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 = int(input("enter first variable : "))
  second_variable = int(input("enter second variable : "))


  addition = first_variable + second_variable
  subtraction = first_variable - second_variable
  multiplication = first_variable * second_variable
  division = first_variable / second_variable


  print("First variable is ",first_variable,"& second variable is ",second_variable )
  print("Addition:",first_variable,"+",second_variable, "=",addition)
  print("Subtraction:" ,first_variable,"-",second_variable,"=",subtraction)
  print("Multiplication:",first_variable,"*",second_variable,"=",multiplication)
  print("Division:",first_variable,"/",second_variable,"=",division)
  ```

Q.2. What is the difference between the following operators: (i) '/' & '//' (ii) '' & '^'**

- The operators '/' and '//' are both used to divide two numbers, but they have different results. The '/' operator returns the quotient of the two numbers, while the '//' operator returns the integer part of the quotient.
- For example, 5 / 2 = 2.5, but 5 // 2 = 2.
- The operators '**' and '^' are both used to raise a number to a power, but they have different results. The '**' operator uses exponentiation, which means that it multiplies the number by itself the specified number of times. The '^' operator uses bitwise exponentiation, which means that it shifts the bits of the number to the left the specified number of times.
- For example, 2 ** 3 = 8, but 2 ^ 3 = 10.

Q.3. List the logical operators.

- And operator
- Or operator
  ```

- Not operator

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

- The right shift operator (>>) shifts the bits of a number to the right by the specified number of places. The left shift operator (<<) shifts the bits of a number to the left by the specified number of places.
- For example, if we have the number 10, which is represented in binary as 00001010, and we shift it to the right by 1 place, we get the number 5, which is represented in binary as 00000101. This is because the rightmost bit is shifted off and the leftmost bit is filled with a 0.
- If we shift the number 10 to the left by 1 place, we get the number 20, which is represented in binary as 00010100. This is because the leftmost bit is filled with a 0 and the rightmost bit is shifted off. Here are some more examples:
- 10 >> 1 = 5
- 10 << 1 = 20
- 10 >> 2 = 2
- 10 << 2 = 40
- 10 >> 3 = 1
- 10 << 3 = 80

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.

- list1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]

```
if 10 in list1:
   print("10 is present in the list")
else:
   print("10 is not present in the list")
```