

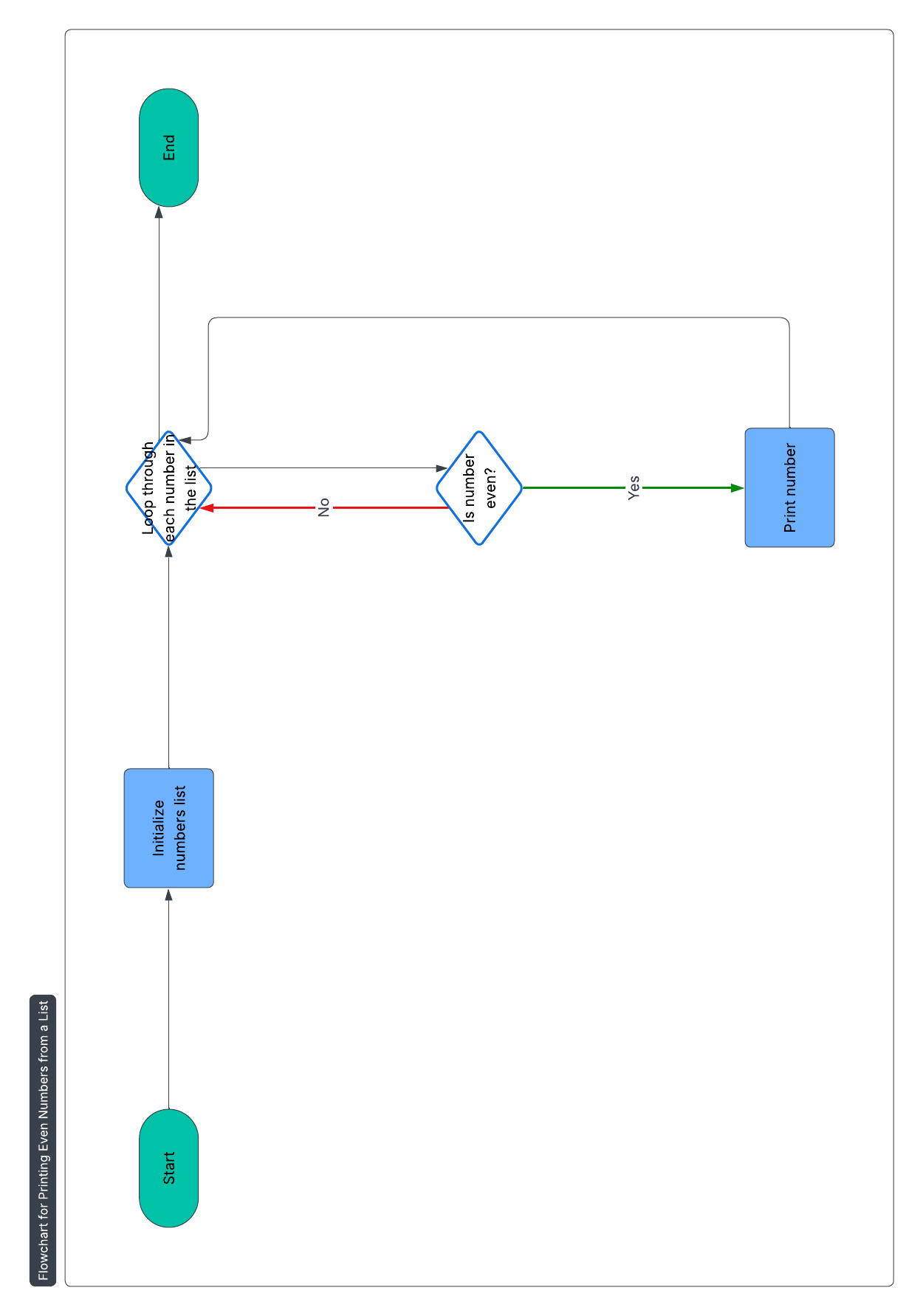
## Source Code

numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
for num in numbers:  
 if num % 2 == 0:  
 print(f"Even number: {num}")

## Output

Even number: 2  
Even number: 4  
Even number: 6  
Even number: 8  
Even number: 10

## Flowchart



Below is the flowchart representing the logic of Question 19.

## Reflection

A concise exercise in looping, list handling, and boolean logic. Using PyCharm’s live-run window makes it easy to see exactly which numbers pass the condition and to catch indentation mishaps quickly.

# Question 303 – Sample Class with Linear Regression (NumPy)

IDE: PyCharm

## Algorithm

1. Initialise Sample(name) → empty stress\_data & strain\_data.  
2. add\_data\_point(stress, strain) → appends paired data to the two lists.  
3. calculate\_youngs\_modulus():  
 - Confirm both lists are the same length.  
 - Require at least two points (otherwise regression is meaningless).  
 - Convert the lists to NumPy arrays.  
 - Perform first-degree linear regression with np.polyfit; the slope is Young’s Modulus (E).  
 - Return the slope (float).

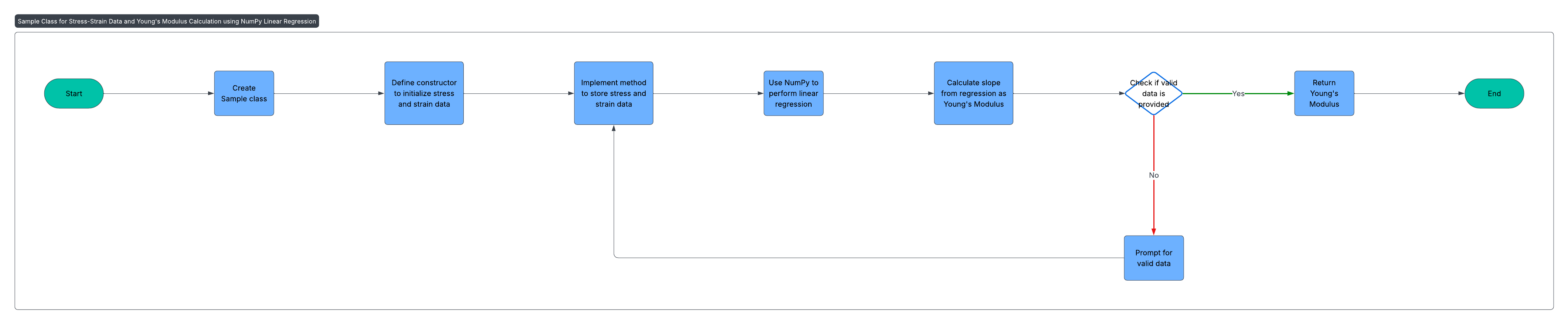
## Source Code

class Sample:  
 def \_\_init\_\_(self, name):  
 self.name = name  
 self.stress\_data = []  
 self.strain\_data = []  
  
 def add\_data\_point(self, stress, strain):  
 self.stress\_data.append(stress)  
 self.strain\_data.append(strain)  
  
 def calculate\_youngs\_modulus(self):  
 if len(self.stress\_data) != len(self.strain\_data):  
 raise ValueError("Stress and strain data points must be equal")  
  
 if len(self.stress\_data) < 2:  
 raise ValueError("At least two data points are required")  
  
 import numpy as np  
 stress\_array = np.array(self.stress\_data)  
 strain\_array = np.array(self.strain\_data)  
  
 youngs\_modulus = np.polyfit(strain\_array, stress\_array, 1)[0]  
 return youngs\_modulus  
  
# Example usage  
sample = Sample("Material X")  
sample.add\_data\_point(100, 0.01)  
sample.add\_data\_point(200, 0.02)  
sample.add\_data\_point(300, 0.03)  
  
youngs\_modulus = sample.calculate\_youngs\_modulus()  
print(f"Young's Modulus: {youngs\_modulus}")

## Output

Young's Modulus: 10000.000000000002

## Flowchart



## Reflection

Data validity is prioritized via explicit checks, ensuring accurate and meaningful computation.  
NumPy enables fast and reliable regression analysis, giving an engineering-level solution.  
PyCharm enhances productivity through its smart features like autocompletion and inline debugging.