

## Booleans represent one of two values: True or False.

### Boolean Values

In programming you often need to know if an expression is True or False.

You can evaluate any expression in Python, and get one of two answers, True or False.

When you compare two values, the expression is evaluated and Python returns the Boolean answer:

```
In [2]: print(10 > 9)
        print(10 == 9)
        print(10 < 9)
```

```
True
False
False
```

```
In [3]: # Print a message based on whether the condition is True or False:
```

```
a = 200
b = 33

if b > a:
    print("b is greater than a")
else:
    print("b is not greater than a")
```

```
b is not greater than a
```

### Evaluate Values and Variables

The bool() function allows you to evaluate any value, and give you True or False in return,

```
In [4]: # Evaluate a string and a number:
```

```
print(bool("Hello"))
print(bool(15))
```

```
True
True
```

In [5]: *# Evaluate two variables:*

```
x = "Hello"
y = 15

print(bool(x))
print(bool(y))
```

True

True

## Most Values are True

Almost any value is evaluated to True if it has some sort of content.

Any string is True, except empty strings.

Any number is True, except 0.

Any list, tuple, set, and dictionary are True, except empty ones.

In [7]: *# The following will return True:*

```
bool("abc")
bool(123)
bool(["apple", "cherry", "banana"])
```

Out[7]: True

## Some Values are False

In fact, there are not many values that evaluate to False, except empty values, such as (), [], {}, "", the number 0, and the value None. And of course the value False evaluates to False.

In [8]: *# The following will return False:*

```
bool(False)
bool(None)
bool(0)
bool("")
bool(())
bool([])
bool({})
```

Out[8]: False

**One more value, or object in this case, evaluates to False, and that is if you have an object that is made from a class with a len function that returns 0 or False:**

```
In [9]: class myclass():
        def __len__(self):
            return 0

myobj = myclass()
print(bool(myobj))
```

False

## Functions can Return a Boolean

You can create functions that returns a Boolean Value:

```
In [10]: # Print the answer of a function:

def myFunction() :
    return True

print(myFunction())
```

True

```
In [11]: # Print "YES!" if the function returns True, otherwise print "NO!":

def myFunction() :
    return True

if myFunction():
    print("YES!")
else:
    print("NO!")
```

YES!

**Python also has many built-in functions that return a boolean value, like the isinstance() function, which can be used to determine if an object is of a certain data type:**

```
In [12]: # Check if an object is an integer or not:

x = 200
print(isinstance(x, int))
```

True