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

Ans: Two values of Boolean data type is ‘0’ and ‘1’.

‘0’ write as ‘false’ ‘1’ write as ‘True’.

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

Ans: 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 evaluates ).

Ans: Below truth tables for each of the five basic Boolean operators:

1. AND operator:

| **A** | **B** | **A AND B** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

1. OR operator:

| **A** | **B** | **A OR B** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

1. NOT operator:

| **A** | **NOT A** |
| --- | --- |
| 0 | 1 |
| 1 | 0 |

1. XOR operator:

| **A** | **B** | **A XOR B** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

1. NAND operator:

| **A** | **B** | **A NAND B** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Note that in the truth tables, 0 represents "false" and 1 represents "true".

4. What are the values of the following expressions?

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

not (5 > 4) TRUE

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

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

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

(not False) or (not True) True

5. What are the six comparison operators?

i.==(Equal)

ii.!=(Not Equal)

iii.Greater Than >

iv.Less Than <

v.Greater than or equal to >=

vi.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.

The “=” is an assignment operator is used to assign the value on the right to the variable on the left. The '==' operator checks whether the two given operands are equal or not. If so, it returns true.

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: sky colour is the first block.

Red colour here is the second Block.

Violet colour is here in the Third block.

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:Here's the Python code to print "Hello" if the value of the variable spam is 1, "Howdy" if the value of spam is 2, and "Greetings!" for any other value:

spam=3

if spam==1:

print (“Hello”)

elif spam==2:

print(“Howdy”)

else:

print(“Greetings!”)

pam

In this code, we use an if statement to check the value of the spam variable. If spam is equal to 1, we print "Hello". If spam is equal to 2, we print "Howdy". If spam is not equal to 1 or 2, we print "Greetings!".

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

If a program is stuck in an endless loop and is not responding, you can try the following steps to break out of the loop:

1. Press the "Ctrl" and "C" keys simultaneously: This sends a SIGINT signal to the program, which is a request for it to terminate.
2. If the above step does not work, you can try to force quit the program by pressing the "Ctrl" and "Alt" keys and then the "Delete" key. This will bring up the Task Manager, where you can select the program and choose to end the task.
3. If the program is still not responding, you may need to restart your computer to terminate the process.

It's worth noting that if you have unsaved work in the program, you may lose it when you force quit or restart your computer.

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

Ans:Both break and continue are control flow statements used in loops, but they have different effects on the loop's execution.

break is used to immediately terminate the loop's execution and continue with the next statement following the loop. When break is encountered inside a loop, the loop's body is immediately exited, and control is transferred to the statement immediately following the loop.

For example, consider the following code:

for i in range(1, 6):

if i == 3:

break

print(i)

print("Loop ended.")

In this code, the loop prints the values from 1 to 2, but when i is equal to 3, the break statement is executed, and the loop immediately terminates. The program then continues with the statement after the loop, which is to print "Loop ended."

On the other hand, continue is used to skip the current iteration of the loop and continue with the next iteration. When continue is encountered inside a loop, the current iteration's remaining statements are skipped, and control is transferred back to the top of the loop for the next iteration.

For example, consider the following code:

for i in range(1, 6):

if i == 3:

continue

print(i)

print("Loop ended.")

In this code, the loop prints the values from 1 to 5, but when i is equal to 3, the continue statement is executed, and the loop skips the remaining statements for that iteration. The program then continues with the next iteration of the loop.

In summary, break is used to immediately terminate the loop's execution and continue with the next statement following the loop, while continue is used to skip the current iteration of the 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)?**

Ans:In a for loop, range(10), range(0, 10), and range(0, 10, 1) are equivalent, but there are subtle differences between them that are worth noting.

range(10) creates a sequence of numbers from 0 up to (but not including) 10. This is equivalent to range(0, 10).

range(0, 10) creates a sequence of numbers starting from 0 up to (but not including) 10. The first argument (0) is the starting value, and the second argument (10) is the stopping value.

range(0, 10, 1) creates a sequence of numbers starting from 0 up to (but not including) 10, with a step of 1. The third argument (1) is the step value, which determines the increment between each number in the sequence.

In practice, these differences are not very significant when using them in a for loop, as they all generate the same sequence of integers. However, range() function's flexibility allows you to generate a sequence of integers with different starting values, stopping values, and step sizes, which can be useful in some cases.

It's worth noting that in Python 3, range() returns a generator object rather than a list, which can be more efficient when working with large sequences. If you need to use the sequence as a list, you can convert it using the list() function.

Regenerate response

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.

Here is a short program that prints the numbers 1 to 10 using a for loop in Python:

for i in range(1, 11):

print(i)

This will output:

1

2

3

4

5

6

7

8

9

10

Here is an equivalent program that prints the numbers 1 to 10 using a while loop:

In the while loop version, we start with i initialized to 1, and then repeatedly execute the loop body while i is less than or equal to 10. Inside the loop, we print the current value of i, and then increment it by 1 using i += 1. This continues until i becomes greater than 10, at which point the loop terminates.

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

Assuming that the bacon() function is defined in the spam module, you can call it after importing the spam module by using dot notation to specify the module name and the function name:

import spam

spam.bacon()

This code imports the spam module and then calls the bacon() function that is defined inside it. The spam. prefix specifies the module name, and bacon() specifies the function name.

Alternatively, if you only need to use the bacon() function from the spam module, you can import just the bacon() function directly into your code:

from spam import bacon

bacon()

This code imports only the bacon() function from the spam module, allowing you to call it directly without using the spam. prefix.