06/11/23
Mon

Relational Operators:-

- Relation operators are used for comparing values. An expression created using relational operators is called Boolean expression.

⇒ Boolean expression always returns Boolean values (T/F)

Operators	Description
>	greater than
<	less than
>=	greater than or equal.
<=	less than or equal
==	Equal
!=	Not equal.

⇒ These operators are used to comparing Object values or states.

Eg1>>> 10>5
True.

Eg2>>> 40>30
True.

Eg3 10<5
False.

Eg4 12>1
True.

Eg5>>> 10>=10
True

Eg6 10>=5
True.

Eg7>>> 10<=10
True.

Eg8 5<=10
True

Eg9 20<=10
False.

Eg10>>> 10==10
True

Eg11:- 10!=10
False.

Eg:12 10!=20
True.

Eg:- "A" > "B"
⇒ (65) > (66) — ASCII value.
→ False

Eg:- "B" > "A"
→ True.

Eg:- "a" > "A"
→ True.

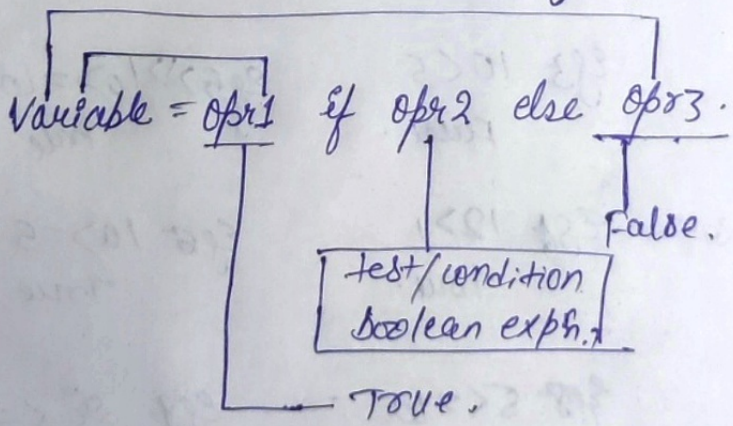
* MORE THAN ONE OPERATOR WE CAN USE:-
Eg:- a = 5
⇒ 1 <= a <= 10
True.

Eg:- a = 20
⇒ 1 <= a <= 10
False.

Eg:- 5 >= 1 <= 10
True.

Conditional operators:-
→ Conditional operator is a ternary operator, this operator required 3 operands. conditional operator is used for creating conditional expression.
→ Conditional operators allows evaluating expression based on conditional test.

Syntax:- <variable-name> = opr1 if opr2 else opr3



W.A.P to find a person is eligible to vote or not.
Soln:- name = input ("Enter Name")
age = int (input ("Enter the age"))

result = f'{name} is eligible' if age>=18 else f'{name} is not eligible to vote'.
print(result).

Eg2:- # W.A.P to find input no is even or odd.

Soln:- num = int(input("enter any value"))
rem = num%2 (# remender should be zero).
a = f'{num} is even' if rem==0 else f'{num} is odd'
print(a)

O/P 20 - even
15 - odd.

Eg:- W.A.P to find max. of two numbers.

Soln: num1 = int(input("enter first no"))
num2 = int(input("enter 2nd no"))
num3 = num1 if num>num2.
or, result = f'{num1} is max' if num1>num2 else f'{num2} is max'.
print(result).

Logical Operators:-

Logical operators are used to combine two or more conditions or boolean expression. In Python logical operators are represented using 3 keywords.

1.) and → Binary operator
2.) or → Binary "
3.) not → unary ".

1) and operator: Both condition / All conditional should be satisfied or True.

Truth table of and operator:

opr1	opr2	opr1 & opr2.
True	False	False
False	True	False
False	False	false
True.	True	true.

Eg:)) True and True
True.

Eg:) True and False
False.

Eg:)) False and True
False

Eg:- 10>5 and 5<10
True.

Eg:- 20>10 and 5>10
False.

Eg:- 10>20 and 5<10
false.

W.A.P to read name and 2 subject marks:
calculate total, avg and result.

Sol^n:
```
name = input("Enter Name")
sub1 = int(input("enter sub1 marks"))
sub2 = int(input("enter sub2 marks"))
total = sub1 + sub2
avg = total/2.
result = "pass" if sub1>=40 and sub>=40 else "fail".
print(f"Name {name}
subject1 {sub1} subject2 {sub2}
```

Total {total}

Examples: Avg {avg:.2f}
Result {result}")

Examples:- >>> 100 and 200
200

Eg:- 0 and 200
0

Eg:- 100 and 200 and 300
300

Eg:- 100 and 0 and 300
0

Eg:- "Java" and "python"
'python'

Eg:- "A" and "B"
'B'

Eg:- bool(100)
True.

Eg:- bool(0)
false.

W.A.P to
Login Appln.

Soln:-
```
user = input("enter user name:") # nit
Pswd = input("enter password:") # nit123
print("welcome") if user == "nit" and pswd == "nit123"
        else print("invalid username or password")
```

O/P enter user name: Nenitam
enter password: nit123
invalid username or password.

(or)

O/P enter user name: nit
enter password: nit123
Welcome.

2> Or operator/keyword:- If any operand is True, the complete expression return True.

Truth Table:

opr1	opr2	opr1 or opr2
True	True	True
False	True	True.
False	False	False
True	False	True.

→ If opr1 is "True", PVM return result of opr1 without evaluating opr2.

→ If opr1 is "False", PVM evaluates opr2 & return result of opr2.

Eg:- True or False
True.

Eg:- False or False
False.

Eg:- False or False
, True.

Eg:- 100 or 200.
100.

Eg:- 100 or 200 or 300
100.

Precedence of logical operator:-

1>not
2>and
3> or.

Examples:-

1>> 100 or 200
100

Eg:- 100 or 200 or 300
100

Eg:- 100 or 200 and 300
100

Eg:- 300 and 300 or 100
300

Eg:- 0 and 100 or 200
200.