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.

True and False must be written with capital letters in Python.

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

There are three different types of Boolean operators in Python:

and: The and operator returns True if both the operands (i.e., expressions being evaluated) are True, and False otherwise.

or: The or operator returns True if either of the operands is True, and False otherwise.

not: The not operator negates the value of a Boolean expression, i.e., not True is False, and not False is True.

For Example,

>>> True and False

False

>>> True or False

True

>>> not True

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 ).

Here are the truth tables for each Boolean operator in Python:

1)and operator:

True and True = True

True and False = False

False and True = False

False and False = False

2) or operator:

True or True = True

True or False = True

False or True = True

False or False = False

3) not operator:

not True = False

not False = 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](5 > 4) and (3 == 5) evaluates to False because (5 > 4) is True, but (3 == 5) is False, so the expression as a whole is False.

2]not (5 > 4) evaluates to False because (5 > 4) is True, and not True is False.

3](5 > 4) or (3 == 5) evaluates to True because (5 > 4) is True, and the expression as a whole is True.

4]not ((5 > 4) or (3 == 5)) evaluates to False because (5 > 4) or (3 == 5) is True, and not True is False.

5](True and True) and (True == False) evaluates to False because (True and True) is True, but (True == False) is False, so the expression as a whole is False.

6](not False) or (not True) evaluates to True because not False is True, and not True is False, so the expression as a whole is True.

5. What are the six comparison operators?

The six comparison operators in Python are:

== (equal to): Returns True if the operands are equal, and False otherwise.

!= (not equal to): Returns True if the operands are not equal, and False otherwise.

> (greater than): Returns True if the left operand is greater than the right operand, and False otherwise.

< (less than): Returns True if the left operand is less than the right operand, and False otherwise.

>= (greater than or equal to): Returns True if the left operand is greater than or equal to the right operand, and False otherwise.

<= (less than or equal to): Returns True if the left operand is less than or equal to the right operand, and False otherwise.

For Example

>>> 5 == 5

True

>>> 5 != 4

True

>>> 5 > 4

True

>>> 4 < 5

True

>>> 5 >= 5

True

>>> 4 <= 5

True

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

The difference between the "equal to" operator == and the assignment operator = in Python is that == is used for comparison, while = is used for assignment.

The "equal to" operator == returns True if the values on either side of the operator are equal, and False otherwise. For example:

>>> x = 5

>>> y = 6

>>> x == y

False

The assignment operator = assigns a value to a variable. For example:

>>> x = 5

>>> y = 6

>>> x = y

>>> x

6

You would use the "equal to" operator == when you want to compare two values to see if they are equal.

You would use the assignment operator = when you want to assign a value to a variable.

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 this code, there are three blocks:

The first block is the block for the first if statement:

if spam == 10:

print('eggs')

The second block is the block for the second if statement:

if spam > 5:

print('bacon')

else:

print('ham')

The third block is the block that contains the remaining print statements:

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.

spam = 1

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?

CTRL + C

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

The break and continue statements in Python are used to control the flow of a loop. The difference between the two statements is as follows:

break: The break statement is used to exit a loop prematurely. When the break statement is executed, the loop is terminated and the program continues with the next statement after the loop. For example:

for i in range(10):

if i == 5:

break

print(i)

This code will print the numbers 0 to 4, and then stop the loop because i is equal to 5.

continue: The continue statement is used to skip the current iteration of a loop and continue with the next iteration. When the continue statement is executed, the program skips the rest of the statements in the current iteration and continues with the next iteration. For example:

for i in range(10):

if i % 2 == 0:

continue

print(i)

This code will print the odd numbers from 1 to 9, because the continue statement skips the current iteration when i is an even number.

In summary, break is used to exit a loop completely, while continue is used to skip the current iteration of a loop and continue with the next iteration.

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

range(10): This generates a sequence of numbers from 0 to 9 (10 numbers in total).

range(0, 10): This generates a sequence of numbers from 0 to 9 (10 numbers in total).

range(0, 10, 1): This generates a sequence of numbers from 0 to 9 (10 numbers in total), with a step of 1 between each number.

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.program that prints the numbers 1 to 10 using a for loop in Python:

for i in range(1, 11):

print(i)

program that prints the numbers 1 to 10 using a while loop in Python:

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()