**Lab 2 – Mini Projects Explanation**

**Mini Project 1: FizzBuzz Game (Using Loops)**

**Overview**

This project is an interactive version of the classic FizzBuzz problem.  
Instead of the computer printing the entire sequence automatically, the player participates in the game.  
The player must correctly type **“Fizz”**, **“Buzz”**, or **“Fizz Buzz”** according to the divisibility rules.  
The game runs for **20 rounds** or ends when the player makes a mistake.

**Explanation**

1. The program uses a **while loop** that continues as long as the number is less than or equal to 20.
2. For each number, the program checks whether it is divisible by 3, 5, or both using the **modulo (%) operator**.
3. If divisible by 3 → print **“Fizz”**, if divisible by 5 → print **“Buzz”**, and if divisible by both → print **“Fizz Buzz”**.
4. The player must enter the correct response based on these rules.
5. If the answer is correct, the program increases the score and moves to the next number.
6. If the player enters the wrong answer, the program ends immediately and shows the correct one.
7. At the end, the player’s **total score** out of 20 is displayed.

**Learning Outcome**

This project helps you understand:

* **Loops** (while loop)
* **Conditionals** (if, elif, else)
* **User input handling**
* **Game logic and control flow**

It also introduces the concept of interactive programs where user responses affect program execution.

**Mini Project 2: Movie Budget Analysis**

**Overview**

In this project, we analyze a dataset of movies to calculate the **average budget** and find which movies have **higher-than-average budgets**.  
The dataset is stored as a **list of tuples**, where each tuple contains a movie’s name and its budget.  
The program also allows the user to **add new movies** before performing the calculations.

**Explanation**

1. **Data Structure:**  
   The movie data is stored as a list of tuples — each tuple has the movie name and its budget value.
2. **Adding New Movies:**  
   The user can choose to add more movies before running the analysis using a for loop and input() function.
3. **Average Budget Calculation:**  
   The program sums all movie budgets and divides the total by the number of movies to calculate the **average budget**.
4. **Identifying High-Budget Movies:**  
   A for loop checks which movies have a budget higher than the average.  
   It also calculates how much higher each movie’s budget is compared to the average.
5. **Counting Above-Average Movies:**  
   A counter variable keeps track of how many movies exceeded the average budget.
6. **Output:**
   * The average budget of all movies
   * The list of movies above average with their differences
   * The total count of above-average movies

**Learning Outcome**

Through this project, you learn:

* **Lists and tuples**
* **Loops for iteration**
* **Conditional checks**
* **Mathematical operations** (sum, average, difference)
* **Basic data analysis** concepts in Python

It also strengthens logic-building and real-world data processing skills.