2.2. Values and Data Types

A **value** is one of the fundamental things — like a word or a number — that a program manipulates. Some values are `5` (the result when we add `2 + 3`), and `"Hello, World!"`. These objects are classified into different classes, or data types: 4 is an integer, and “Hello, World!” is a string, so-called because it contains a string or sequence of letters. You (and the interpreter) can identify strings because they are enclosed in quotation marks.

We can specify values directly in the programs we write. For example we can specify a number as a **literal** just by (literally) typing it directly into the program (e.g., `5` or `4.32`). In a program, we specify a word, or more generally a string of characters, by enclosing the characters inside quotation marks (e.g., `"Hello, World!"`).

During execution of a program, the Python interpreter creates an internal representation of literals that are specified in a program. It can then manipulate them, for example by multiplying two numbers. We call the internal representations **objects** or just **values**.

You can’t directly see the internal representations of values. You can, however, use the `print` function to see a printed representation in the output window.

The printed representation of a number uses the familiar decimal representation (reading Roman Numerals is a fun challenge in museums, but thank goodness the Python interpreter doesn’t present the number 2014 as MMXIV in the output window). Thus, the printed representation of a number shown in the output window is the same as the literal that you specify in a program.

The printed representation of a character string, however, is not exactly the same as the literal used to specify the string in a program. For the literal in a program, you enclose the string in quotation marks. The printed representation, in the output window, omits the quotation marks.

Save & Run 4/17/2021, 9:08:38 AM - 2 of 2 Show in CodeLens

```
print(3.2)
print("Hello, World!")

```

```
3.2
Hello, World!
```

Activity: 1 -- ActiveCode (ac2_2_1)

Note

Literals appear in programs. The Python interpreter turns literals into **values**, which have internal representations that people never get to see directly. **Outputs** are external representations of values that appear in the output window. When we are being careful, we will use the terms this way. Sometimes, however, we will get a little sloppy and refer to literals or external representations as values.

Numbers with a decimal point belong to a class called **float**, because these numbers are represented in a format called *floating-point*. At this stage, you can treat the words *class* and *type* interchangeably. We’ll come back to a deeper understanding of what a class is in later chapters.

You will soon encounter other types of objects as well, such as lists and dictionaries. Each of these has its own special representation for specifying an object as a literal in a program, and for displaying an object when you print it. For example, list contents are enclosed in square brackets `[ ]`. You will also encounter some more complicated objects that do not have very nice printed representations: printing those won’t be very useful.

Check your understanding

data-2-1: What appears in the output window when the following statement executes?

```
print("Hello World!")
```

○ A. Nothing is printed. It generates a runtime error.

○ B. "Hello World!"

● C. Hello World!

Check me Compare me

✓ The printed representation omits the quote marks that are included in the string literal.

Activity: 2 -- Multiple Choice (question2_2_1)

You have attempted 3 of 2 activities on this page

✓ Completed. Well Done!

| Back to top