**INTRODUCTION TO STRINGS**

**Introduction to Strings**

Words and sentences are fundamental to how we communicate, so it follows that we’d want our computers to be able to work with words and sentences as well.

In Python, the way we store something like a word, a sentence, or even a whole paragraph is as a [**string**](https://www.codecademy.com/resources/docs/python/strings?page_ref=catalog). A string is a sequence of characters contained within a pair of 'single quotes' or "double quotes". A string can be any length and can contain any letters, numbers, symbols, and spaces.

In this lesson, we will learn more about strings and how they are treated in Python. We will learn how to slice strings, select specific characters from strings, search strings for characters, iterate through strings, and use strings in conditional statements.

Let’s get started.

**Instructions**

**1.**

Save your favorite word as a string to the variable favorite\_word.

Checkpoint 2 Passed

Hint

Strings are written inside either single quotes ' ' or double quotes " ". Just be sure they match!

# Valid string using double quotes  
favorite\_word = "coding"  
   
# Also valid string using single quotes  
favorite\_word = 'coding'

**2.**

Print favorite\_word.

Checkpoint 3 Passed

Hint

It should look something like:

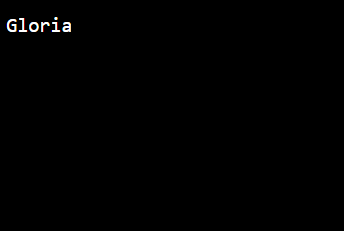
print(string\_name)

Notice how the quotes don’t appear in the output

**script.py**

favorite\_word = "Gloria"

print(favorite\_word)

****

**They're all Lists!**

A string can be thought of as a [**list**](https://www.codecademy.com/resources/docs/python/lists?page_ref=catalog) of characters.

Like any other list, each character in a string has an index. Consider the string:

favorite\_fruit = "blueberry"

We can select specific letters from this string using the *index*. Let’s look at the first letter of the string.

print(favorite\_fruit[1])  
# Output: l

Whoops, is that the first letter you expected? Notice that the letter at index 1 of "blueberry" isn’t b, it’s l. This is because the indices of a string start at 0. b is located at the zero index and we could select it using:

print(favorite\_fruit[0])  
# Output: b

It’s important to note that indices of strings must be integers. If we were to try to select a non-integer index we would get a TypeError.

For example:

print(favorite\_fruit[1.5])

This would result in:

Traceback (most recent call last):

File "script.py", line 3, in <module>

print(favorite\_fruit[1.5])

TypeError: string indices must be integers

**Instructions**

**1.**

One of the most common things that are represented by strings is names.

Save your name as a string to the variable my\_name.

Checkpoint 2 Passed

Hint

Remember, the string can include more than one word:

my\_name = "Python Person"

**2.**

Select the first letter of the variable my\_name and save it to first\_initial.

Checkpoint 3 Passed

Hint

The first character of a string is at index 0:

first\_initial = my\_name[0]

**script.py**

my\_name = "Andres"

first\_initial = my\_name[0]

**Cut Me a Slice of String**

Not only can we select a single character from a string, but we can also select entire chunks of characters from a string. We can do this with the following syntax:

string[first\_index:last\_index]

This is called *slicing* a string. When we slice a string we are creating a [substring](https://www.codecademy.com/resources/docs/python/substrings?page_ref=catalog) - a brand new string that starts at (and includes) the first\_index and ends at (but excludes) the last\_index.

Let’s look at some examples of this. Recall our favorite fruit:

favorite\_fruit = "blueberry"

The indices of this string are shown in the diagram below.

Let’s say we wanted a new string that contains the letters be. We could slice favorite\_fruit as follows:

print(favorite\_fruit[4:6])  
# Output: be

Notice how the character at the first index, b, is included, but the character at the last index, r, is excluded. If you look for the indices 4 and 6 in the diagram, you can see how the r is outside that range.

We can also have open-ended selections. If we remove the first index, the slice starts at the beginning of the string and if we remove the second index the slice continues to the end of the string.

print(favorite\_fruit[:4])  
# Output: blue  
   
print (favorite\_fruit[4:])  
# Output: berry

Again, notice how the b from berry is excluded from the first example and included in the second example.

**Instructions**

**1.**

You’re a programmer working for *Copeland’s Corporate Company*. At this company, each employee’s user name is generated by taking the first five letters of their last name.

A new employee, Rodrigo Villanueva, is starting today and you need to create his account. His first\_name and last\_name are stored as strings in **script.py**.

Create a variable new\_account by slicing the first five letters of his last\_name.

Checkpoint 2 Passed

Hint

It should look something like:

new\_account = last\_name[:5]

**2.**

Temporary passwords for new employees are also generated from their last names.

Create a variable called temp\_password by creating a slice out of the third through sixth letters of last\_name. Your string should have a total of 4 characters.

Checkpoint 3 Passed

Hint

Remember, because indices start at 0, to get the third through sixth characters of a string you would want to use

string\_name[2:6]

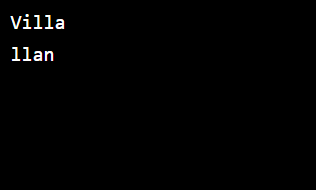
**script.py**

irst\_name = "Rodrigo"

last\_name = "Villanueva"

new\_account = last\_name[:5]

temp\_password = last\_name[2:6]

****

**Concatenating Strings**

We can also *concatenate*, or combine, two existing strings together into a new string. Consider the following two strings:

fruit\_prefix = "blue"  
fruit\_suffix = "berries"

We can create a new string by concatenating them together as follows:

favorite\_fruit = fruit\_prefix + fruit\_suffix  
   
print(favorite\_fruit)  
# Output: blueberries

Notice that there are no spaces added here. We have to manually add in the spaces when concatenating strings if we want to include them.

fruit\_sentence = "My favorite fruit is" + favorite\_fruit  
   
print(fruit\_sentence)  
# Output: My favorite fruit isblueberries  
   
fruit\_sentence = "My favorite fruit is " + favorite\_fruit  
   
print(fruit\_sentence)  
# Output: My favorite fruit is blueberries

It’s subtle, but notice that in the first example, there is no space between “is” and “blueberries”.

**Instructions**

**1.**

*Copeland’s Corporate Company* has realized that their policy of using the first five letters of an employee’s last name as a user name isn’t ideal when they have multiple employees with the same last name.

Write a function called account\_generator() that takes two inputs, first\_name and last\_name and concatenates the first three letters of each and then returns the new account name.

Checkpoint 2 Passed

Hint

Remember, a new function can be defined using:

def function\_name(parameter):  
  # Code goes here

**2.**

Test your function on the first\_name and last\_name provided in **script.py** and save it to the variable new\_account.

**script.py**

first\_name = "Julie"

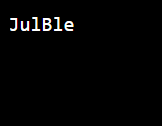
last\_name = "Blevins"

def account\_generator(first\_name, last\_name):

  return first\_name[:3] +  last\_name[:3]

new\_account = account\_generator("Julie", "Blevins")

print(new\_account)

****