

Module - 4 (Basics of Python)

① $\backslash n \rightarrow$

new

② Variables \rightarrow

$\langle \text{int} \rangle \rightarrow 2, 3, 4, \dots$

$\langle \text{float} \rangle \rightarrow 2.0, 2.1, 2.2, \dots$

$\langle \text{str} \rangle \rightarrow "Hi", 'Hi'$

boolean $\rightarrow \text{True}, \text{False}, \text{None}, \dots$

③ del \rightarrow

you can't use number for a variable name

$1 = "Sunny" \rightarrow \text{wrong}$

$a1 = "sunny" \rightarrow \text{Right}$

$A = "Sunny" \rightarrow \text{Right}$

$a = "Sunny" \rightarrow \text{Right}$

$1a = "Sunny" \rightarrow \text{wrong}$

$-a = "sunny" \rightarrow \text{Right}$

$-1 = "Bunny" \rightarrow \text{Right}$

④ Keywords

keyword.kwlist

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def',  
'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

35, 58, 7.9

⑤ Length of a string

0 1 2 3 4 5 → +ve index
-3 -2 -1 -2 -1 → -ve index

total index:
index = (length - 1)
starts from 0

0 1 2 3 4 5 6 7 8 9 10

⑥ adding two strings

str + float X convert float to str then concatenate
str + int X

① Type Conversion (int, str, float)

→ Type conversion

★ $a = 56.45$ $b = 56.75$
 $\text{type}(a) = \text{float}$ $\text{type}(b) = \text{float}$
 $\text{int}(a) = 56$ $\text{int}(b) = 56$ doesn't get rounded off

we can convert 'int' to 'str' if there are integers in
' ' but if there are alphabets then it will show an
error.

$a = \text{'Hello'}$

$\text{int}(a) = \text{error}$

$c = \text{'55.43'}$

$\text{int}(c) = \text{error}$

$b = \text{'12345'}$

$\text{int}(b) = 12345$

$d = \text{'55.62'}$

$\text{float}(d) = 55.62$

⑧ Type conversion (Boolean)



→ False
only for
these

Q Arithmetic Operators

$$\rightarrow 10/3 = 3.3333\bar{3}$$

$10//3 = 3 \rightarrow$ only gives quotient.

$$\rightarrow 10\%3 = 1 \rightarrow \text{only gives remainder}$$

$$\rightarrow 2^{**}5 = 32$$

Precedence

$() > ** > (*, /, //, \%) > (+, -)$

10) Assignment operators

$$\rightarrow n = n + 1$$

$$n = n - 1$$

$$n = n * 10$$

$$\rightarrow n = n / 10$$

$$\rightarrow n = n \ll 10$$

$$\rightarrow n = n \gg 10$$

$$\rightarrow n = n \% 10$$

⑪ Comparison operators ($=$, $!=$, $>$, $<$, $>=$, $<=$)

↓ ↓ ↓
equal not greater
to equal or equal
to to

⑫ Logical Operators

Precedence
and > or

② Precedence and Associativity.

Operator Precedence and Associativity List in Python

Consider following list of operator precedence and associativity in Python. It shows all operators from highest precedence to lowest precedence.

1. **()** : Parentheses (highest precedence) -> **Associativity**: Left to right
2. **x[index], x[index:index]** : Subscription, slicing -> **Associativity**: Left to right
3. **await x** : Await expression
4. ****** : Exponentiation -> **Associativity**: Right to left
5. **+x, -x, ~x** : Unary plus, unary minus, bitwise NOT -> **Associativity**: Right to left
6. ***, @, /, //, %** : Multiplication, matrix multiplication, division, floor division, remainder -> **Associativity**: Left to right
7. **+, -** : Addition and subtraction -> **Associativity**: Left to right
8. **<<, >>** : Bitwise shifts -> **Associativity**: Left to right
9. **&** : Bitwise AND -> **Associativity**: Left to right
10. **^** : Bitwise XOR -> **Associativity**: Left to right
11. **|** : Bitwise OR -> **Associativity**: Left to right
12. **in, not in, is, is not, <, <=, >, >=, !=, ==** : Comparisons, membership, identity tests -> **Associativity**: Left to right
13. **not x** : Boolean NOT -> **Associativity**: Right to left
14. **and** : Boolean AND -> **Associativity**: Left to right
15. **or** : Boolean OR -> **Associativity**: Left to right
16. **if-else** : Conditional expression -> **Associativity**: Right to left
17. **lambda** : Lambda expression
18. **:=** : Assignment expression (Walrus operator) -> **Associativity**: Right to left

(14) Print()

↳ separator

(15) input()

↳ what ever input you receive is (str).

(16) Numeric function

Module 5

(17) Slicing in String

0 1 2 3 4 5 6 7 8 9 10

↳ 2, 3, 4, 5, 6

[No space]

↳ 5th index is space.

[No space]

2, 4, 6, 8

↳ There are only 11 indexes but still it gives result