

SUPERIOR UNIVERSITY LAHORE

GOLD CAMPUS

**NAME:** MUHAMMAD MOIZ RAZA

**ROLL NO:** SU92-BSAIM-F24-**061**

**PROGRAM:** ARTIFICIAL INTELLIGENCE

**SEMESTER:** 3rd SEMESTER

**SECTION:** BSAI-3A

**SUBJECT:** ARTIFICIAL INTELLIGENCE

**Submission Title:** LAB TASK-2

* FIZZ/BUZZ
* MOVIE BUDGET ANALYZER

**Submitted To: Sir RASHIK ALI**

**LAB TASK-2**

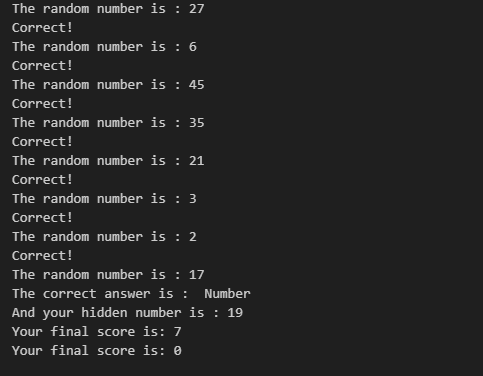
**TASK:** FIZZ/BUZZ

## **Why We Write This Code?**

## We write this code to make the FizzBuzz problem more fun and interesting. Instead of only checking numbers with math rules, this program turns it into a guessing game. It gives the player a chance to think and guess the answer, which makes learning enjoyable. The main idea is to practice coding and logic while also having fun at the same time. In short, it’s a way to learn programming in a playful way.

## **How the Code Works**

When the game starts, the computer picks a random number between 1 and 50. This number is added to the previous number to create a hidden total. The player does not see the hidden number but has to guess if it is fizz, buzz, fizzbuzz, or just a number. If the guess is correct, the score increases and the game continues to the next round. If the guess is wrong, the computer shows the right answer and asks if the player wants to keep playing or stop.

**OUTPUTS:**

**TASK:** MOVIE BUDGET ANALYZER

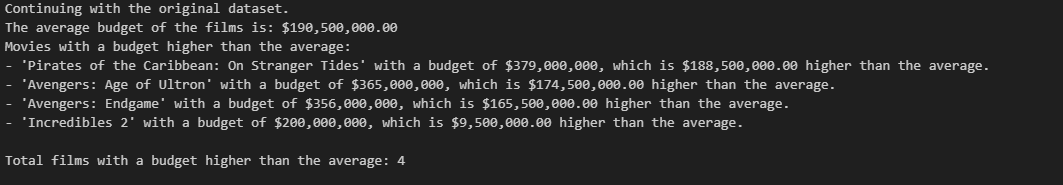
## **Why We Write This Code?**

We write this code to **analyze the budgets of movies**. Instead of just keeping movie names and numbers, the program helps us calculate the **average budget**, find which movies are **above average**, and see by how much. It also allows the user to **add new movies interactively**, making it flexible. In short, the purpose is to organize data and perform analysis in a structured way using Python.

## **How the Code Works**

The code creates a **class (MovieBudgetAnalyzer)** that stores movies and their budgets in a list. First, it asks the user if they want to add more movies. If yes, the user enters the movie name and budget, which get added to the list. Then the program calculates the **average budget** of all movies by summing the budgets and dividing by the total number of movies. After that, it goes through each movie using a **loop**, compares its budget to the average, and prints out the movies that are above average along with the difference. The code also uses **try/except** to handle invalid inputs safely. Finally, when executed, it runs these steps in sequence and prints a clear report.

**OUTPUTS:**



THE END