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

The two values of the Boolean data type are **True** and **False**. They are written exactly as shown, with the initial letter capitalized.

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

The three different types of Boolean operators are:

* AND operator (**and**)
* OR operator (**or**)
* NOT operator (**not**)

3. Here are the truth tables for each Boolean operator:

AND operator (`and`):

```

True and True = True

True and False = False

False and True = False

False and False = False

```

OR operator (`or`):

```

True or True = True

True or False = True

False or True = True

False or False = False

```

NOT operator (`not`):

```

not True = False

not False = True

```

4. Values of the given 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

5. The six comparison operators are:

- Equal to (`==`)

- Not equal to (`!=`)

- Greater than (`>`)

- Less than (`<`)

- Greater than or equal to (`>=`)

- Less than or equal to (`<=`)

6. The equal to operator (`==`) is used to compare two values for equality, while the assignment operator (`=`) is used to assign a value to a variable. To differentiate them:

- Equal to (`==`) is used in conditions to check if two values are equal, for example, `if x == 5:`.

- Assignment operator (`=`) is used to assign a value to a variable, for example, `x = 5`.

7. The three blocks in the code are identified by the indentation level:

- Block 1: `spam = 0`

- Block 2: `if spam == 10:` (with its nested `print('eggs')`)

- Block 3: `if spam > 5:` (with its nested `print('bacon')`) and the `else:` block (with its nested `print('ham')`, `print('spam')`, and `print('spam')`)

8. Here is the code that prints different messages based on the value of `spam`:

```python

spam = 1 # You can change this value to test different cases

if spam == 1:

print('Hello')

elif spam == 2:

print('Howdy')

else:

print('Greetings!')

```

9. If your program is stuck in an endless loop, you can press `Ctrl+C` (or `Cmd+C` on macOS) in most command-line environments to interrupt the program and stop its execution.

10. The `break` statement is used to exit the current loop and resume execution after the loop. The `continue` statement, on the other hand, is used to skip the rest of the current iteration and move to the next iteration of the loop.

11. In a `for` loop, `range(10)`, `range(0, 10)`, and `range(0, 10, 1)` are all equivalent. They all generate a sequence of numbers from 0 to 9. The third argument in `range()` specifies the step or increment, which is 1 by default.

12. Using a `for` loop:

```python

for i in range(1, 11):

print(i)

```

Using a `while` loop:

```python

i = 1

while i <= 10:

print(i)

i += 1

```

13. If you had a function named `bacon()` inside a module named `spam`, you can call it after importing `spam` like this:

```python

import spam

spam.bacon()

```