Assignment No-7

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: __ / __ = __
Solution: int firstVariable = 5;
int secondVariable = 3;

int addition = firstVariable + secondVariable;
int subtraction = firstVariable - secondVariable;
int multiplication = firstVariable * secondVariable;
int division = firstVariable / secondVariable;

System.out.println("First variable is " + firstVariable + " & second variable is " + secondVariable);
System.out.println("Addition: " + firstVariable + " + " + secondVariable + " = " + addition);
System.out.println("Subtraction: " + firstVariable + " - " + secondVariable + " = " + subtraction);
System.out.println("Multiplication: " + firstVariable + " * " + secondVariable + " = " + multiplication);
System.out.println("Division: " + firstVariable + " / " + secondVariable + " = " + division);

Q.2. What is the difference between the following operators:
(i) '/' & '//'
(ii) '' & '^'**
Solution: (i) The operators '/' and '//' are used for division in programming languages but have different behaviors.

- '/' is the division operator that performs normal division. Regardless of the operands ' data types, it returns the quotient as a floating-point number.

For example:
5 / 2 = 2.5 (floating-point division)

- '//' is the floor division operator. It returns the quotient as an integer, discarding any fractional part.

For example:
5 // 2 = 2 (floor division)

(ii) The operators '**' and '^' are used for exponentiation but have different behaviors.

- '**' is the exponentiation operator in many programming languages, including Python. It raises the left operand to the power of the right operand.

For example:
2 ** 3 = 8 (2 raised to the power of 3)
- '^' is sometimes used to represent exponentiation in mathematical notations, but it is not a standard operator in most programming languages. In some languages (like MATLAB), '^' is used for exponentiation, but in many other languages, it has different meanings or is not recognized as an operator.

Q.3. List the logical operators.

Solution: a) AND operator (&&)

b) OR operator (||)

c) NOT operator (!)

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

Solution: The right shift operator (>>) and the left shift operator (<<) are bitwise operators used for shifting the bits of a number to the right or left, respectively. These operators are commonly used in programming languages like C++, Java, and Python.

The right shift operator (>>):
When the right shift operator is applied to a number, it shifts all the bits of the number to the right by a specified number of positions. The shifted positions are filled with 0s on the left side. The syntax for the right shift operator is: x >> y, where x is the number to be shifted and y is the number of positions to shift.

Example:

Let's say the binary representation of the number x is 110101. When we perform x >> 2, the bits are shifted two positions to the right and two 0s are filled in from the left side. So, the result would be 001101.

In Java, the right shift operator can also be used to divide a number by 2. For example, if we have a number x = 10, x >> 1 would yield 5.

The left shift operator (<<):

Similar to the right shift operator, the left shift operator shifts all the bits of a number to the left by a specified number of positions. However, the shifted positions are filled with 0s on the right side. The syntax for the left shift operator is: x << y, where x is the number to be shifted and y is the number of positions to shift.

Example:

If we have a number x with the binary representation 110101, performing x << 2 would shift the bits two positions to the left and fill in 0s on the right side. The result would be 010100.

In some programming languages like Java, the left shift operator can also be used as a multiplication by 2. For example, if x = 5, x << 1 would yield 10.

To summarize, the right shift operator (>>) shifts the bits of a number to the right, filling in 0s from the left side, while the left shift operator (<<) shifts the bits to the left, filling in 0s on the right side.

Q.5. Create a list containing int-type data of length 15. Then, write a code to check whether 10 is in the list.

Solution: Here is an example code that creates a list of integers of length 15 and checks if 10 is present in the list:

```
# Step 1: Create a list of integers

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

# Step 2: 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")
```

I created a list `my_list` containing 15 integers in this code. Then I used an `if` statement to check if 10 is present in the list. If it is, the code prints "10 is present in the list"; otherwise, it prints "10 is not present in the list".