1.

first\_variable=int(input(“Enter first number: ”)

second\_variable=int(input(“Enter second number: ”)

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)

2. (i) The ‘/’ operator is used for division, and it returns a floating value as an answer. For example, 5 / 2 would result in 2.5.

The ‘//’ is also used for division but it returns the whole number as an answer and ignores the point value. For example, 5 // 2 would result in 2.

(ii) The \*\* operator is used to raise a number to a power. For example, 2 \*\* 3 would result in 8, which is 2 raised to the power of 3.

the ^ operator is the bitwise XOR (exclusive OR) operator. It performs a bitwise XOR operation on the binary representations of two numbers. For example, 2 ^ 3 would result in 1, which is the bitwise XOR of the binary representations of 2 and 3 (10 ^ 11).

3. The logical operators in Python are:

and: Returns True if both operands are True.

or: Returns True if at least one of the operands is True.

not: Returns the opposite Boolean value of the operand.

4. The right shift (>>) and left shift (<<) operators are bitwise shift operators. They shift the bits of a number to the right or left by a specified number of positions.

The right shift operator (>>) shifts the binary representation of a number to the right by the specified number of positions. The rightmost bits are discarded, and the leftmost bits are filled with the sign bit (0 for positive numbers, 1 for negative numbers). For example, 5 >> 1 would result in 2, which is the binary representation of 5 (101) shifted one position to the right.

The left shift operator (<<) shifts the binary representation of a number to the left by the specified number of positions. The rightmost bits are filled with zeroes. For example, 5 << 1 would result in 10, which is the binary representation of 5 (101) shifted one position to the left.

5. my\_list = [1, 5, 7, 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")