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

ANSWER

The Boolean data type represents a logical value that can have one of two states: true or false.

Example : x=True

y=False

In this example, the variable x is assigned the value True, and the variable y is assigned the value False. These values can then be used in conditional statements, loops, and other constructs that require logical evaluations.

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

ANSWER

Logical AND : The logical AND operator returns true if both operands are true; otherwise, it returns false.

Logical OR : The logical OR operator returns true if at least one of the operands is true; otherwise, it returns false.

Logical NOT : The logical NOT operator negates the boolean value of the operand. If the operand is true, the logical NOT operator returns false, and if the operand is false, it returns 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 evaluates).

ANSWER

Logical AND operator:

Operand 1	Operand 2	Result
False	False	False
False	True	False
True	False	False
True	True	True

Logical OR operator:

| Operand 1 | Operand 2 | Result |

| False | False | False |

| False | True | True |

| True | False | True |

| True | True | True |

Logical NOT operator:

| Operand | Result |

| False | True |

| True | False |

4. What are the values of the following expressions?

ANSWER

(5 > 4) and (3 == 5) :**False**

not (5 > 4) : **False**

(5 > 4) or (3 == 5) : **True**

not ((5 > 4) or (3 == 5)) : **False**

(True and True) and (True == False) : **False**

(not False) or (not True) : **True**

5. What are the six comparison operators?

ANSWER

1. Equal to (==)
2. Not equal to (!=)
3. Greater than (>)
4. Lesser than (<)
5. Greater than or equal to (>=)
6. Lesser 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.

ANSWER

The equal to operator (==) is used to compare the equality of two values, whereas the assignment operator (=) is used to assign a value to a variable.

Here's an example of using the equal to operator in a conditional statement:

```
x = 5

y = 10

if x == y:

    print("x is equal to y")

else:

    print("x is not equal to y")
```

In this example, the equal to operator (==) compares the values of `x` and `y` and determines if they are equal.

Here's an example of using the assignment operator to assign a value to a variable:

```
x = 5

y = x
```

In this example, the assignment operator (=) assigns the value of **x** to the variable **y**.

It's important to note that the assignment operator does not check for equality but rather performs an assignment of 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')
```

ANSWER

```
spam=0

If spam==10:

        print('eggs')


If spam >5:

        print('bacon')

else:

        print('ham')

        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.

ANSWER

```
spam=int(input("enter the value for spam i.e 1,2,...:"))

if int(spam)==1:

    print("hello")

elif int(spam)==2:

    print("howdy")

else:

    print("Greetings!")
```

9.If your program is stuck in an endless loop, what keys will you press?

ANSWER

By pressing Ctrl + C, you are sending a signal to the operating system, which is then received by the Python interpreter. This action is commonly used to interrupt the execution of a program that is not responding or stuck in an infinite loop.

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

ANSWER

break statement: When encountered within a loop, the `break` statement immediately terminates the loop and resumes execution at the next statement after the loop. It effectively breaks out of the loop's iteration, regardless of whether the loop condition is still true or not.

Here's an example that demonstrates the usage of break:

```
for i in range(1, 10):

    if i == 5:
```

```
        break

    print(i)
```

In this example, the loop will iterate from 1 to 9. However, when **i** becomes equal to 5, the **break** statement is encountered, causing the loop to terminate. The output will be **1 2 3 4.**

continue statement: When encountered within a loop, the continue statement skips the remaining code within the loop for the current iteration and moves on to the next iteration. It essentially jumps back to the beginning of the loop for the next iteration.

Here's an example that demonstrates the usage of continue:

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

    if i == 3:

        continue

    print(i)
```

In this example, when **i** equals 3, the **continue** statement is encountered, causing the loop to skip the remaining code for that iteration. The output will be **1 2 3 4**

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

range(0, 0,1)?

ANSWER

range(10): This statement creates a sequence of numbers starting from 0 and ending at 9 (10 numbers in total). The range() function is called with a single argument, which represents the stopping point of the sequence. The starting point is assumed to be 0 by default, and the step size is assumed to be 1.

range(0,10): This statement also creates a sequence of numbers starting from 0 and ending at 9 (10 numbers in total). Here, the range() function is called with two arguments: the starting point and the stopping point. The step size is still assumed to be 1 by default.

range(0,0,1): This statement creates an empty sequence because the starting point (0) and the stopping point (0) are the same. The range() function is called with three arguments: the starting point, the stopping point, and the step size. In this case, the step size is 1, but since the starting and stopping points are the same, there are no numbers in the sequence.

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.

ANSWER

Using for loop

```
for i in range(1,11):
    print(i)
```

Using while loop

```
i=0
while (i<10):
    i=i+1
    print(i)
```

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

ANSWER

```
import spam
spam.bacon()
```