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

* true and False
* a = true
* b= Flase

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

* AND
* OR
* NOT

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 ).

1. **AND (&&)**:
   * True AND True evaluates to True.
   * True AND False evaluates to False.
   * False AND True evaluates to False.
   * False AND False evaluates to False.
2. **OR (||)**:
   * True OR True evaluates to True.
   * True OR False evaluates to True.
   * False OR True evaluates to True.
   * False OR False evaluates to False.
3. **NOT (!)**:
   * NOT True evaluates to False.
   * NOT False evaluates to True.

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)

1. **Expression 1**: (5 > 4) and (3 == 5)
   * 5 > 4 is True.
   * 3 == 5 is False.
   * So, True and False evaluates to False.
2. **Expression 2**: not (5 > 4)
   * 5 > 4 is True.
   * not True evaluates to False.
3. **Expression 3**: (5 > 4) or (3 == 5)
   * 5 > 4 is True.
   * 3 == 5 is False.
   * So, True or False evaluates to True.
4. **Expression 4**: not ((5 > 4) or (3 == 5))
   * 5 > 4 is True.
   * 3 == 5 is False.
   * True or False evaluates to True.
   * not True evaluates to False.
5. **Expression 5**: (True and True) and (True == False)
   * True and True evaluates to True.
   * True == False evaluates to False.
   * So, True and False evaluates to False.

Here are the results:

* Expression 1: False
* Expression 2: False
* Expression 3: True
* Expression 4: False
* Expression 5: False

1. What are the six comparison operators?
2. **Less Than (**<**)**: Checks if the left operand is less than the right operand.
3. **Greater Than (**>**)**: Checks if the left operand is greater than the right operand.
4. **Equal To (**==**)**: Compares if the left operand is equal to the right operand.
5. **Not Equal To (**!=**)**: Verifies if the left operand is not equal to the right operand.
6. **Less Than or Equal To (**<=**)**: Determines if the left operand is less than or equal to the right operand

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

1. **Assignment Operator (**=**)**:
   * The = operator is used to **assign** a value to a variable.
   * Example: x = 10; assigns the value 10 to the variable x.
   * It does not compare values; instead, it sets the value of the left operand to the right operand.
2. **Equal To Operator (**==**)**:
   * The == operator checks whether the two given operands are **equal** or not.
   * If the values are equal, it returns true; otherwise, it returns false.
   * Example: 5 == 5 evaluates to true.
   * Use it when you need to compare values for equality, such as in conditional statements or loops.

8. 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')

* + spam = 0
  + if spam == 10:
  + print('eggs')
  + elif spam > 5:
  + print('bacon')
  + else:
  + print('ham')
  + print('spam')
  + print('spam')

1. 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.

spam = input()

if int(spam) == 1:

print('Hello')

elif int(spam) == 2:

print('Howdy')

else:

print('Greetings!')

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

1. **Windows/Linux**:
   * Press Ctrl + C to terminate the running program.
   * This sends a KeyboardInterrupt signal to the process, causing it to exit.
2. **macOS**:
   * Press Command + . (period) to interrupt the program.

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

break**Statement**:

* The break statement alters the flow of a loop by **terminating** it prematurely.
* When encountered, it immediately exits the loop, regardless of the loop condition.
* Use break when you need to stop the loop execution entirely.

continue**Statement**:

* The continue statement skips the **current iteration** of the loop and moves to the next one.
* It does not terminate the loop; instead, it jumps to the next iteration.
* Use continue when you want to skip specific iterations but continue looping.

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

1. range(10):
   * This generates a sequence of numbers starting from **0** and up to, but **not including**, **10**.
   * The default value for the **start** parameter is **0**, and the default value for the **step** parameter is **1**.
   * [So, range(10) is identical to range(0, 10, 1) 1](https://www.transtutors.com/questions/what-is-the-difference-between-range-10-range-0-10-and-range-0-10-1-in-a-for-loop--6702058.htm).
2. range(0, 10):
   * This also generates a sequence of numbers from **0** up to, but **not including**, **10**.
   * Explicitly specifies the **start** and **stop** values.
   * The **step** value defaults to **1**.
3. range(0, 10, 1):
   * Again, it generates a sequence from **0** up to, but **not including**, **10**.
   * Specifies the **start**, **stop**, and **step** values explicitly.
   * The **step** value is explicitly set to **1**.

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.

While loop

print("Numbers from 1 to 10:")

n = 1

while n <= 10:

print(n, end=" ")

n += 1

for loop

print("Numbers from 1 to 10:")

for num in range(1, 11):

print(num, end=" ")

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

from spam import bacon

bacon()