# How can educational institutions help tackle the issue of urbanisation. An interview with Mr Hammad Noor AHMED from Superior University in Lahore | by ASEFEdu (Editor) | ASEFEdu (Blog) | Medium

# Name : Tahir Fareed

# Roll No: SU92-BSAIM-F24-036

# Section: BSAI (3A)

# Subject: AI Lab

# Task : 02

# Number Addition with FizzBuzz

# Python Program Documentation: Number Addition with FizzBuzz

This document explains a simple Python program step by step. The program loops through numbers from 1 to 100, adds each number to the previous one, and applies a modified FizzBuzz logic.

## Python Code

print(" Lets Begin \n")  
  
previous\_number = 0   
  
for number in range(1, 101):   
 print(f" Number : {number}")  
  
 if previous\_number == 0:  
 print("No previous number\n")  
 previous\_number = number  
 continue  
  
 total = previous\_number + number  
 print(f"Total = {previous\_number} + {number} = {total}")  
  
 if total % 3 == 0:  
 print("Fizz\n")  
 elif total % 5 == 0:  
 print("Buzz\n")  
 elif total % 3 == 0 and total % 5 == 0:  
 print("FizzBuzz\n")  
 else:  
 print(f"{total}\n")

## Explanation (Step by Step)

1. The program starts with printing 'Lets Begin'.  
2. A variable called `previous\_number` is set to 0.  
3. The program uses a loop to go through numbers from 1 to 100.  
4. For each number:  
 - If there is no previous number (first loop), it prints 'No previous number'.  
 - Otherwise, it adds the current number to the previous number and prints the total.  
5. Then the program checks the total:  
 - If divisible by 3 → prints 'Fizz'.  
 - If divisible by 5 → prints 'Buzz'.  
 - If divisible by both 3 and 5 → prints 'FizzBuzz'.  
 - If not divisible by 3 or 5 → prints the total number itself.  
6. This continues until the loop reaches 100.

## Sample Output (First Few Iterations)

Number : 1  
 No previous number  
  
 Number : 2  
 Total = 1 + 2 = 3  
 Fizz  
  
 Number : 3  
 Total = 2 + 3 = 5  
 Buzz  
  
 Number : 4  
 Total = 3 + 4 = 7  
 7

## Detailed Explanation

This program is a learning exercise that demonstrates how to use loops, variables, and conditions in Python. Below is a more detailed explanation of each part:  
  
1. \*\*Printing a Start Message\*\*:  
 - The program begins by printing 'Lets Begin'. This is just to mark the start of the program.  
  
2. \*\*Variable Initialization\*\*:  
 - A variable named `previous\_number` is created and set to 0. This will store the number from the last loop iteration, so it can be added with the current number.  
  
3. \*\*For Loop (1 to 100)\*\*:  
 - The program uses a for loop: `for number in range(1, 101):`  
 - This means the loop will run 100 times, with `number` taking values from 1 to 100.  
  
4. \*\*First Iteration Check\*\*:  
 - When the loop starts with number = 1, the program checks if `previous\_number == 0`.  
 - Since it is 0, the program prints 'No previous number' and then updates `previous\_number` to 1.  
 - This avoids adding 0 with 1 unnecessarily.  
  
5. \*\*Adding Numbers\*\*:  
 - From the second iteration onward, the program adds the current number with the previous number.  
 - Example: when number = 2, previous\_number = 1, so total = 1 + 2 = 3.  
  
6. \*\*FizzBuzz Logic\*\*:  
 - Once the sum is calculated, the program applies divisibility checks:  
 • If total is divisible by 3 → prints 'Fizz'.  
 • If total is divisible by 5 → prints 'Buzz'.  
 • If total is divisible by both 3 and 5 → prints 'FizzBuzz'.  
 • Otherwise, it just prints the total number.  
 - Example: 3 → Fizz, 5 → Buzz, 15 → FizzBuzz.  
  
7. \*\*Updating the Previous Number\*\*:  
 - At the end of each loop, the current number becomes the new `previous\_number`.  
 - This allows the process to continue correctly in the next iteration.  
  
8. \*\*Loop Continues\*\*:  
 - The program repeats this logic until number = 100, showing the total and FizzBuzz outputs.  
  
This program is a great practice for beginners to understand:  
- How loops work in Python.  
- How variables can store values between loop iterations.  
- How conditional statements control program behavior.  
- How simple math rules like divisibility can be applied in code.

**Screenshot:**

