

**Lab # 02**

**Student Name: Sahil kumar**

**Roll Number: BIT-23F-006**

**Section: A**

**Subject: Artificial Intelligence**

**Department: BS Information Technology**

**GETTING STARTED WITH PYTHON**

**Lab 02. Introduction to Python**

The Python interpreter is the program that executes Python code. It reads and executes Python scripts or commands line by line.

**Objectives: To get to know about the Variables, Data types, Logical operators (And OR NOT)**

**Variables in python**

Variables in Python are used to store data values. They act as labels that refer to locations in memory where data is stored.

1. Naming: Variable names in Python can consist of letters (both uppercase and lowercase), digits, and underscores. However, they must start with a letter or an underscore. Variable names are case-sensitive.

2. Assignment: Variables are assigned values using the assignment operator `=`. For example:

**x = 10**

**name = "Alice"**

3. Dynamic Typing: Python is dynamically typed, meaning you don't need to declare the type of a variable before assigning a value to it. The type of the variable is inferred based on the value assigned to it. For example:

**x = 10 # x is an integer**

**name = "Alice" # name is a string**

4. Reassignment: Variables can be reassigned to different values of any type. For example:

**x = 10**

**x = "Hello"**

5. Memory Management: Python automatically handles memory allocation and reallocation for variables through its garbage collector.

6. Variable Types: Python supports various data types for variables, including integers, floats, strings, and lists, tuples, dictionaries, sets, and more. Variables can hold values of any of these types.

7. Scope: The scope of a variable determines where in the code that variable can be accessed.

**Data types In Python**

**Lab 2. Variables to Python**

1. Make program to see if value is true of false.
2. Make a calculator using the operands in the class
3. Write programs for the different type string “”,’’,””””””

**Lab 2. Variables to Python**

01: Make program to see if value is true of false:

Code:

def check\_boolean(value):

    if value:

        return "The value is True."

    else:

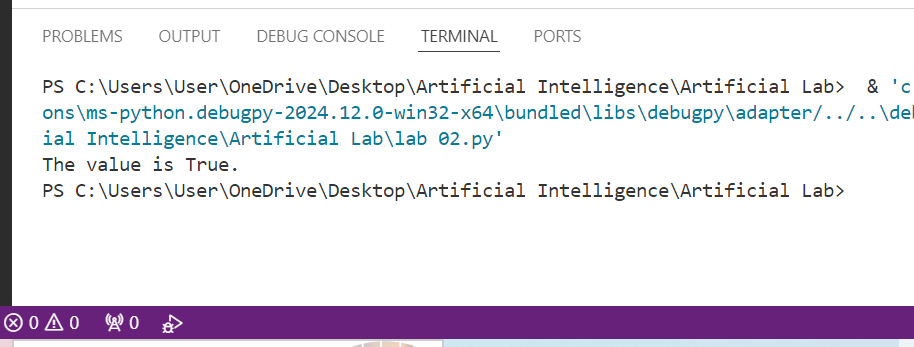
        return "The value is False."

# Example usage

value = True  # Change this to any value to test

print(check\_boolean(value))

output:



02: Make a calculator using the operands in the class:

Code:

class Calculator:

    def add(self, a, b):

        return a + b

    def subtract(self, a, b):

        return a - b

    def multiply(self, a, b):

        return a \* b

    def divide(self, a, b):

        if b == 0:

            return "Error: Division by zero is undefined."

        return a / b

# Example usage

calc = Calculator()

# Perform operations

num1 = 76

num2 = 89

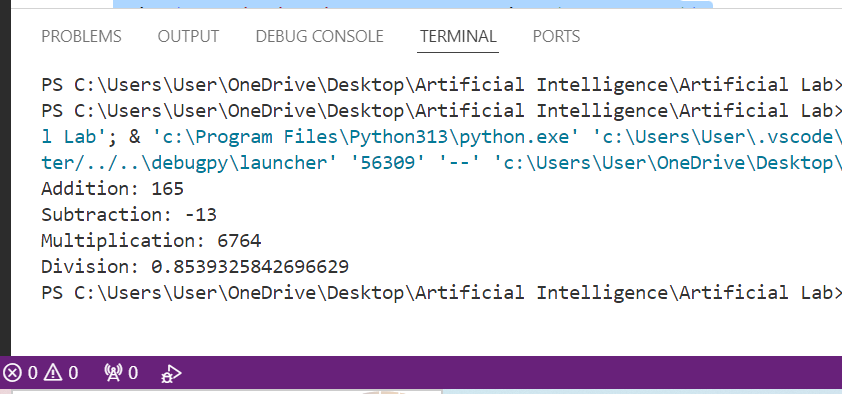
print("Addition:", calc.add(num1, num2))

print("Subtraction:", calc.subtract(num1, num2))

print("Multiplication:", calc.multiply(num1, num2))

print("Division:", calc.divide(num1, num2))

output:



03: Write programs for the different type string “”,’’,””””””:

Code:

# Using Double Quotes

string\_double\_quotes = " 01: This is a string with double quotes."

string\_with\_single\_quote = " 02: It's easy to use double quotes when there's an apostrophe."

# Using Single Quotes

string\_single\_quotes = '01: This is a string with single quotes.'

string\_with\_double\_quote = '02: She said, "Hello!" with a smile.'

# Using Triple Quotes

multi\_line\_string = """ 01: This is a multi-line string.

You can write text over multiple lines without any issues.

It's very helpful for long paragraphs or text blocks."""

string\_with\_both\_quotes = """ 02: She said, "It's a beautiful day!" and smiled."""

# Print all strings

print("Double Quotes Example:")

print(string\_double\_quotes)

print(string\_with\_single\_quote)

print("\nSingle Quotes Example:")

print(string\_single\_quotes)

print(string\_with\_double\_quote)

print("\nTriple Quotes Example:")

print(multi\_line\_string)

print(string\_with\_both\_quotes)

Output:

