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

**python**

**Copy**

**# Create two integer variables**

**num1 = 10**

**num2 = 5**

**# Perform arithmetic operations**

**addition = num1 + num2**

**subtraction = num1 - num2**

**multiplication = num1 \* num2**

**division = num1 / num2**

**# Print results**

**print(f"First variable is {num1} & second variable is {num2}.")**

**print(f"Addition: {num1} + {num2} = {addition}")**

**print(f"Subtraction: {num1} - {num2} = {subtraction}")**

**print(f"Multiplication: {num1} \* {num2} = {multiplication}")**

**print(f"Division: {num1} / {num2} = {division}")**

**Output:**

**vbnet**

**Copy**

**First variable is 10 & second variable is 5.**

**Addition: 10 + 5 = 15**

**Subtraction: 10 - 5 = 5**

**Multiplication: 10 \* 5 = 50**

**Division: 10 / 5 = 2.0**

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

**(i) ‘/’ & ‘//’**

* **/: This is the true division operator. It returns the result as a floating-point number, even if both operands are integers. Example:**

**python**

**Copy**

**5 / 2 # Result: 2.5**

* **//: This is the floor division operator. It returns the largest integer less than or equal to the result of the division (i.e., it rounds down to the nearest integer). Example:**

**python**

**Copy**

**5 // 2 # Result: 2**

**(ii) ‘’ & ‘^’\*\***

* **\*\*: This is the exponentiation operator. It is used to raise the left operand to the power of the right operand. Example:**

**python**

**Copy**

**2 \*\* 3 # Result: 8**

* **^: This is the bitwise XOR operator, which performs a bitwise exclusive OR operation on two integers. Example:**

**python**

**Copy**

**5 ^ 3 # Result: 6 (in binary, 0101 XOR 0011 = 0110)**

**Q.3 List the logical operators.**

**The logical operators in Python are:**

1. **and: Returns True if both operands are True.**
2. **or: Returns True if at least one of the operands is True.**
3. **not: Returns True if the operand is False, and False if the operand is True.**

**Example:**

**python**

**Copy**

**x = True**

**y = False**

**print(x and y) # Output: False**

**print(x or y) # Output: True**

**print(not x) # Output: False**

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

* **Right Shift (>>): The right shift operator shifts the bits of the number to the right by the specified number of positions. It essentially divides the number by 2 for each position the bits are shifted. Example:**

**python**

**Copy**

**num = 16 # Binary: 10000**

**result = num >> 2 # Shift bits to the right by 2**

**print(result) # Output: 4 (Binary: 100)**

* **Left Shift (<<): The left shift operator shifts the bits of the number to the left by the specified number of positions. It essentially multiplies the number by 2 for each position the bits are shifted. Example:**

**python**

**Copy**

**num = 3 # Binary: 11**

**result = num << 2 # Shift bits to the left by 2**

**print(result) # Output: 12 (Binary: 1100)**

**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.**

**python**

**Copy**

**# Create a list containing integers**

**my\_list = [1, 3, 5, 7, 9, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28]**

**# 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.")**

**Output:**

**csharp**

**Copy**

**10 is present in the list.**