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

**The two values of the Boolean data type in Python are True and False.**

**You can write them directly in Python as follows:**

**True**

**False**

**It's important to note that in Python, the keywords True and False are case-sensitive, which means that if you write "true" or "false" instead, it will be interpreted as a variable or an undefined name, and you will get a NameError.**

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

**AND operator: The AND operator returns True if both operands are True, and False otherwise.**

**OR operator: The OR operator returns True if at least one of the operands is True, and False otherwise.**

**NOT operator: The NOT operator returns True if the operand is False, and False if the operand is True.**

**These operators are commonly used in Boolean expressions to evaluate conditions or test multiple conditions at once.**

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 ).

**AND operator truth table:**

|  |  |  |
| --- | --- | --- |
| Operand 1 | Operand 2 | Result |
| TRUE | TRUE | TRUE |
| TRUE | FALSE | FALSE |
| FALSE | TRUE | FALSE |
| FALSE | FALSE | FALSE |

**OR operator truth table:**

|  |  |  |
| --- | --- | --- |
| Operand 1 | Operand 2 | Result |
| TRUE | TRUE | TRUE |
| TRUE | FALSE | TRUE |
| FALSE | TRUE | TRUE |
| FALSE | FALSE | FALSE |

**NOT operator truth table:**

|  |  |
| --- | --- |
| Operand | Result |
| TRUE | FALSE |
| FALSE | TRUE |

4. What are the values of the following expressions?

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

**(5 > 4) and (3 == 5) evaluates to False.**

**Explanation: The first expression (5 > 4) is True, but the second expression (3 == 5) is False. The and operator returns False because both expressions need to be True for the entire expression to be True.**

not (5 > 4)

**not (5 > 4) evaluates to False.**

**Explanation: The expression (5 > 4) is True, and the not operator returns the opposite Boolean value, which is False.**

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

**(5 > 4) or (3 == 5) evaluates to True.**

**Explanation: The first expression (5 > 4) is True, and the or operator returns True because at least one of the expressions needs to be True for the entire expression to be True.**

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

**not ((5 > 4) or (3 == 5)) evaluates to False.**

**Explanation: The expression (5 > 4) or (3 == 5) is True, because the first expression is True. The not operator returns the opposite Boolean value, which is False.**

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

**(True and True) and (True == False) evaluates to False.**

**Explanation: The first expression (True and True) is True, but the second expression (True == False) is False. The and operator returns False because both expressions need to be True for the entire expression to be True.**

(not False) or (not True)

**(not False) or (not True) evaluates to True.**

**Explanation: The first expression not False is True, and the second expression not True is False. The or operator returns True because at least one of the expressions needs to be True for the entire expression to be True.**

5. What are the six comparison operators?

**Equal to: ==**

This operator checks if two values are equal to each other.

**Not equal to: !=**

This operator checks if two values are not equal to each other.

**Greater than: >**

This operator checks if the left operand is greater than the right operand.

**Less than: <**

This operator checks if the left operand is less than the right operand.

**Greater than or equal to: >=**

This operator checks if the left operand is greater than or equal to the right operand.

**Less than or equal to: <=**

This operator checks if the left operand is less than or equal to the right operand.

*These operators are used to compare values and create Boolean expressions that evaluate to either True or False. They are commonly used in conditional statements and loops to control the flow of program execution based on certain conditions.*

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

the equal to operator is ==, which is used to compare two values for equality. On the other hand, the assignment operator is =, which is used to assign a value to a variable.

To differentiate between the two operators, you need to look at their usage and the context in which they are used. The equal to operator is used in Boolean expressions to check if two values are equal, while the assignment operator is used to assign a value to a variable.

Here's an example that demonstrates the difference between the two operators:

*# Assigning a value to a variable using the assignment operator*

*x = 5*

*# Using the equal to operator to compare two values*

*if x == 5:*

*print("x is equal to 5")*

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')

**Block 1 consists of a single statement that assigns the value 0 to the variable spam. The subsequent if statement is not executed because spam is not equal to 10.**

**Block 2 contains another if statement that checks if spam is greater than 5. Since spam is not greater than 5, the else block is executed.**

**Block 3 consists of a single statement that prints the string 'ham'. It is executed because the condition in Block 2 was not met.**

**The last two lines are not part of any block and will always be executed, regardless of the conditions in the if statements.**

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.

*if spam == 1:*

*print("Hello")*

*elif spam == 2:*

*print("Howdy")*

*else:*

*print("Greetings!")*

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

**If your program is stuck in an endless loop, you can press Ctrl + C (i.e., the Ctrl key and the C key at the same time) to stop the program.**

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

**The main difference between them is that break is used to exit out of a loop completely, while continue is used to skip over an iteration of the loop and continue with the next iteration.**

*# Example using break*

*for i in range(1, 6):*

*if i == 3:*

*break*

*print(i)*

*# Output: 1 2*

*# Example using continue*

*for i in range(1, 6):*

*if i == 3:*

*continue*

*print(i)*

*# Output: 1 2 4 5*

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

**In a for loop, range(10), range(0, 10), and range(0, 10, 1) are three different ways to generate a sequence of numbers that can be used to iterate over in the loop.**

*range(10) generates a sequence of numbers starting from 0 up to, but not including, 10.*

*range(0, 10) generates a sequence of numbers starting from 0 up to, but not including, 10.*

*range(0, 10, 1) generates a sequence of numbers starting from 0 up to, but not including, 10, with a step size of 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.

*# Using for loop*

*for i in range(1, 11):*

*print(i)*

*# Using while loop*

*i = 1*

*while i <= 10:*

*print(i)*

*i += 1*

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

*import spam*

*spam.bacon()*