1.What are the two values of the Boolean data type? How do you write them?

In [1]:
```
# boolean data type only return two type value either True or False or we say 1 for true and 0 for false
a=0
b=7
c=b<a
print(c)
print(type(c))
d=bool(a) # type conversion "integer into bool"
print(d, type(d))
```

```
False
<class 'bool'>
False <class 'bool'>
```

2. What are the three different types of Boolean operators?

In [2]:
```
# 'and ' ,'or','not' are the three boolean operator
a=5
b=7
if a>0 and b<0:    # each condition is true then it is true,anyone of them is false the it become false
    print('points lie in second quadrant ')
else:
    print('not lie ')
```

```
not lie
```

In [3]:
```
a=5
b=7
if a>b or b<10:    # if any one condition is true then it become true in "or " operator
    print('true')
else:
    print('false')
```

```
true
```

In [4]:
```
a=5
b=7
if not(a<b):               # 'not' operator change the result i.e true into false and false into true
    print('true')
else:
    print('false')
```

```
false
```

3. Make a list of each Boolean operator's truth tables (i.e. every possible combination of Boolean values for the operator and what it evaluate).

In [5]:
```
''' truth table of 'and' operator NOTED: '0' for false and '1' for true
x   y   x and y
0   0     0
0   1     0
1   0     0
1   1     1

    truth table of 'or'
x   y    x or y
0   0      0
1   0      1
0   1      1
1   1      1

      truth table of 'not'
x    not x
1     0
0     1
'''
```

Out[5]:
```
"  truth table of 'and' operator NOTED: '0' for false and '1' for true\nx   y   x and y\n0   0     0\n0   1     0\n1
0     0\n1   1     1\n\n     truth table of 'or'\nx   y    x or y\n0   0      0\n1   0      1\n0   1      1\n1   1
1\n      \n      truth table of 'not'\nx    not x\n1     0\n0     1      \n"
```

4. What are the values of the following expressions?

(5 > 4) and (3 == 5)

not (5 > 4)

(5 > 4) or (3 == 5)

not ((5 > 4) or (3 == 5))

(True and True) and (True == False)

(not False) or (not True)

In [6]:
```
a=(5 > 4) and (3 == 5)
print(a)
b=not(5>4)
print(b)
c=(5 > 4) or (3 == 5)
print(c)
d=not ((5 > 4) or (3 == 5))
print(d)
e=(True and True) and (True == False)
print(e)
f=(not False) or (not True)
print(f)
```

```
False
False
True
False
False
True
```

5. What are the six comparison operators?

In [7]:
```
'''
1. Less than (<)

2. Greater than (>)

3. Less than or equal to (<=)

4. Greater than or equal to (>=)

5. Equal to (==)

6. Not equal to (!=)

'''
```

Out[7]:
```
'\n1. Less than (<)\n\n2. Greater than (>)\n\n3. Less than or equal to (<=)\n\n4. Greater than or equal to (>=)\n\n5. E
qual to (==)\n\n6. Not equal to (!=)\n\n'
```

6. How do you tell the difference between the equal to and assignment operators?Describe a condition and when you would use one.

In [8]:
```
# '==' use for compare two value
#  '=' use for assigning the value
a=3+2        # sum of 3 and 2 is assigned to variable 'a
print('value of a is',a)
if a==5:   # compare the value 5 to the variable
    print('true')
```

```
value of a is 5
true
```

7. Identify the three blocks in this code:

spam = 0

if spam == 10:

print('eggs')

if spam > 5:

print('bacon')

else:

print('ham')

print('spam')

print('spam')

In [9]:
```
spam = 0
if spam == 10:
print('eggs') # first indented block
if spam > 5:
print('bacon')  # second indented block
else:
print('ham')     # third indented block
print('spam')
print('spam')
```

```
  Input In [9]
    print('eggs') # first indented block
    ^
IndentationError: expected an indented block
```

In [10]:
```
spam = 0
if spam == 10:
    print('eggs')
if spam > 5:
    print('bacon')
else
    print('ham')
print('spam')
print('spam')
```

```
ham
spam
spam
```

8. Write code that prints Hello if 1 is stored in spam, prints Howdy if 2 is stored in spam, and prints Greetings! if anything else is stored in spam.

In [11]:
```
spam=int(input('enter the integer '))
if spam == 1:
    print('Hello')
elif spam ==2:
    print('Howdy')
else:
    print('Greetings!')
```

```
enter the integer 2
Howdy
```

9.If your programme is stuck in an endless loop, what keys you'll press?

''' In general, typing Control+C cannot be counted on to interrupt a running Python program.

Depending on what is happening in your loop:

1) Canopy's Run menu > Interrupt kernel (for most simple programs, this will work)

or

2) Run menu > Restart kernel

or

3) Quit Canopy, then restart it '''

10. How can you tell the difference between break and continue?

In [12]:
```
# 'break ' statement is use to break the loop and come out
for i in range(10):
    sum=0
    sum = sum +i
    if sum == 8:
        print('time to break')
        break
print(sum)
```

```
time to break
8
```

In [13]:
```
# 'continue' statement that force to execute next iteration and skip  for current iteration
for i in 'hello world':
    if i=='w': # when 'i' will be going to 'w' then loop is continue and below print statement skip for current iterati
        continue
    print(i)
```

```
h
e
l
l
o

o
r
l
d
```

11. In a for loop, what is the difference between range(10), range(0, 10), and range(0, 10, 1)?

In [15]:
```
for i in range(10):
    print(i)    # 12. Write a short program that prints the numbers 1 to 10 using a for loop. Then write an equivalent
print('\n')
for j in range(0,10):
    print(j)             # here starting  given i.e 0 and ending  is 10
print("\n")
for k in range(0,10,1):
       print(k)           # here a skipping given i.e 1   with starting 0 and ending 10
```

```
0
1
2
3
4
5
6
7
8
9


0
1
2
3
4
5
6
7
8
9


0
1
2
3
4
5
6
7
8
9
```

12. Write a short program that prints the numbers 1 to 10 using a for loop. Then write an equivalent program that prints the numbers 1 to 10 using a while loop.

In [16]:
```
for i in range(1,11):
    print(i)
print('\r')
a=1
while a<11:
    print(a)
    a=a+1
```

```
1
2
3
4
5
6
7
8
9
10

1
2
3
4
5
6
7
8
9
10
```

13. If you had a function named bacon() inside a module named spam, how would you call it after importing spam?

In [17]:
```
# understand with example
import math
a=math.pow(2,3)  # math is module and pow() is funcion
print(a)
    # so we call bacon() function by 'spam.bacon()'
```

```
8.0
```