

**BS Artificial Intelligence**

**Name:** Hafiz M. Muneeb Akbar

**Roll No:** SU92-BSAIM-F24-048

**Subject:** Artificial Intelligence (Lab)

**Lab Task 2**

**Mini Project 1**

**Fizz Buzz Game**

**Introduction**

In this lab task, I created a modified version of the classic Fizz Buzz program in Python. The purpose of this program is to practice loops, conditionals, random number generation, and user input handling. Unlike the traditional Fizz Buzz, this program takes a number from the user as input and also generates a random number for the computer. Both numbers are added together, and the sum is checked against the Fizz Buzz rules.

**Steps Explanation**

**Step 1: Setting the Rounds**

First, I created a variable **'rounds'** to define how many times the game will be played. In this version, it was set to 4 rounds.

**Step 2: Generating the Computer's Number**

Inside the loop, the computer generates a random number using the **randint** function from the random module. The number is chosen between 1 and 20 and displayed on the screen.

**Step 3: Taking User Input**

The program then asks the user to enter a number. The input is checked using **isNumeric()** to ensure that the user entered a valid number. If valid, the input is converted into an integer.

**Step 4: Adding the Numbers**

The user's number and the computer's number are added together to form a sum. This sum is then used to apply the Fizz Buzz rules.

**Step 5: Applying Fizz Buzz Rules**

The program checks the sum with the following rules:  
- If divisible by both 3 and 5, print **'Fizz Buzz'.**  
- If divisible by 3, print **'Fizz'.**  
- If divisible by 5, print **'Buzz'.**  
- Otherwise, print the sum itself.

**Step 6: Handling Invalid Input**

If the user enters something other than a number, the program displays 'Invalid input, please enter a number.' and continues running.

**Output Explanation**

During each round, the program shows the round number, the random number chosen by the computer, and the number entered by the user. It then prints the sum and the result according to the Fizz Buzz rules. If the input is invalid, it shows an error message instead.

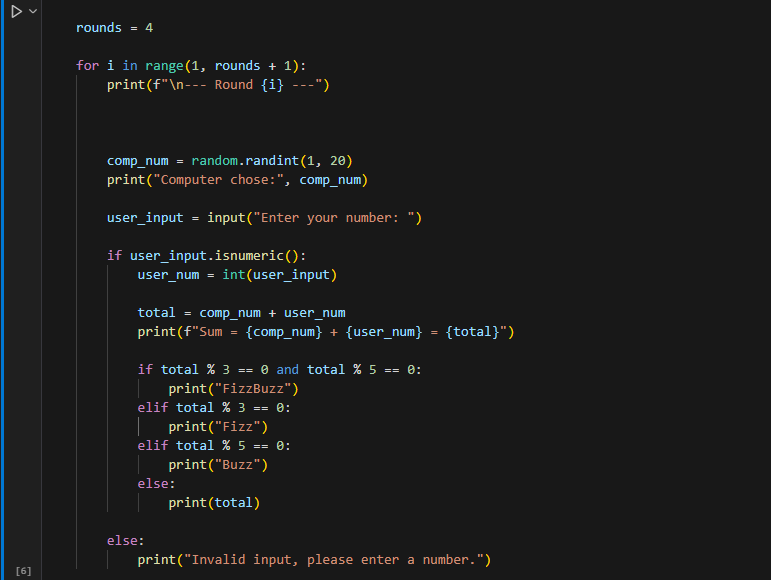


Figure 1: This shows the code that how we do multiple things here

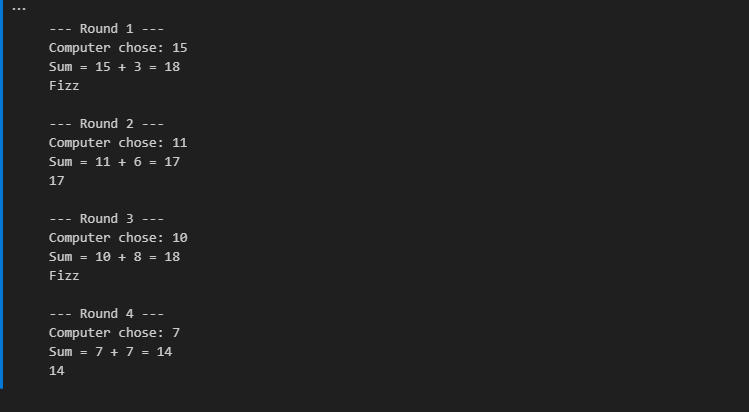


Figure 2: This shows the output of the game.

**Mini Project 2**

**Movie Budget Analysis Program**

**Introduction**

In this lab task, I created a Movie Budget Analysis Program in Python. The purpose of this program is to calculate the average budget of movies, identify the movies that exceed the average budget, and count how many movies spent more than the average. This helps in learning the use of lists, tuples, loops, conditionals, and basic calculations in Python.

# Steps Explanation

**Step 1: Creating the Movie List**

First, I created a list of tuples called 'movies'. Each tuple contains two values: the movie name and its budget. This forms the dataset for the program.

**Step 2: Extra Challenge (Adding More Movies)**

As an extra challenge, the program allows the user to add more movies before running the analysis. The program asks how many movies the user wants to add. Using a for loop with range, it then collects the movie name and budget as input from the user and appends this data as a tuple to the **'movies'** list.

**Step 3: Calculating the Average Budget**

To calculate the average budget, I initialized a variable **'total'** as 0 and used a for loop to add all movie budgets. Then I divided the total by the number of movies using **len(movies)** to find the average.

**Step 4: Identifying Above Average Movies**

Next, I **looped** through the movies again to check which ones had a budget greater than the average. For each above-average movie, the program also calculates how much higher its budget is compared to the average and displays the **result.**

**Step 5: Counting Above Average Movies**

While checking the movies, I maintained a **counter variable**. Each time a movie exceeded the average, the counter increased by **1**. At the end, the program prints how many movies had budgets above the average.

**Step 2: Extra Challenge (Adding More Movies)**

As an extra challenge, the program allows the user to add more movies before running the analysis. The program asks how many movies the user wants to add. Using a for loop with range, it then collects the movie name and budget as input from the user and appends this data as a tuple to the **'movies'** list.

**Output Explanation**

The program first displays the average budget. Then it prints all movies whose budgets are above the average, along with the difference. It also shows how many movies exceeded the average budget. If the user adds more movies, they are included in the calculations.

A computer screen shot of a program

AI-generated content may be incorrect.

Figure 3: It shows the list of the movies, input to add more movies and to find the Average.

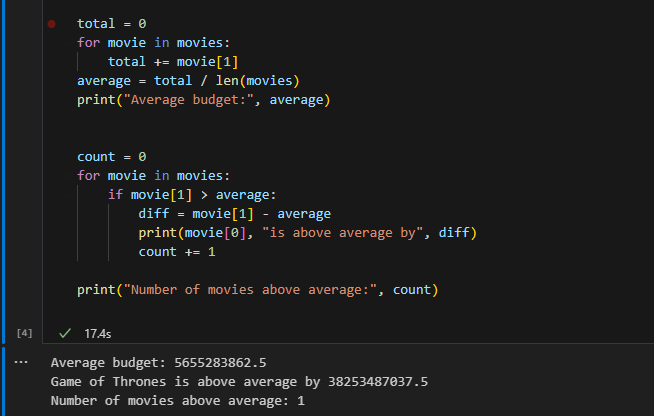


Figure 4: this shows the output