Chapter 1 - Basic Skills

Fundamentals

Variables

Learning Objectives - Variables

- **Define the rules when naming a variable**
- **Assign a value to a variable**
- **Define the four basic data types: strings, boolean, integers (ints), and floats**

Variables

Variable Names

Variables are used to store a value, and these values have a data type. Data types describe the kind of information that is being stored. Numbers are different than text, and integers are different from numbers with decimals. Variable declaration is when you create a variable and assign it a value. Enter the name of the variable you want to create, a `=` (called the assignment operator), and the value you want to store in the variable. You do not have to indicate the data type when declaring a variable. Use the `print` statement to see the value of the variable.

```
my_variable = "Hello world"
print(my_variable)
```

Do not use quotation marks when printing a variable. Using quotation marks will print the variable name, not its value.

```
my_variable = "Hello world"
print(my_variable)
print("my_variable")
```

Variable Naming Rules

Here are the rules for declaring a variable.

Rule	Correct	Incorrect
Start with a letter or underscore	`variable`, `_variable`	`1variable`
Remainder of variable name is letters, numbers, or underscores	`var_i_able`, `var1able`	`var-i-able`, `var!able`
Cannot use a Python keyword	`my_class`	`class`
Variables are case sensitive	`variable`, `Variable`, and `VARIABLE` are all different variables	

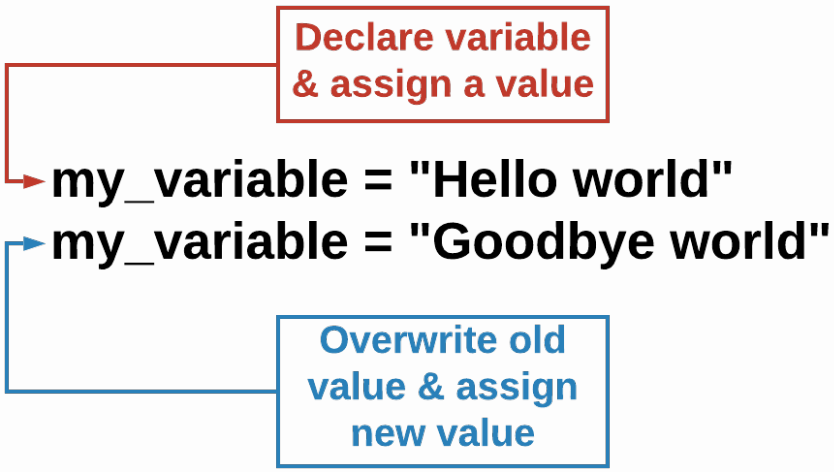
▼ **What are the Python key words?**

and	as	assert	break
class	continue	def	del
elif	else	except	FALSE
finally	for	from	global
if	import	in	is
lamda	None	nonlocal	nont
or	pass	raise	return
TRUE	try	while	with
yield			

Assigning Value

Assigning Value

The value stored in a variable can change. Use the assignment operator to give a variable a new value.



Overwriting Values

The image above is **not** declaring two variables called `my_variable`. The first line declares the variable because this is the first instance. The second line overwrites `Hello world` with `Goodbye world`. Enter the code below and see the results of the `print` commands. Use the code visualizer to see how the value of `my_variable` changes.

```
my_variable = "Hello world"
print(my_variable)
my_variable = "Goodbye world"
print(my_variable)
```

Code Visualizer

Data Types - Strings

Strings

A string is a collection of text, numbers, or symbols. Strings are always surrounded by quotation marks.

```
string_variable = "This is a string"
second_string = 'This is a string also'
print(string_variable)
print(second_string)
```

challenge

What happens if you:

- Mix single (') and double (") quotation marks?

  ▼

  What happened?

  This causes an error because Python requires that you be consistent with quotation marks. If you start with a single quote (') you must end with a single quote. The same is true for double quotes ("). You may use either style of quotation marks, just be consistent.

- Forget one of the quotation marks?

  ▼

  What happened?

  This causes an error because Python requires that quotation marks be used in pairs.

- Forget both quotation marks?

  ▼

  What happened?

  This causes an error because to Python a string without quotes appears to be a series of variables that have not been defined.

Notice that when you print a string, the quotation marks are not printed.

Data Types - Boolean

Boolean

Boolean values mean `True` or `False`. You will see how boolean values are used when we talk about conditionals and while loops.

```
boolean_variable = True
print(boolean_variable)
```

> challenge
>
> ## What happens if you:
>
> - Change the variable to `False`?
> - Change the variable to `true`?
> - Change the variable to `false`?

Data Types - Integers

Integers

Integers (often called ints) are whole numbers. They can be positive or negative. Do not use a comma when typing large numbers

▼ **5 vs. “5”**
`5` is not the same thing as `“5”`. The first one is an integer, the second is a string. You will see in a later lesson the different operations you can perform on strings and numbers. Treating a string as a number can cause errors.

```
integer_variable = 50
print(integer_variable)
```

challenge

What happens if you:

- Change the variable to `5000`?
- Change the variable to `5,000`?
- Change the variable to `050`?

Data Types - Floating Point Numbers

Floating Point numbers

Floating point numbers (often called floats) are numbers with a decimal. They can be positive or negative.

```
float_variable = 50.0
print(float_variable)
```

challenge

What happens if you:

- Change the variable to `50.`?
- Change the variable to `.001`?