

Comparison Operators

These operators compare variables & o/p Boolean values (True or False)

For e.g. $>$, $<$, $<=$, $=$, $<=>$

Chained Comparison Operators

Python allows us to chain multiple operators to perform more complex operations.

Logical operators

1) and 2) or 3) not

Qn: $1 > 2$

Qn: $1 < 2 < 3$

O/p: True

Qn: $1 < 2$ and $2 < 3$

O/p: True

Qn: not $(1 == 2)$

O/p: True

Qn: $1 == 1$ or $100 == 1$

O/p: True

If, elif & else

Syntax:

Indentation very imp.
colon very imp.

```
if case 1:  
    perform action 1
```

elif case 2:

perform action 2

~~else case 3:~~

~~perform action 3~~

else:

perform action 3

Indentation:

If maintains structure & order of code

For loop

A for loop acts as an iterator in Python. It iterates thro. Iterable items like strings, lists, tuples, & dictionaries.

Syntax:

for item in object:
└── statements

for e.g. ①

l1 = [1, 2, 3, 4]

for num in l1:

└── if num % 2 == 0:
 └── print(num)

else:

 print("Odd")

O/p:

Odd
2

Odd
4

We can

② for let in 'This is':
 print(let)

O/p:

T

h

i

s

s

When Iterating thro. tuples sequence that contains tuples, the Item can actually be the tuple itself, this is an example of tuple unpacking.

For e.g.

V/p:

`l2 = [(2,4), (6,8), (10,12)]`

`for tup in l2:`

`print(tup)`

O/p:

`(2,4)`

`(6,8)`

`(10,12)`

UNPACKING

V/p:

`for (t1,t2) in l2:`

`print(t1)`

O/p:

`2`

`6`

`10`

e.g. with Dictionaries:

gn: `d = {'k1': 1, 'k2': 2, 'k3': 3}`

for item in d:
 ~~print(item)~~

o/p: k1
 k2
 k3

DICTIONARY UNPACKING

gn: for k, v in d.items():
 print(k)
 print(v)

o/p k1
 1
 k2
 2
 k3
 3

SORTED() → To sort a sequence

gn: `sorted(d.values())`

o/p: [1, 2, 3]

WHILE LOOP:

Syntax:

```
while test:  
    | code statements
```

```
else:  
    | final code statements
```

BREAK, CONTINUE, PASS

break = breaks out of current closest enclosing loop

continue: skips the current iteration & moves to the next iteration

pass: does nothing at all

For e-g.

```
while True:
```

~~pass~~

In: while True:

```
    | pass
```

o/p: [No Error]

If you by mistakenly run a loop restart
the kernel