



Lab 111: Lista, tupla, diccionario

Link: <https://awsrestart.instructure.com/courses/1632/modules/items/886822>

Información general sobre el laboratorio

En Python, los tipos de datos numéricos y de cadena se suelen utilizar en grupos denominados *colecciones*. La lista, la tupla y el diccionario son tres de estas colecciones compatibles con Python.

En este laboratorio, deberá realizar lo siguiente:

- utilizar el tipo de dato de lista
- utilizar el tipo de datos de “tupla” (tuple)
- utilizar el tipo de dato de diccionario ____

Ejercicio 1: Presentar el tipo de dato de lista

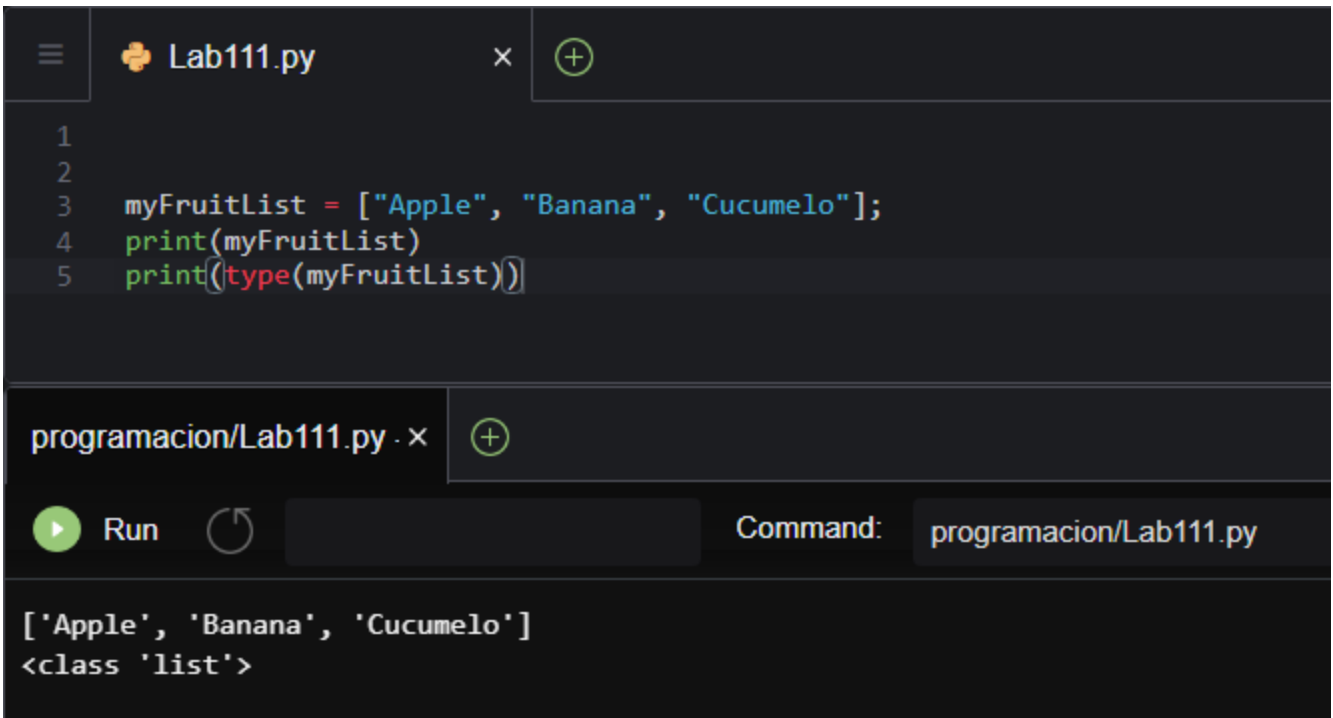
Definición de una lista

En esta actividad, editará un script en Python para almacenar una colección de nombres de frutas o una lista.

1. En el panel de navegación del IDE, elija el archivo **.py** que creó en la sección *Creación del archivo de ejercicios de Python* anterior.
2. En el archivo, escriba el siguiente código:

```
myFruitList = ["apple", "banana", "cherry"]
print(myFruitList)
print(type(myFruitList))
```

3. Guarde y ejecute el archivo.
4. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.



Acceso a una lista por posición

Puede acceder al contenido de una lista por su posición. En esta actividad, mostrará cada elemento de nuestra lista por su posición:

1. En los lenguajes de programación, el posicionamiento en una lista comienza en el cero (0). Los corchetes indican a Python qué posición en la lista desea. Para acceder a la cadena `apple`, escriba el siguiente código:

```
print(myFruitList[0])
```

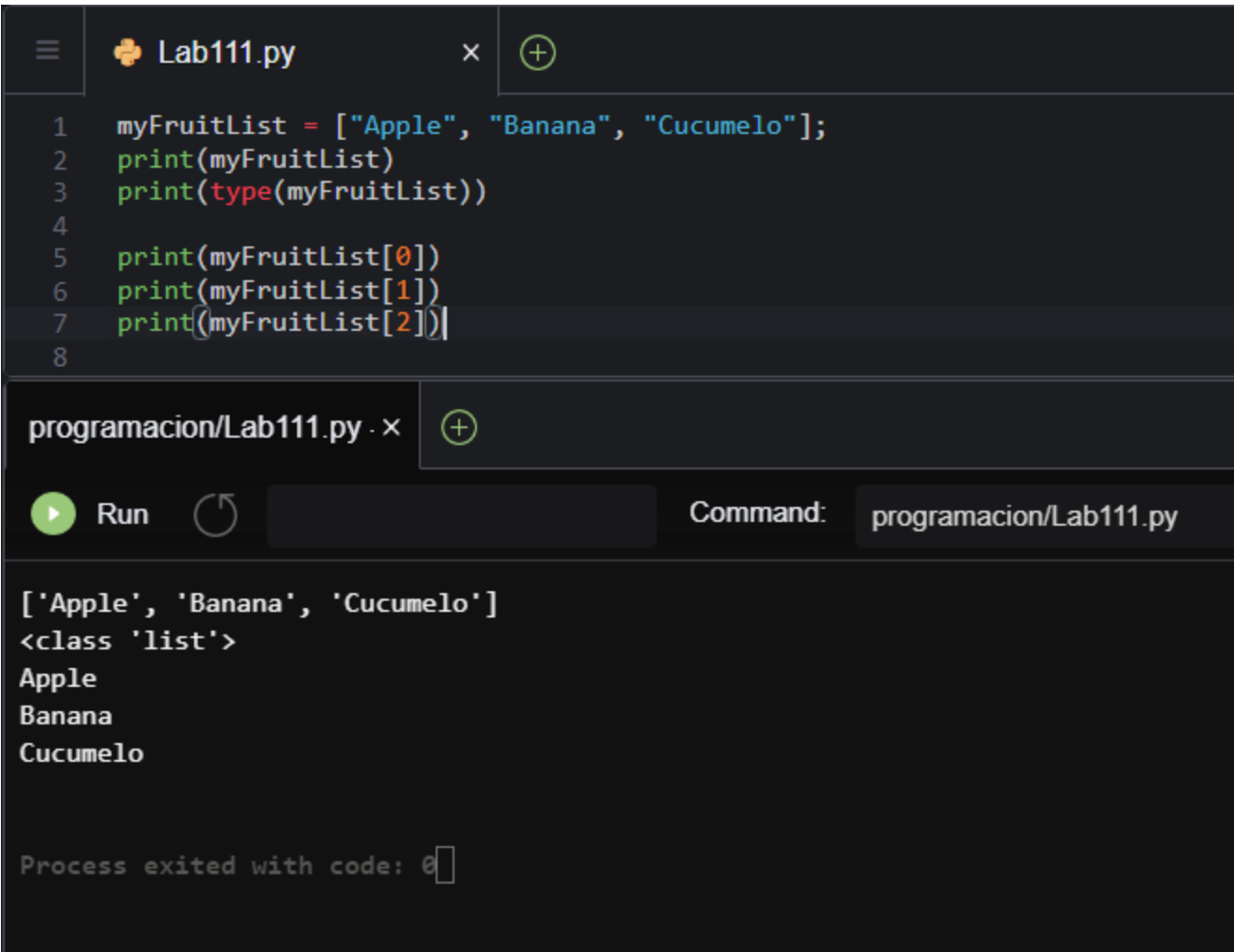
2. Para acceder a la cadena `banana`, escriba lo siguiente:

```
print(myFruitList[1])
```

3. Para acceder a la cadena `cherry`, escriba el siguiente código:

```
print(myFruitList[2])
```

4. Guarde y ejecute el archivo.
5. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.



The screenshot shows a code editor with a file named 'Lab111.py'. The code defines a list 'myFruitList' with elements 'Apple', 'Banana', and 'Cucumelo'. It prints the list, its type, and each element individually. Below the code, the output is displayed: the list ['Apple', 'Banana', 'Cucumelo'], its class '<class 'list'>', and the individual elements 'Apple', 'Banana', and 'Cucumelo'. At the bottom, it states 'Process exited with code: 0'.

```
1 myFruitList = ["Apple", "Banana", "Cucumelo"];
2 print(myFruitList)
3 print(type(myFruitList))
4
5 print(myFruitList[0])
6 print(myFruitList[1])
7 print(myFruitList[2])
8
```

programacion/Lab111.py · ×

Run Command: programacion/Lab111.py

```
['Apple', 'Banana', 'Cucumelo']
<class 'list'>
Apple
Banana
Cucumelo

Process exited with code: 0
```

Modificación de los valores de una lista

Los valores de una lista se pueden cambiar. En esta actividad, cambiará `cherry` por `orange`.

1. En Python, el posicionamiento en la lista comienza en cero (0), por lo que tiene que utilizar el número 2 para acceder a la tercera posición. Escriba el siguiente código:

```
myFruitList[2] = "orange"
```

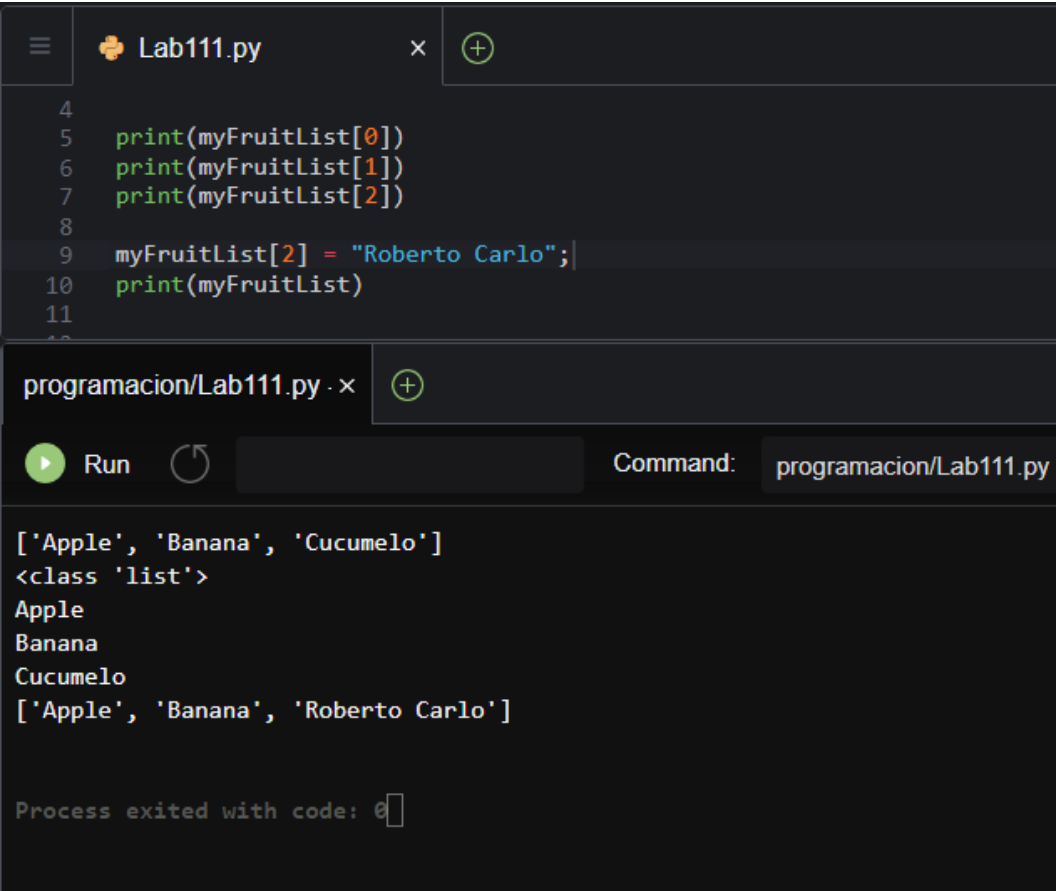
2. Muestre la lista actualizada:

```
print(myFruitList)
```

3. Guarde y ejecute el archivo.
4. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.

```
['apple', 'banana', 'cherry']
<class 'list'>
apple
banana
```

```
cherry
['apple', 'banana', 'orange']
```



The screenshot shows a code editor with a file named 'Lab111.py'. The code in the editor is as follows:

```
4
5 print(myFruitList[0])
6 print(myFruitList[1])
7 print(myFruitList[2])
8
9 myFruitList[2] = "Roberto Carlo";
10 print(myFruitList)
11
```

Below the code editor, there is a 'Run' button and a 'Command' field containing 'programacion/Lab111.py'. The output of the script is displayed in a terminal window below the command field:

```
['Apple', 'Banana', 'Cucumelo']
<class 'list'>
Apple
Banana
Cucumelo
['Apple', 'Banana', 'Roberto Carlo']

Process exited with code: 0
```

Ejercicio 2: Presentar el tipo de dato de tupla

Definición de una tupla

Una tupla es similar a una lista, pero no se puede cambiar. Un tipo de dato que no se puede cambiar después de su creación se conoce como *immutable*. Para definir una tupla, se utilizan paréntesis en lugar de corchetes ([]).

1. Cree una tupla escribiendo el siguiente código:

```
myFinalAnswerTuple = ("apple", "banana", "pineapple")
print(myFinalAnswerTuple)
print(type(myFinalAnswerTuple))
```

2. Guarde y ejecute el archivo.
3. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.

☰

🔥 Lab111.py

×

⊕

```
11 myFruitList[4] = roberto carlo ;
12 print(myFruitList)
13
14
15 # Tipo de dato Tupla
16
17 myFinalAnswerTuple = ("apple", "banano", "pineapple")
18 print(myFinalAnswerTuple)
19 print(type(myFinalAnswerTuple))
```

programacion/Lab111.py · ×

⊕

▶ Run

↺

Command: programacion/Lab111.py

```
['Apple', 'Banana', 'Cucumelo']
<class 'list'>
Apple
Banana
Cucumelo
['Apple', 'Banana', 'Roberto Carlo']
('apple', 'banano', 'pineapple')
<class 'tuple'>
```

Acceso a una tupla por posición

Al igual que con una lista, también se puede acceder a los elementos de una tupla por su posición:

1. Para acceder a la cadena `apple`, escriba el siguiente código:

```
print(myFinalAnswerTuple[0])
```

2. Para acceder a la cadena `banana`, escriba el siguiente código:

```
print(myFinalAnswerTuple[1])
```

3. Para acceder a la cadena `pineapple`, escriba el siguiente código:

```
print(myFinalAnswerTuple[2])
```

4. Guarde y ejecute el archivo.
5. Cerca de la parte superior de la ventana del IDE, elija el botón **Run** (Play) (Ejecutar [Reproducir]).
6. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto

```
['apple', 'banana', 'cherry']
<class 'list'>
apple
banana
cherry
['apple', 'banana', 'orange']
('apple', 'banana', 'pineapple')
<class 'tuple'>
apple
banana
pineapple
```

```
16
17 myFinalAnswerTuple = ("apple", "banano", "pineapple")
18 print(myFinalAnswerTuple)
19 print(type(myFinalAnswerTuple))
20 print("Posición 1:", myFinalAnswerTuple[0])
21 print("Posición 2:", myFinalAnswerTuple[1])
22 print("Posición 3:", myFinalAnswerTuple[2])
```

programacion/Lab111.py · ×

Run Command: programacion/Lab111.py

```
('apple', 'banano', 'pineapple')
<class 'tuple'>
Posición 1: apple
Posición 2: banano
Posición 3: pineapple
```

Ejercicio 3: Presentar el tipo de dato de diccionario

Definición de un diccionario

Un diccionario es una lista cuyas posiciones tienen nombres asignados (claves). Imagine que su lista muestra la fruta favorita de distintas personas.

1. Regrese al script en Python y escriba el siguiente código:

```
myFavoriteFruitDictionary = {
    "Akua" : "apple",
    "Saanvi" : "banana",
    "Paulo" : "pineapple"
}
```

2. Utilice la función `print()` para escribir el diccionario en el shell:

```
print(myFavoriteFruitDictionary)
```

3. Utilice la función `type()` para escribir el tipo de dato en el shell:

```
print(type(myFavoriteFruitDictionary))
```

4. Guarde y ejecute el archivo.
5. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.

Acceso al diccionario por nombre

En esta actividad, en lugar de utilizar números, recurrirá al nombre de las personas para acceder a su fruta favorita.

1. Para acceder a la fruta favorita de Akua, escriba el siguiente código:

```
print(myFavoriteFruitDictionary["Akua"])
```

2. Para acceder a la fruta favorita de Saanvi, escriba el siguiente código:

```
print(myFavoriteFruitDictionary["Saanvi"])
```

3. Para acceder a la fruta favorita de Paulo, escriba el siguiente código:

```
print(myFavoriteFruitDictionary["Paulo"])
```

- 4. Guarde y ejecute el archivo.
- 5. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.

```
['apple', 'banana', 'cherry']
<class 'list'>
apple
banana
cherry
['apple', 'banana', 'orange']
('apple', 'banana', 'pineapple')
<class 'tuple'>
apple
banana
pineapple
{'Akua': 'apple', 'Saanvi': 'banana', 'Paulo': 'pineapple'}
<class 'dict'>
apple
banana
pineapple
```

☰

📄 Lab111.py

×

⊕

```
29     "Paul" : "pineapple"
30 }
31
32 print(myFavoriteFruitDictionary)
33 print(type(myFavoriteFruitDictionary))
34
35 print(myFavoriteFruitDictionary["Pedro Pascal"])
36 print(myFavoriteFruitDictionary["Rick"])
37 print(myFavoriteFruitDictionary["Paul"])
38
```

programacion/Lab111.py · ×

⊕

▶ Run

↺

Command: programacion/Lab111.py

```
{'Pedro Pascal': 'apple', 'Rick': 'banana', 'Paul': 'pineapple'}
<class 'dict'>
apple
banana
pineapple
```