Python class 4

⇒ Strings → 1. Concatenation
2. Slicing

⇒ Operators -

String Concatenation

Concat ⇒ joining / combining

String concat. ⇒ joining your strings

1. Joining Name ⇒ First + Last
2. Address ⇒ House no. + area + Street
3. URL ⇒ host port /____

dB connection

www. "google" .com

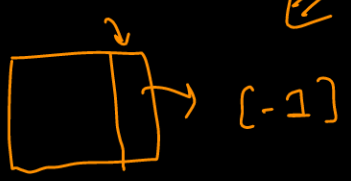
4.

Slicing of Strings

seq
- str
- list
- range
- tupple

str

P	Y	T	H	O	N
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1



delhi · 33

[ ] ✓

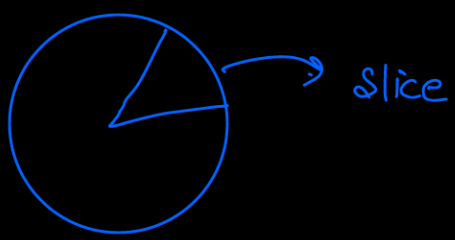
Slicing

str[-5] =

-5 + len = 1
(6)

str[1]

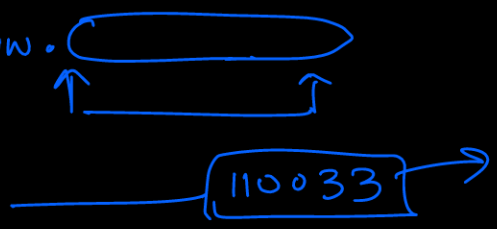
str[8] = error



When we want a part/piece of your str/list

25/Feb/98

491



How to do slicing in python

[ ] → indexing

[x : y]

x ⇒ start index

y ⇒ end ~~index~~ → (y-1)

PYTHON
0 1 2 3 4 5

str[0:3] = PYT

str[1:3] = YT

Even if start or end index are out of bounds . it returns empty string.

It goes till end index - 1 .

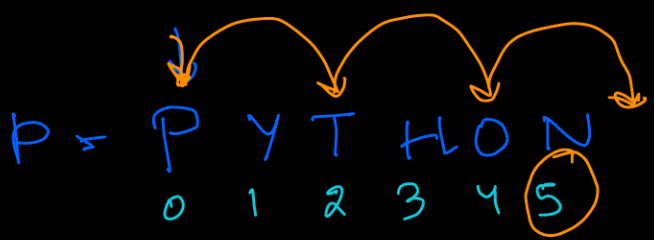
Step Value

[x : y : s]

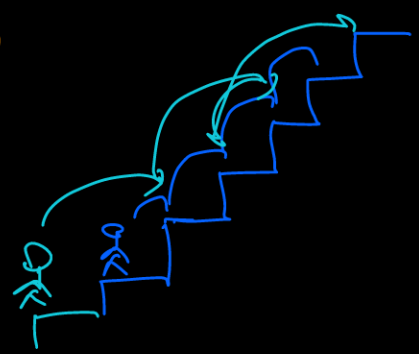
x → begin
y → end
s ⇓ step value

default = 1

how many characters to move forward/backward after the first char. is retrieved



1 by default



p[0:6:2]
[value color]

P T O

p[1:8:2]

Sign of your step value defines where to go. ⟹ direction

5 3 3 4 8 9 8 7

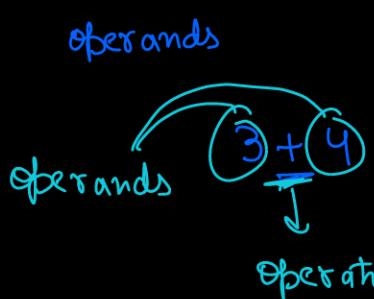
0 0 0 0

OPERATORS

What are operators ⇒ special symbols which do computations on values

Types :-

a) Arithematic
b) Comparision / Relational
c) Logical
d) Assignment
e) Identity
f) Membership



Arithematic

+	Addition
-	Subtraction
*	multiplication
/	division

Our operator behaviours changes based on operands

"PYTH" + "HON" ⇒ 3 + 4 = 7

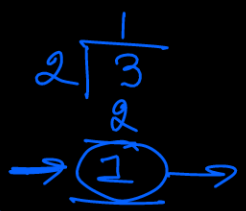
% Modulus

** Exponentiation

// Floor division

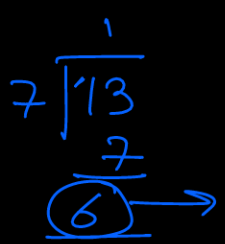
% ⇒ modulus ≡ remainder

3 % 2

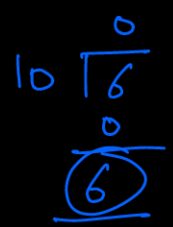


13 % 7

6



6 % 10



// ⇒ floor division

not rounding off

⇒ integral part / Quotient

29//10 = 2 not 3 not 2.99

Relational operators

⇓

get relation b/w

2 things getting compared

They give either T or F as output

>	greater than
<	Less than
==	Equal To
>=	Greater than or equal to
<=	Less than or equal to
!=	Not equal to

a > b

Can be used to compare diff. operands

no. string list

6 > 8 ⇒

Relational operators in strings

"Unicode" is compared

Lexicographical Comparision

a → 97
b → 98
c → 99

abc acb

no addition
no multiplication

a 97 = 97 a
b 98 < 99 c ⇒ greater
c b

a 97 97 a ==
b 98 98 b ==
c 99 0 99 > 0
⇒ abc > ab

Q = Add operator for T ans

① xyz ☐ xya ③ xyzzz ☐ z
② XYa ☐ XYA

Chaining of relational operators

⇒ Chaining allowed

$1 < 2 < 3$

Python evaluates each & every expression individually & returns T if all True else False

input = ( )

$10 < input < 20$

$7 > 6 > 5$

$7 > 6$ $\quad\quad$ $6 > 5$

$5 < 6 > 7$

$5 < 6 \Rightarrow T$ $\quad\quad$ $6 > 7 \Rightarrow F$

$\Rightarrow$ False

Special Behaviour of == & !=

Both type & value are compared

1 == '1' False str & int

97 == 'a' False

!= ⇒ true if types are diff.

Logical Operators

and
or
not

to combine 2 or more equations

a > b > c ⇒ a > b and b > c

		and	or
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F
		(*)	(+)

T → 1
F → 0

with numericals & Booleans ⇒ easy

a) None, 0, "", 0.0 ⇒ False

b) return value of log ^ and log $ ^ or is not T or F when applied to non boolean Types

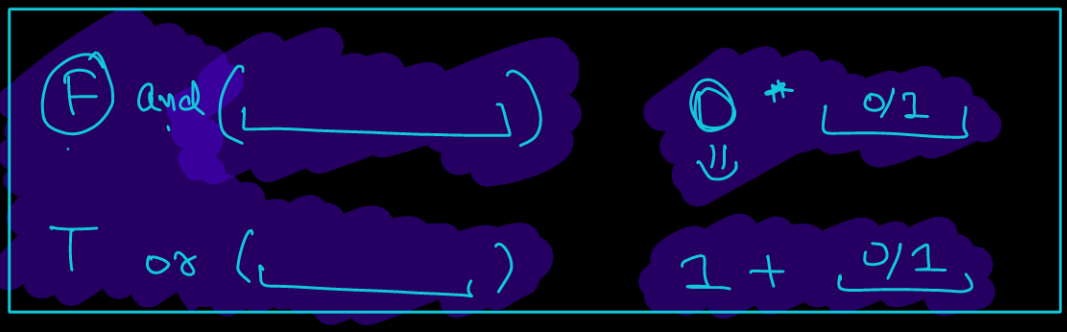
String
integer
list

c) If first value is False, then logical and returns first value else it returns 2nd value

'sachin' and 10

d) If first value is True,
then logical or returns first val
else returns 2nd value

- 0 and "Hello"
- 0 or "hello"



e) not operator on non-boolean types
False if its True
True if false

not 0