21 May **Python Basic - 2**

**Q.1. Create two int type variables, apply addition, subtraction, division and multiplications and store the results in variables. Then print the data in the following format by calling the variables:**

**First variable is \_\_ & second variable is \_\_.**

**Addition: \_\_ + \_\_ = \_\_**

**Subtraction: \_\_ - \_\_ = \_\_**

**Multiplication: \_\_ \* \_\_ = \_\_**

**Division: \_\_ / \_\_ = \_\_**

first\_variable = 10

second\_variable = 5

addition\_result = first\_variable + second\_variable

subtraction\_result = first\_variable - second\_variable

multiplication\_result = first\_variable \* second\_variable

division\_result = first\_variable / second\_variable

# Print the data in the specified format

print(f"First variable is {first\_variable} & second variable is {second\_variable}.")

print(f"Addition: {first\_variable} + {second\_variable} = {addition\_result}")

print(f"Subtraction: {first\_variable} - {second\_variable} = {subtraction\_result}")

print(f"Multiplication: {first\_variable} \* {second\_variable} = {multiplication\_result}")

print(f"Division: {first\_variable} / {second\_variable} = {division\_result}")

**Q.2. What is the difference between the following operators:**

1. **‘/’ & ‘//’**
2. **‘\*\*’ & ‘^’**

- `'/'` is the division operator in Python, and it performs regular floating-point division. It calculates the quotient of the division and returns a floating-point result.

- `'//'` is the floor division operator in Python. It calculates the quotient of the division, but rounds down to the nearest integer, returning an integer result.

regular\_division = a / b # Result: 3.333333...

floor\_division = a // b # Result: 3

- `'\*\*'` is the exponentiation operator in Python. It raises the left operand to the power of the right operand.

- `' ^ '` is not an exponentiation operator in Python. Instead, it's the bitwise XOR operator, which performs a bitwise exclusive OR operation between two integers.

exponentiation = x \*\* y # Result: 8

bitwise\_xor = x ^ y # Result: 1

* 1. **List the logical operators.**

The logical operators are used to perform logical operations on Boolean values or expressions

1. `and`: Returns `True` if both operands are `True`, otherwise returns `False`.

2. `or`: Returns `True` if at least one of the operands is `True`, otherwise returns `False`.

3. `not`: Returns the opposite Boolean value of the operand; if `True`, it returns `False`, and if `False`, it returns `True`.

* 1. **Explain right shift operator and left shift operator with examples.**

The right shift operator (`>>`) shifts the bits of a number to the right by a specified number of positions.

x = 16 # Binary: 10000

y = x >> 2

print(y) # Output: 4 (Binary: 100)

The left shift operator (`<<`) shifts the bits of a number to the left by a specified number of positions.

x = 4 # Binary: 100

y = x << 3

print(y) # Output: 32 (Binary: 100000)

* 1. **Create a list containing int type data of length 15. Then write a code to check if 10 is present in the list or not.**

# Create a list of integers

my\_list = [5, 8, 2, 10, 15, 6, 12, 18, 7, 3, 20, 25, 9, 14, 30]

# Check if 10 is present in the list

if 10 in my\_list:

print("10 is present in the list.")

else:

print("10 is not present in the list.")