**Demonstrate the use of Arithimetic Operation in python.**

**Code :-**

a = 10

b = 3

# Addition

addition = a + b

print(f"Addition: {a} + {b} = {addition}")

# Subtraction

subtraction = a - b

print(f"Subtraction: {a} - {b} = {subtraction}")

# Multiplication

multiplication = a \* b

print(f"Multiplication: {a} \* {b} = {multiplication}")

# Division

division = a / b

print(f"Division: {a} / {b} = {division}")

# Floor Division

floor\_division = a // b

print(f"Floor Division: {a} // {b} = {floor\_division}")

# Modulus

modulus = a % b

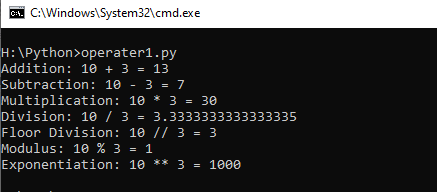
print(f"Modulus: {a} % {b} = {modulus}")

# Exponentiation

exponentiation = a \*\* b

print(f"Exponentiation: {a} \*\* {b} = {exponentiation}")

**Output :-**



**Demonstrate the use of Relational Operation in python.**

**Code :-**

# Define two numbers

a = 10

b = 5

# Equal to

equal = (a == b)

print(f"{a} == {b}: {equal}")

# Not equal to

not\_equal = (a != b)

print(f"{a} != {b}: {not\_equal}")

# Greater than

greater\_than = (a > b)

print(f"{a} > {b}: {greater\_than}")

# Less than

less\_than = (a < b)

print(f"{a} < {b}: {less\_than}")

# Greater than or equal to

greater\_equal = (a >= b)

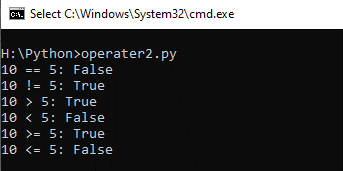
print(f"{a} >= {b}: {greater\_equal}")

# Less than or equal to

less\_equal = (a <= b)

print(f"{a} <= {b}: {less\_equal}")

**Output** :-



**Demonstrate the use of Logical Operation in python.**

**Code** :-

a = 10

b = 20

# Check the AND conditions

if a > 5 and b > 15:

print("Both conditions are True.")

else:

print("At least one condition is False.")

# Check the OR conditions

if a > 15 or b < 10:

print("At least one condition is True.")

else:

print("Both conditions are False.")

# Check the NOT conditions

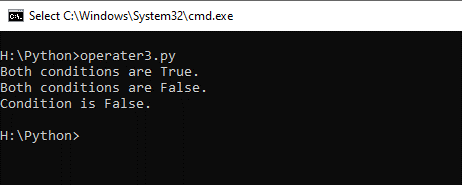
if not a > 15:

print("Condition is False.")

else:

print("Condition is True.")

**Output** :-



**Demonstrate the use of Bit wise Operation in python.**

**Code** : -

a = 12 # In binary: 1100

b = 5 # In binary: 0101

# Check the Bitwise AND (&) conditions

result = a & b

print(f"{a} & {b} = {result}")

# Check the Bitwise OR (|) conditions

result = a | b

print(f"{a} | {b} = {result}")

# Check the Bitwise XOR (^) conditions

result = a ^ b

print(f"{a} ^ {b} = {result}")

# Check the Bitwise NOT (~) conditions

result = ~a

print(f"~{a} = {result}")

# Check the Bitwise Left Shift (<<)conditions

result = a << 2

print(f"{a} << 2 = {result}")

# Check the Bitwise Right Shift (>>)conditions

result = a >> 2 # Result: 3 (In binary: 0011)

print(f"{a} >> 2 = {result}")

**Output** :-

