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:

**Code:**

a = 10

b = 5

addition = a + b

subtraction = a - b

division = a / b

multiplication = a \* b

print("Addition: {}".format(addition))

print("Subtraction: {}".format(subtraction))

print("Division: {}".format(division))

print("Multiplication: {}".format(multiplication))

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

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

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

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

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

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

1. ‘/’ & ‘//’

**Difference between '/' and '//':**

The / operator is the division operator in Python and performs normal division, resulting in a float value. For example, 5 / 2 will give the result 2.5.

The // operator is the floor division operator in Python and performs integer division, discarding the decimal part of the result. For example, 5 // 2 will give the result 2.

1. ‘\*\*’ & ‘^’

**Difference between '\*\*' and '^':**

The \*\* operator is the exponentiation operator in Python and is used to raise a number to a power. For example, 2 \*\* 3 will give the result 8 (2 raised to the power of 3).

The ^ operator is not used for exponentiation in Python. Instead, it is the bitwise XOR operator.

* 1. List the logical operators.

The logical operators are:

**and:** The and operator returns True if both operands are True, and False otherwise.

**or:** The or operator returns True if at least one of the operands is True, and False if both operands are False.

**not:** The not operator is a unary operator that negates the value of the operand. It returns True if the operand is False, and False if the operand is True.

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

**Right Shift Operator (>>):** The right shift operator shifts the bits of a number to the right by a specified number of positions. It effectively divides the number by 2 for each right shift by one position. The rightmost bit is discarded, and the leftmost bit is filled with the sign bit (0 for non-negative numbers, 1 for negative numbers)

**Left Shift Operator (<<):** The left shift operator shifts the bits of a number to the left by a specified number of positions. It effectively multiplies the number by 2 for each left shift by one position. The rightmost positions are filled with zeros.

* 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.

**Code:**

my\_list = [2, 5, 8, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65]

if 10 in my\_list:

print("10 is present in the list")

else:

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