2. ASSIGMENT OPERATOR

In [ ]:
```
=>The purpose of Assigment operator is that "To assign or transfer right hand side
  to the LHS side variable
=>The symbol is the Assigment operator is that single equal (=)
=>In python programming, we can use Assigment operator in two ways.
1.single Line Assigment operator
2.Multi Line Assigment operator
```

In [4]:
```
a=20
b=30
print(a,b)
```

```
20 30
```

In [6]:
```
a,b=b,a
print(a,b)
```

```
30 20
```

3. RELATIONAL OPERATOR

In [ ]:
```
                                        3. RELATIONAL OPERATOR
=>The purpose of Relational operator is that "To compare Two are more values"
=>Two or more values connected with Relational Operators then it is called Relation
=>The reult of exprission is either True or False (bool values)
=>The Relation exprission is also called simple test condition whose result can be
=>In Python Programming 6 type of Relation Operators. They are:
1.greater than >
2.Less than <
3.double equal to ==
4.Not eual to !=
5.greater than equal >=
6.Less than equal <=
```

In [14]:
```
#1.greater than >
print(10>2)
print(10>20)
```

```
True
False
```

In [16]:
```
#2.Less than <
print(10>20)
print(20>15)
```

```
False
True
```

In [18]:
```
#3.double equal to == (Equality operator)
print(10==10)
print(10==20)
```

```
True
False
```

In [10]:
```
#4.Not eual to !=
print(10!=10)
```

```
False
```

In [20]:
```
#5.greater than equal >=
print(10>=2)
print(10>=20)
```

```
True
False
```

In [22]:
```
#6.Less than equal <=
print(10<=20)
print(10<=5)
```

```
True
False
```

In [32]:
```
ord("A")
```

Out[32]: 65

In [34]:
```
ord("Z")
```

Out[34]: 90

In [50]:
```
for val in range(65,91):
    print(val)
```

```
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
```

In [2]:
```
chr(65)
```

Out[2]: 'A'

*** Disply all uper case alphabets for unicode values(65-A----90-Z) ***

In [4]:
```
#Disply all uper case alphabets for unicode values(65-A----90-Z)
for val in range(65,91):
    print("\t{}--->{}".format(val,chr(val))

         )
```

```
65--->A
66--->B
67--->C
68--->D
69--->E
70--->F
71--->G
72--->H
73--->I
74--->J
75--->K
76--->L
77--->M
78--->N
79--->O
80--->P
81--->Q
82--->R
83--->S
84--->T
85--->U
86--->V
87--->W
88--->X
89--->Y
90--->Z
```

In [12]:
```
"ABC">"ACB"
```

Out[12]: False

In [4]:
```
"ABB">="AA"
```

Out[4]: True

In [6]:
```
"ABC">="ACB"
```

Out[6]: False

In [8]:
```
"MAHABOOB">="KHAN"
```

Out[8]: True

In [10]:
```
"MRIIRS">="CDOE"
```

Out[10]: True

*** Disply all Lowercase alphabets for unicode values(97-a----122-z) ***

In [16]:
```
for val in range(97,123):
    print("\t{}--->{}".format(val,chr(val))

        )
```

```
97--->a
98--->b
99--->c
100--->d
101--->e
102--->f
103--->g
104--->h
105--->i
106--->j
107--->k
108--->l
109--->m
110--->n
111--->o
112--->p
113--->q
114--->r
115--->s
116--->t
117--->u
118--->v
119--->w
120--->x
121--->y
122--->z
```

In [18]:
```
"python">"PYTHON"
```

Out[18]: True

In [20]:
```
"PYTHON">"python"
```

Out[20]: False

In [22]:
```
"MRIIRS"<"cdoe"
```

Out[22]: True

In [24]:
```
"MEHBOOB">"khan"
```

Out[24]: False

In [2]:
```
"aBC">="abc"
```

Out[2]: False

In [4]:
```
"wrong">="wrnog"
```

Out[4]: True

In [6]:
```
"this">="thsi"
```

Out[6]: False

In [8]:
```
"cat">="cta"
```

Out[8]: False

In [42]:
```
#Program demonstrating the functionality of relational operators?
a,b=float(input("Enter First value:")),float(input("Enter second value:"))
print("*"*50)
print("Result of Realation operator")
print("*"*50)
print("\t\t {}>{}={}".format(a,b,a>b))
print("\t\t {}<{}={}".format(a,b,a<b))
print("\t\t {}=={}={}".format(a,b,a==b))
print("\t\t {}!={}={}".format(a,b,a!=b))
print("\t\t {}>={}={}".format(a,b,a>=b))
print("\t\t {}<={}={}".format(a,b,a<=b))
print("*"*50)
#NOTE:a>b, a<b, a==b, a!=b, a>=b, a<=b are called relational expressions.
```

```
**************************************************
Result of Realation operator
**************************************************
                20.0>10.0=True
                20.0<10.0=False
                20.0==10.0=False
                20.0!=10.0=True
                20.0>=10.0=True
                20.0<=10.0=False
**************************************************
```

4.LOGICAL OPERATORS (COMPARISION OPERATORS)

In [ ]:
```
=>The purpose of use logical operators is that "to combine two are more Relational
=>If two or more Relational Expressions combined two Logical Operators then it is c
=>The result of Logical Expression is either True or False
=>The Logical Expression is also compound test condition and whose result can be ei
=>In Python programming we have 3 types of Logical operators:
1.and
2.or
3.not
```

In [ ]:
```
1.and operator:
syntax: relation Expression1 and relational Expression2
=>The functionalty of and operator is shown the following Truth Tables:
```

In [51]:
```
True and False
```

Out[51]: False

In [53]:
```
False and True
```

Out[53]: False

In [55]:
```
True and True
```

Out[55]: True

In [57]:
```
False and False
```

Out[57]: False

In [65]:
```
10>2 and 20>4
```

Out[65]: True

In [67]:
```
10>5 and 20>10 and 50>10
```

Out[67]: True

In [69]:
```
10>20 and 40>20 and 10>5
```

Out[69]: False

In [71]:
```
10>5 and 40>100
```

Out[71]: False

In [73]:
```
10>5 and 20>40 and 30>20
```

Out[73]: False

In [75]:
```
100 and 200 #second True is Answer
```

Out[75]: 200

In [77]:
```
-100 and -200
```

Out[77]: -200

In [79]:
```
0 and 30
```

Out[79]: 0

In [81]:
```
100 and 0
```

Out[81]: 0

In [83]:
```
100 and 200 and 300
```

Out[83]: 300

In [85]:
```
100 and 0 and 400
```

Out[85]: 0

In [87]:
```
"False" and "True" #Second Non-zero is Answer
```

Out[87]: 'True'

In [91]:
```
"True" and "False" #Second Non-zero is Answer
```

Out[91]: 'False'

In [93]:
```
"Java" and "Python"
```

Out[93]: 'Python'

In [101…
```
0b1010 and 0xF
```

Out[101… 15

In [111…
```
100 and ""
```

Out[111… ''

In [113…
```
"  " and 100
```

Out[113… 100

In [115…
```
len("  ")
```

Out[115… 2

In [117…
```
bool("False")
```

Out[117… True

In [119…
```
bool(False)
```

Out[119… False

In [121…
```
int(False)
```

Out[121… 0

In [123…
```
"True" and bool("False")
```

Out[123… True