**2.13 Python Operators**

Operators are tokens that trigger some computation when applied to variables or elements of expressions. Python supports many operators to do computational works. Operators are divided into two.

1. Binary operator: They are the operators that require two operands on the left and right of the operator. Eg: +, -, x etc.
2. Unary operator: They are the operators that require only one operand to operate upon. Eg: +, -

The following list gives a brief description of commonly using operators and their functions.

| **Exponentiation** | | | |
| --- | --- | --- | --- |
| Symbol \*\* | Example 3\*\*2 | Result  8 | To find 32 |
| **Multiplication** | | | |
| Symbol \* | Example 3\*2 | Result 6 | Ordinary multiplication. If any one of the operands is a float, the result will be float. If both are integer, the result will be an integer |
| **Division** | | | |
| Symbol  / | Example 4.8/2 | Result 2.4 | Ordinary division. If any one of the operands is a float, the result will be float. If both are integer, the result will be an integer by truncation. |
| **Floor division** | | | |
| Symbol // | Example 4.4//2  7//2 | Result 2.0  3 | A special type of division to get the quotient. If any one of the operands is a float, the result will be float. If both are integer, the result will be an integer. In both cases, the result will be a truncated one to the lowest whole number. |

**Arithmetic operators**

| **Modulus division** | | | |
| --- | --- | --- | --- |
| Symbol  % | Example 4.4%2  7%2 | Result  0.4  1 | A special type of division to get the reminder. If any one of the operands is a float, the result will be float. If both are integer, the result will be an integer. In both cases, the result will be the reminder after getting a quotient of the whole number |
| **Addition** | | | |
| Symbol + | Example 10+8 | Result 18 | Ordinary addition. If any one of the operands is a float, the result will be float. If both are integer, the result will be an integer |
| **Subtraction** | | | |
| Symbol - | Example 10-8 | Result 2 | Ordinary subtraction. If any one of the operands is a float, the result will be float. If both are integer, the result will be an integer |
| **Negation** | | | |
| Symbol - | Example -x | Result -x | The value stored in x reverses the sign. It is a unary operator. |

**Relational operators**

These operators are binary operators, to check wheatear a mathematical condition is true or false. This is to make a decision in ‘while loop’, ‘if loop’ etc. Always the result will be Boolean.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of operator** | **symbol** | **Example** | **Result** |
| Less than | < | 2<3 | .T. |
| Less than or equal | <= | 3<=2 | .F. |
| Greater than | > | 3>2 | .T. |
| Greater than or equal | >= | 5>= 5 | .T. |
| Not equal | != | 6 !=6 | .F. |
| Equal | = = | 12==12 | .T. |

We must give extra care on using conditional equal to during a conditional check. If the variables involved are of integer type, exactness is possible. If it is a float, a perfect exactness is impossible in most case. It is due to the difference in the accuracy level of the microprocessor.

**Logical operators**

These operators are binary operators, to check wheatear more than one mathematical condition is true or false. This is to make a decision in ‘while loop’, ‘if loop’ etc with multiple conditions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of operator** | **Symbol** | **Example** | **Result** |
| Logical OR | or | x=2 or y=4 | True if x=2 or y=4 |
| Logical AND | and | x=2 and y=4 | True only if x=2 or y=4 |
| Logical NOT | not | not y>9 | It is a unary operator. True only if y>9 is false. That means it is true only if y<9 |

**Assignment operator**

As discussed before, the assignment operator connects the data with a variable name.

|  |  |  |  |
| --- | --- | --- | --- |
| *Name of operator* | *Symbol* | *Example* | *Result* |
| Assignment equal to | = | x=2 | Value of x in memory location becomes 2 |

Remember that assignment equal to (=) and relational equal to (= =) are with a different meaning.

**Precedence of operators**

On solving a mathematical expression, the actions of operators are based on some rules of precedence. The computer will scan the expression from left to right. It will execute the first precedence operators in the order of occurrence. Then again scan from left to right for second precedence operators and will execute the second precedence operators in the order of occurrence. This will continue up the final answer. At any instant precedence rule can be violated with the proper use of parenthesis. So special care is to be given in framing a mathematical expression, which is free from semantic error. The following table lists the precedence order from high to law.

1. Logical NOT, Negation
2. Exponentiation
3. Multiplication, Division, Modulus division, Floor division
4. Addition, Subtraction
5. Less than, Less than or equal, Greater than, Greater than or equal
6. Equal, Not equal
7. Logical AND
8. Logical OR
9. Simple assignment