**Truthy vs Falsy Values in Python**

Last Updated : 05 Sep, 2020

In this article, we will see about the concept of **Truthy**and **Falsy**values in Python and also see how to determine a value is **Truthy**or **Falsy**by using **bool()** built-in Python function.

Now, Let’s start with the following two codes:

* Python3

|  |
| --- |
| number **=** 7  **if** number:    print(number) |

**Output:**

7

Let’s change value of number to 0

* Python3

|  |
| --- |
| number **=** 0  **if** number:    print(number) |

**Output:**

There is no Output

Have you wondered why the above code run successfully despite **number**not being an expression?

The answer lies in concept of **Truthy**and **Falsy**Values.

**Truthy vs Falsy Values**

In Python, individual values can evaluate to either **True**or **False**.

The Basis rules are:

* Values that evaluate to False are considered **Falsy**.
* Values that evaluate to True are considered **Truthy**.

**Falsy Values Includes:**

**1) Sequences and Collections:**

* Empty lists []
* Empty tuples ()
* Empty dictionaries {}
* Empty sets set()
* Empty strings ” “
* Empty ranges range(0)

**2) Numbers: Zero of any numeric type.**

* Integer: 0
* Float: 0.0
* Complex: 0j

**3) Constants:**

* None
* False

**Falsy**values were the reason why there was no output in our initial example when the value of **number**was zero.

**Truthy Values Includes:**

* Non-empty sequences or collections (lists, tuples, strings, dictionaries, sets).
* Numeric values that are not zero.
* Constant: True

This is why the value of a printed in our initial example because its value of a **number**was **7**(a **truthy**value):

**Built-in bool() function**

You can check if a value is either **truthy**or **falsy**with the built-in[**bool()**](https://www.geeksforgeeks.org/bool-in-python/)function. This function is used to return or convert a value to a Boolean value i.e., True or False, using the standard truth testing procedure

***Syntax:*** *bool(parameter)*

You only need to pass the value as an argument.

**Example:**

* Python3

|  |
| --- |
| bool(7)  # True    bool(0)   #False    bool([])  # False    bool({7,4})  #True    bool(**-**4)  # True    bool(0.0)  # False    bool(None)  # False    bool(1)  #True    bool(range(0))  # False    bool(set())  # False    bool([1,2,3,4])  # True |

**Output:**

True

False

False

True

True

False

False

True

False

False

True

Now let’s see a program for better understanding of Truthy and Falsy value.

**Example:**

* Python3

|  |
| --- |
| # define a function for checking  # number is even or odd  **def** even\_odd(number):    **if** number **%** 2:         # since num % 2 is equal to 1       # which is Truthy Value  **return** 'odd number'    **else**:         # since num%2 is equal to 0       # which is Falsy Value.  **return** 'even number'    result1 **=** even\_odd(7)    # prints odd  print(result1)    result2 **=** even\_odd(4)    # prints even  print(result2) |

**Output:**

odd number

even number

Since in first function call num % 2 is equal to 1 which is **Truthy**Value, so output is **‘odd number’**and in second function call num % 2 is equal to 0 which **Falsy**Value, so output is **‘even number’.**