Data Types and Variables

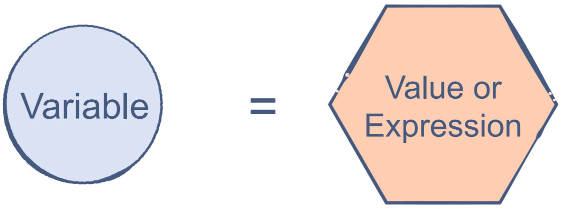
In this lesson, we will learn about data types and variables in Python.

WE'LL COVER THE FOLLOWING

Variables

> A **variable** is simply a name to which a value can be *assigned*.

The simplest way to assign a value to a variable is through the `=` operator. Variables allow us to store data so that we can use it later to perform operations in the code.



Data Types

Python provides three main data types:

- Numbers
- Strings
- Booleans

Let’s cover these in detail below:

Numbers

There are three main types of numbers in Python:

- Integers
- Floating Point Numbers
- Complex Numbers

Integers

The **integer** data type can hold positive and negative whole numbers.

```
num = 2780 # Assigning an integer to a variable
print (num)
num = -1625 # Assigning a new integer
print (num)
```

Floating point number

Floating point numbers, or floats, refer to positive and negative numbers with a fractional part.

```
print (2.02975745) # A positive float
print (-5.7289201) # A negative float

flt_pt = 0.758391739
print (flt_pt)
```

Complex numbers

Python also supports complex numbers. There are two ways to create a complex number:

1. `complex()` is used to create complex numbers. The first argument is the real part and the second argument is the imaginary part.
2. Simply write the value as `x+yj`, where `x` is the real part and `y` is the complex part and `j` represents the iota.

```
print(complex(10, 20))

complex_1 = complex(0, 2)
complex_2 = 2 + 3j
print(complex_1)
print(complex_2)
```

Extracting real and imaginary

- To extract the real part of a complex number, we use `.real`.

```
z.real
```

- To extract the imaginary part of a complex number, we use `.imag`.

```
z.imag
```

Let's look at an example of this below:

```
z  = complex(12, 5)
print('z =', z)

print("Real part of z is", z.real)
print("Imaginary part of z is", z.imag)
```

Booleans

The **Boolean** (also known as bool) data type allows us to choose between two values: true and false. In Python, we can simply use `True` or `False` to values: true and false. In Python, we can simply use `True` or `False` to represent a bool:

```
print (True)

f_bool = False
print (f_bool)
```

Booleans are often used in data comparisons.

Strings

A **string** is a collection of characters enclosed within single or double quotation marks.

```
print ("Edpresso") # String with double quotation marks

edu = 'Educative' # String with single quotation marks
print (edu)

empty = ""  # Empty string
print(empty)
```

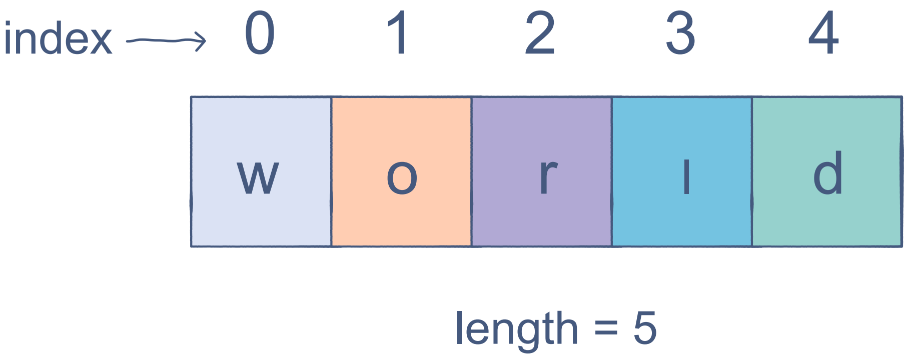
Length of a string

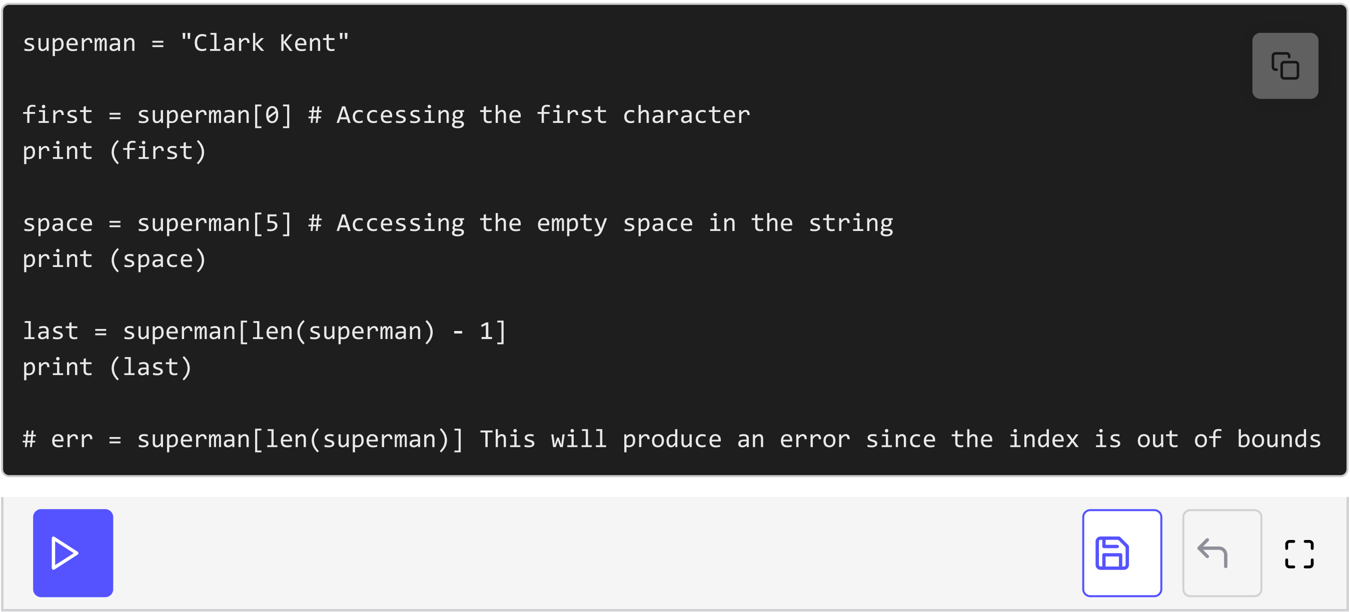
The length of a string can be found using the `len()` function.

```
str1 = "Educative" # string has 9 characters
print (len(str1))
```

Indexing

A string in Python is indexed from `0` to `n-1` where `n` is its length. This means that the index of the first character in a string is `0`. Each character in a string can be accessed using its index. The index must be closed within square brackets, `[]`.





```
superman = "Clark Kent"

first = superman[0] # Accessing the first character
print (first)

space = superman[5] # Accessing the empty space in the string
print (space)

last = superman[len(superman) - 1]
print (last)

# err = superman[len(superman)] This will produce an error since the index is out of bounds
```

We’ll learn about operators in the next lesson.