ASSIGNMENT 7 Solution Python Basic - 2

Submitted by: Sweta Dhara

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 \_\_.

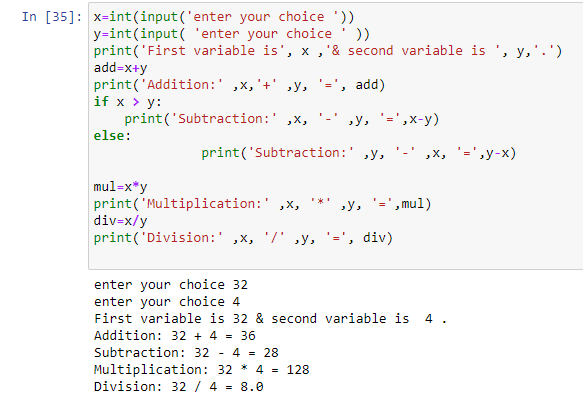
Addition: \_\_ + \_\_ = \_\_

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

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

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

A:



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

(i) ‘/’ & ‘//’

(ii) ‘\*\*’ & ‘^’

A: In Python:

(i) '/ ' and '//' Operators:

The '/' operator is the division operator. It performs normal division and returns the quotient as a floating-point number. For example, 5 / 2 would result in 2.5.

The '//' operator is the floor division operator. It performs integer division and returns the quotient as an integer, rounding down towards negative infinity. For example, 5 // 2 would result in 2, as the fractional part is discarded.

(ii) '\*\*' and '^' Operators:

The '\*\*' operator is the exponentiation operator. It raises the left operand to the power of the right operand. For example, 2 \*\* 3 would result in 8 (2 raised to the power of 3).

The '^' operator is not used for exponentiation in Python. Instead, it is the bitwise XOR operator. It performs the bitwise XOR operation on the binary representation of two numbers. For example, 2 ^ 3 would result in 1 (bitwise XOR of 2 and 3).

Q.3. List the logical operators.

A: In Python, the logical operators are:

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

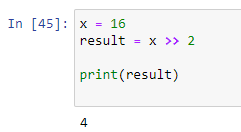
OR: The or operator returns True if at least one of the operands is True, otherwise it returns False.

NOT: The not operator is a unary operator that returns the opposite of the operand's logical value. If the operand is True, not returns False, and if the operand is False, not returns True.

Q.4. Explain right shift operator and left shift operator with examples.

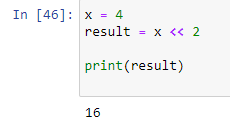
A: In Python, the right shift (>>) and left shift (<<) operators are used for shifting the bits of a number to the right or left, respectively. These operators are commonly used in bitwise operations to manipulate binary representations of numbers. Let's explore each operator with examples:

Right Shift (>>) Operator: The right shift operator shifts the bits of a number to the right by a specified number of positions. It is equivalent to dividing the number by 2 raised to the power of the specified shift value.



In the example above, the binary representation of x is 10000. Shifting it to the right by 2 positions results in 100, which is the binary representation of decimal number 4.

Left Shift (<<) Operator: The left shift operator shifts the bits of a number to the left by a specified number of positions. It is equivalent to multiplying the number by 2 raised to the power of the specified shift value.



In the example above, the binary representation of x is 100. Shifting it to the left by 2 positions results in 10000, which is the binary representation of decimal number 16.

The right shift and left shift operators are often used in bitwise operations, such as manipulating binary flags, performing efficient multiplication or division by powers of 2, or extracting specific bits from a binary representation.

Q.5. 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.

A:

