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

Soln:

Boolean data type consists of only 2 values. They are True or False.

For e.g., if we give input as 10 > 20, it will give output as False

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

Soln:

There are 3 boolean operators to compare values. They are:

AND, OR , NOT

They evaluate the expressions and result in Boolean values True or False. Boolean represents truth value of an expression.

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 ).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Boolean Operators** | **Details** | **Input** | **Remarks** | **Output** |
| **AND** | True only if both are True | (10 > 1) and (16 == 16) | True + True | True |
| (10 > 1) and (16 > 20) | True + False | False |
| (10 > 30) and (16 > 20) | False + False | False |
| **OR** | True if at least one in True | (10 > 1) and (16 == 16) | True + True | True |
| (10 > 1) and (16 > 20) | True + False | True |
| (10 > 30) and (16 > 20) | False + False | False |
| **NOT** | True only if it is False. It is opposite of original expression | not(10 > 15) | False | True |
| not(10 < 15) | True | False |

4. What are the values of the following expressions?

|  |  |  |
| --- | --- | --- |
| **Input** | **Remarks** | **Output** |
| (5 > 4) and (3 == 5) | True and False  It is True, only if both are True | False |
| not (5 > 4) | not(True)  it is opposite of original expression | False |
| (5 > 4) or (3 == 5) | True or False  It is True , if at least one expression is True | True |
| not ((5 > 4) or (3 == 5)) | First evaluates True or False , so True  Then evaluates not(True) , so False | False |
| (True and True) and (True == False) | True and True is True  True == False is False  So, end result of True and False is False | False |
| (not False) or (not True) | not False is True  not True is False  So, end result of True or False is True | True |

5. What are the six comparison operators?

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Input** | **Output** |
| == | Equals | 2 == 2 | True |
| != | Not equal to | 7 != 3 | True |
| < | Greater than | 5 > 10 | False |
| > | Less than | 6 < 8 | True |
| <= | Greater than or equal to | 8 >= 5 | True |
| >= | Less than or equal to | 7 <= 1 | False |

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

|  |  |  |
| --- | --- | --- |
| **Operator** | **Sign** | **Description** |
| Assignment | = | It will assign a value to variable. It may not give any output  x = 4 . it will assign value 4 to x |
| Equals to | == | It will equate the value. It is comparison operator. It will always give Boolean output.  Eg. 4 == 6 output is False |

7. Identify the three blocks in this code:

|  |  |  |
| --- | --- | --- |
| **Question** | **Solution** | **Remarks** |
| spam = 0  if spam == 10:  print('eggs')  if spam > 5:  print('bacon')  else:  print('ham')  print('spam')  print('spam') | spam = 0  if spam == 10:  print('spam')  if spam > 5:  print('spam')  else:  print('ham') | This is the block of code.  The output here will be ‘ham’, since spam is zero |
|  | print('eggs')  print('bacon') | This is not related to above code. These print statements will just print the value/strings given within them  Output : ‘eggs’  Output: ‘bacon’ |

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 = 7  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?

To end an endless loop, we have to press CNTR + C . This will raise a KeyBoardInterruptError and terminates the whole program.

We can also, Interrupt or Restart the Kernel

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

|  |  |
| --- | --- |
| **Break Statement** | **Continue Statement** |
| It terminates the loop immediately when it is encountered | It skips the current iteration of the loop. Then the control of the program jumps to next iteration |
| It is used to control the sequence of the loop. It terminates that loop and moves to next code | It passes control to the next iteration |
| for i in range(6):  if i==3:  break  print(i) | for i in range(6):  if i==3:  continue  print(i) |
| Output:  0  1  2 | Output:  0  1  2  4  5 |
| Here, when condition is met i.e., i==3, it ends the loop | Here, when condition is met i.e., i==3,it skips 3, and continues with next iteration which is 4 |

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

Solution:

range(start\_number, end\_number, number\_of\_steps)

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Details** | **Example: Input** | **Output** |
| range(10) | 10 indicates the last number of the loop. In python the last number is excluded, it will be upto 9 | for i in range(5):  print(i) | 0,1,2,3,4 |
| range(0,10) | 0 indicates the first number of the loop  10 indicates the last number of the loop | for i in range(1,5):  print(i) | 1,2,3,4 |
| range(0,10,1) | 0 indicates the first number of the loop  10 indicates the last number of the loop  1 indicates the number of steps between each number | for i in range(1,7,2):  print(i) | 1,3,5 |

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.

|  |  |
| --- | --- |
| **For Loop** | **While Loop** |
| for i in range(1,11):  print(i) | 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?

|  |  |
| --- | --- |
| **Code** | **Description** |
| import spam | Importing module spam |
| from spam import bacon | Importing the required function bacon |
| bacon() | Calling the function |