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

Ans. The two values of the Boolean data type are True and False. True and False are used in conditional statements. Boolean values are often the result of logical operations or comparisons in Python. For example, the comparison expression 5 > 3 evaluates to True, while the logical expression 10 == 5 evaluates to False.

The two values of the Boolean data type in Python are written as follows:

- `True`: This represents the boolean value of true and is written as the keyword `True` with the first letter capitalized.

- `False`: This represents the boolean value of false and is written as the keyword `False` with the first letter capitalized.

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

Ans. Three types of Boolean operators are there Logical AND ('and'), Logical OR ('or') and Logical NOT ('not').

Logical AND (and): The 'and' operator returns True if both operands are True, and False otherwise. It evaluates expressions from left to right and stops evaluating as soon as it encounters a False operand.

Logical OR (or): The 'or' operator returns True if at least one of the operands is True, and False if both operands are False. It evaluates expressions from left to right and stops evaluating as soon as it encounters a True operand.

Logical NOT (not): The 'not' operator is a unary operator that negates the value of its operand. It returns True if the operand is False, and False if the operand is True.

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 ).

Ans. Logical AND ('and'):

|  |  |  |
| --- | --- | --- |
| **Operand 1** | **Operand 2** | **Result** |
| False | False | False |
| False | True | False |
| True | False | False |
| True | True | True |

Logical OR (or):

|  |  |  |
| --- | --- | --- |
| **Operand 1** | **Operand 2** | **Result** |
| False | False | False |
| False | True | True |
| True | False | True |
| True | True | True |

Logical NOT (not):

|  |  |
| --- | --- |
| **Operand** | **Result** |
| False | True |
| True | False |

4. What are the values of the following expressions?

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

(5 > 4) = True

(3==5) = False

Since, the and operator returns True only if both operands are True. Therefore, the overall expression evaluates to False.

b. not (5 > 4)

(5 > 4) = True

Since the operand is True, the not operator returns False. Therefore, the expression evaluates to False.

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

(5 > 4) = True

(3==5) = False

The or operator returns True if at least one operand is True. Therefore, the overall expression evaluates to True.

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

(5 > 4) = True

(3 == 5) = False

Due to the or operator the inner value of the expression evaluates to True. But the not operator negates the value of the inner expression, resulting in False.

e. (True and True) and (True == False)

(True and True) = True

(True == False) = False

The and operator returns True only if both operands are True. Therefore, the overall expression evaluates to False.

f. (not False) or (not True)

(not False) = True

(not True) = False

The or operator returns True if at least one operand is True. Therefore, the overall expression evaluates to True.

5. What are the six comparison operators?

Ans. The six comparison operators in Python are:

Equal to (==)

Not equal to (!=)

Greater than (>)

Less than (<)

Greater than or equal to (>=)

Less than or equal to (<=)

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

Equal to operator uses two equal symbols i.e., ==. It is used for comparison. The operator compares the values from it's left hand side and right hand side and evaluates if they are equal or not. The result of the operator is always either True or False. Example, x == y checks if the value of x is equal to the value of y.

The single equal to (=) operator is used for assignment. It assigns the value on the right side of the operator to the variable on the left side. Example: x = 5 assigns the value 5 to the variable x.

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')

Ans. 1. Block 1:

spam = 0

if spam == 10:

print('eggs')

2. Block 2:

if spam > 5:

print('bacon')

else:

print('ham')

3. Block 3:

print('spam')

print('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.

Ans.

spam= int(input("Enter a number for 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?

I will press Ctrl+C to stop the loop.

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

When encountered inside a loop, the break statement immediately terminates the loop and transfers control to the next statement after the loop, it exits the loop. When encountered inside a loop, the continue statement skips the rest of the current iteration and jumps to the next iteration of the loop, it doesn't exit the loop rather skips the remaining code within the loop.

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

Ans. In a `for` loop, the `range(10)` expression generates values from 0 to 9. Since no starting value is specified, it assumes the starting value as 0. The ending value is provided, but it's excluded, so the loop will print the values up to 9. As no step size is given, it defaults to 1, resulting in an increment of 1 between each value.

For `range(0, 10)`, both the starting and ending values are explicitly provided. Since no step size is specified, it assumes a step size of 1. Therefore, it will print the values 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. The ending value of 10 is excluded.

In `range(0, 10, 1)`, the starting value is 0, the ending value is 10, and the step size is 1. This means the values will be written between 0 and 10 with an increment of 1. The output will be the same as before: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 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.

Ans.

Short program-

for num in range(1, 11):

print(num)

Program using while loop-

num = 1

while num <= 10:

print(num)

num += 1

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

Ans. The module name is 'spam', and the function name is 'bacon()'. To call the 'bacon()' function from the 'spam' module, we need to use the module name followed by a dot and then the function name, like 'spam.bacon()'. This allows us to access and execute the 'bacon()' function within the 'spam' module.