**Experiment No : 02**

**Title : Write a simple Python program  using Operators**

**Problem Statement :**

Illustrate use of various operators.

**Theory :**

Operators in general are used to perform operations on values and variables in Python. These are standard symbols used for logical and arithmetic operations.

**Arithmetic operators:** Arithmetic operators are used to perform mathematical operations

|  |  |  |
| --- | --- | --- |
| Operator | Description | Syntax |
| + | Addition: adds two operands | a + b |
| - | Subtraction: subtracts two operands | a- b |
| \* | Multiplication: multiplies two operands | x \* y |
| / | Division (float): divides the first operand by the second | m /n |
| // | Division (floor): divides the first operand by the second | c// d |
| % | Modulus: returns the remainder when first operand is divided by the second | x % y |
| \*\* | Power : Returns first raised to power second | x \*\* y |

**Relational Operators:**Relational operators compares the values. It either returns **True** or **False** according to the condition.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Syntax |
| > | Greater than: True if left operand is greater than the right | x > y |
| < | Less than: True if left operand is less than the right | x < y |
| == | Equal to: True if both operands are equal | x == y |
| != | Not equal to - True if operands are not equal | x != y |
| >= | Greater than or equal to: True if left operand is greater than or equal to the right | x >= y |
| <= | Less than or equal to: True if left operand is less than or equal to the right | x <= y |

**Logical operators:**Logical operators perform **Logical AND**, **Logical OR** and**Logical NOT** operations.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Syntax |
| and | Logical AND: True if both the operands are true | x and y |
| or | Logical OR: True if either of the operands is true | x or y |
| not | Logical NOT: True if operand is false | not x |

**Bitwise operators:**Bitwise operators acts on bits and performs bit by bit operation.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Syntax |
| & | Bitwise AND | x & y |
| | | Bitwise OR | x | y |
| ~ | Bitwise NOT | ~x |
| ^ | Bitwise XOR | x ^ y |
| >> | Bitwise right shift | x>> |
| << | Bitwise left shift | x<< |

**Assignment operators:**Assignment operators are used to assign values to the variables.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Syntax |
| = | Assign value of right side of expression to left side operand | x = y + z |
| += | Add AND: Add right side operand with left side operand and then assign to left operand | a+=b     a=a+b |
| -= | Subtract AND: Subtract right operand from left operand and then assign to left operand | a-=b       a=a-b |
| \*= | Multiply AND: Multiply right operand with left operand and then assign to left operand | a\*=b       a=a\*b |
| /= | Divide AND: Divide left operand with right operand and then assign to left operand | a/=b         a=a/b |
| %= | Modulus AND: Takes modulus using left and right operands and assign result to left operand | a%=b   a=a%b |
| //= | Divide(floor) AND: Divide left operand with right operand and then assign the value(floor) to left operand | a//=b       a=a//b |
| \*\*= | Exponent AND: Calculate exponent(raise power) value using operands and assign value to left operand | a\*\*=b     a=a\*\*b |
| &= | Performs Bitwise AND on operands and assign value to left operand | a&=b     a=a&b |
| |= | Performs Bitwise OR on operands and assign value to left operand | a|=b         a=a|b |
| ^= | Performs Bitwise xOR on operands and assign value to left operand | a^=b       a=a^b |
| >>= | Performs Bitwise right shift on operands and assign value to left operand | a>>=b     a=a>>b |
| <<= | Performs Bitwise left shift on operands and assign value to left operand | a <<= b                    a= a << b |

**Special operators:**There are some special type of operators like-

**Identity operators-**  
**is** and **is not** are the identity operators both are used to check if two values are located on the same part of the memory. Two variables that are equal does not imply that they are identical.

**is** True if the operands are identical

**is not** True if the operands are not identical

**Membership operators**-  
**in** and **not in** are the membership operators; used to test whether a value or variable is in a sequence.

**in** True if value is found in the sequence

**not in** True if value is not found in the sequence

**code :**

a=10

b=56

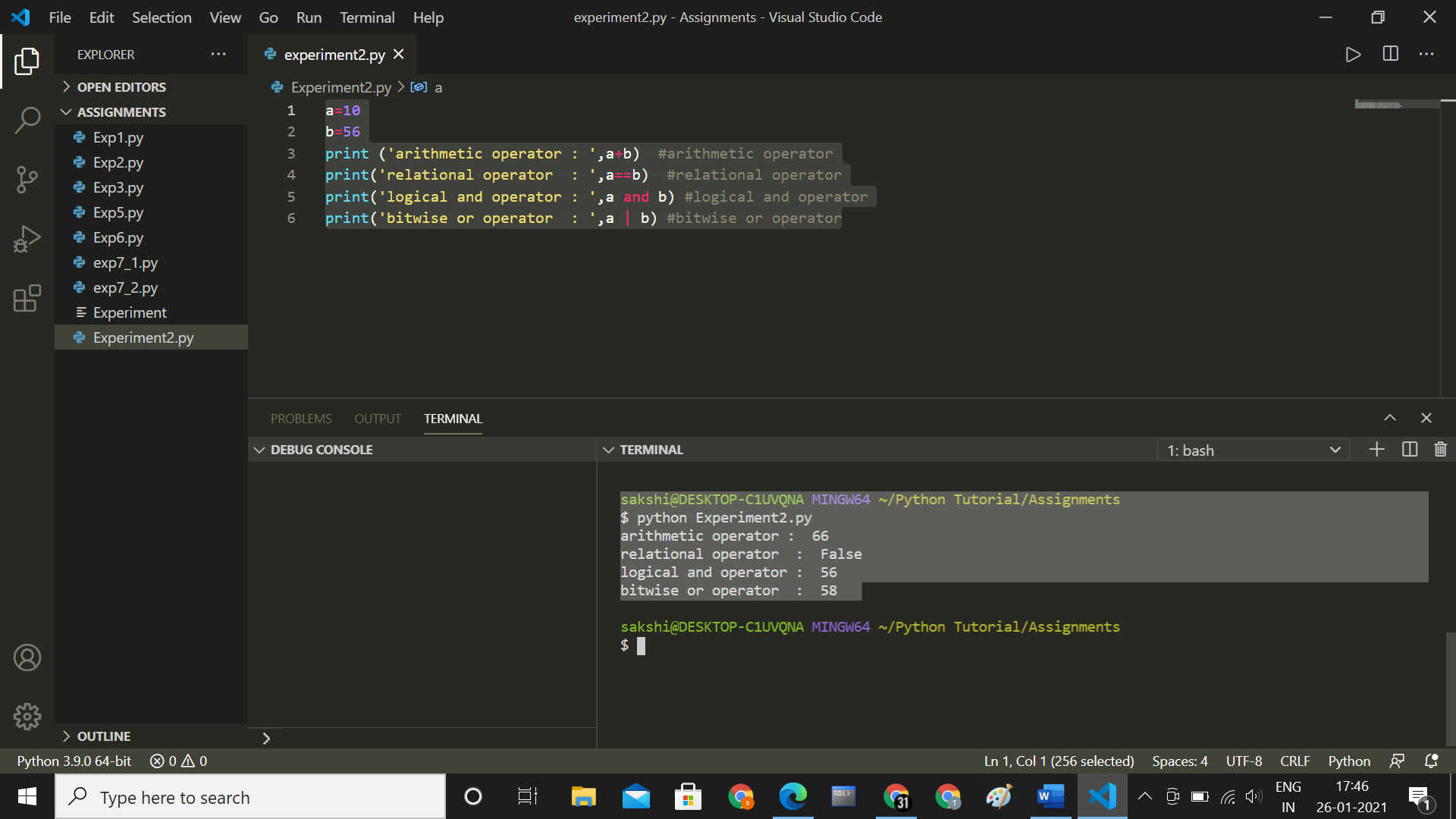
print ('arithmetic operator : ',a+b)  #arithmetic operator

print('relational operator  : ',a==b)  #relational operator

print('logical and operator : ',a and b) #logical and operator

print('bitwise or operator  : ',a | b) #bitwise or operator

**output :**



**Conclusion :**

**Thus we have understood how to write program using operators .**