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

**Answer: T**he Boolean data type has two possible values: True and False.

In most programming languages, including Python, these values are written as True and False (without quotes) and are used to represent the concepts of true and false in a program.

1. **What are the three different types of Boolean operators?**

**Answer:**

1. **and** - This operator returns True if both operands are True, and False otherwise.
2. **or** - This operator returns True if at least one of the operands is True, and False otherwise.
3. **not** - This operator returns True if the operand is False, and False if the operand is True.
4. **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 ).**

**Answer:**   
Here are the truth tables for each of the Boolean operators in Python:

1. **and** operator:

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

1. **or** operator:

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

1. **not** operator:

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

1. **What are the values of the following expressions?**

(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**

1. **What are the six comparison operators?**

**Answer:** The six comparison operators in Python are:

1. < (less than)
2. > (greater than)
3. <= (less than or equal to)
4. >= (greater than or equal to)
5. == (equal to)
6. != (not equal to)
7. **How do you tell the difference between the equal to and assignment operators? Describe a condition and when you would use one.**

**Answer:** The difference between the equal to (**==**) and assignment (**=**) operators can be easily distinguished by their usage.

* The equal to (**==**) operator is used to compare values and determine if they are equal. For example, **x == y** returns **True** if the values of **x** and **y** are equal, and **False** otherwise.
* The assignment (**=**) operator, on the other hand, is used to assign a value to a variable. For example, **x = y** assigns the value of **y** to the variable **x**.

In a condition, you would use the equal to operator to determine if a certain condition is met (e.g. **if x == y:**) and you would use the assignment operator to assign a value to a variable (e.g. **x = y**).

1. **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:**

1. spam = 0
2. if spam == 10:  
   print('eggs')
3. if spam > 5:  
   print('bacon')  
   else:  
   print('ham')  
   print('spam')  
   print('spam')
4. **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:**

if spam == 1:

print("Hello")

elif spam == 2:

print("Howdy")

else:

print("Greetings!")

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

**Answer:** If a program is stuck in an endless loop in Python, we can interrupt the execution of the program by pressing Ctrl + C (on Windows) or Cmd + C (on Mac).

But in jupyternote book, we an interrupt the execution of the program by clicking on the "Interrupt the kernel" option in the Kernel menu or by pressing I, I in rapid succession.

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

**Answer:** The "break" statement will immediately exit the loop and move on to the next line of code after the loop.

The "continue" statement will immediately skip to the next iteration of the loop and continue execution from there, skipping any code after the "continue" statement within the current iteration.

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

**Answer:** The range function is used to generate a sequence of numbers, which is often used to control the iteration of a for loop.

All 3 will generates a sequence of numbers from 0 to 9 (10 numbers in total).

The difference between the three is the start and step arguments. range(10) is a shorthand for range(0, 10), and range(0, 10, 1) specifies the start and step explicitly.

1. **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:** Here is a program that prints the numbers 1 to 10 using a for loop:

for i in range(1, 11):

print(i)

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

i = 1

while i <= 10:

print(i)

i += 1

1. **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()