Educa2 – Clases y cursos de matemáticas y Programación

Curso Básico de Programación en Python

Preparador: Alexanyer Naranjo

Guía Teórica N°5

Estructuras de datos (Listas, Tuplas y Diccionario)

Listas:

Una lista es una secuencia ordenada de objetos de distintos tipos.

Se construyen poniendo los elementos entre corchetes [ ] separados por comas.

Se caracterizan por:

- Tienen orden.
- Pueden contener elementos de distintos tipos.
- Son mutables, es decir, pueden alterarse durante la ejecución de un programa.

```
L1 = [] #Lista vacía
L2 = [1,"dos",True] #Lista con elementos de distintos tipos
```

Creación de listas mediante la función list():

Otra forma de crear listas es mediante la función list().

- **list(c):** Crea una lista con los elementos de la secuencia o colección c.

Se pueden indicar los elementos separados por comas, mediante una cadena, o mediante una colección de elementos iterable.

```
L1 = list() #L1 = []
L2 = list(1,2,3) #L2 = [1,2,3]
```

Acceso a los elementos de una lista:

Se utilizan los mismos operadores de acceso que para cadenas de caracteres.

- **L[i]:** Devuelve el elemento de la lista L con el índice i.
- El índice del primer elemento de la lista es 0.

Sublistas:

- **L[i:j:k]:** Devuelve la sublista desde el elemento de L con el índice i hasta el elemento anterior al índice j, tomando elementos cada k.

Operaciones que no modifican una lista:

- **len(L):** Devuelve el número de elementos de la lista L.
- **min(L):** Devuelve el mínimo elemento de la lista L siempre que los datos sean comparables.
- **max(L):** Devuelve el máximo elemento de la lista L siempre que los datos sean comparables.
- **sum(L):** Devuelve la suma de los elementos de la lista L, siempre que los datos se puedan sumar.
- **dato in L:** Devuelve True si el dato pertenece a la lista L y False en caso contrario.
- **L.index(dato):** Devuelve la posición que ocupa en la lista L el primer elemento con valor dato.
- **L.count(dato):** Devuelve el número de veces que el valor dato está contenido en la lista L.
- **all(L):** Devuelve True si todos los elementos de la lista L son True y False en caso contrario.
- **any(L):** Devuelve True si algún elemento de la lista L es True y False en caso contrario.

Operaciones que modifican una lista:

- **L1 + L2:** Crea una nueva lista concatenan los elementos de las listas L1 y L2.
- **L.append(dato):** Añade dato al final de la lista L.
- **L.extend(secuencia):** Añade los datos de secuencia al final de la lista L.
- **L.insert(índice,dato):** Inserta dato en la posición índice de la lista L y desplaza los elementos una posición a partir de la posición índice.
- **L.remove(dato):** Elimina el primer elemento con valor dato en la lista L y desplaza los que están por detrás de él una posición hacia delante.
- **L.pop([índice]):** Devuelve el dato en la posición índice y lo elimina de la lista L, desplazando los elementos por detrás de él una posición hacia delante.
- **L.sort():** Ordena los elementos de la lista L de acuerdo al orden predefinido, siempre que los elementos sean comparables.
- **L.reverse():** invierte el orden de los elementos de la lista L.

Copia de listas:

Existen dos formas de copiar listas:

- **Copia por referencia, L1 = L2:** Asocia la variable L1 la misma lista que tiene asociada la variable L2, es decir, ambas variables apuntan a la misma dirección de memoria. Cualquier cambio que hagamos a través de L1 o L2 afectará a la misma lista.
- **Copia por valor, L1 = list(L2):** Crea una copia de la lista asociada a L2 en una dirección de memoria diferente y se la asocia a L1. Las variables apuntan a direcciones de memoria diferentes que contienen los mismos datos. Cualquier cambio que hagamos a través de L1 no afectará a la lista de L2 y viceversa.

Tuplas:

Una tupla es una secuencia ordenada de objetos de distintos tipos.

Se construyen poniendo los elementos entre corchetes ( ) separados por comas.

Se caracterizan por:

- Tienen orden.
- Pueden contener elementos de distintos tipos.

Son inmutables, es decir, no pueden alterarse durante la ejecución de un programa.

Se usan habitualmente para representar colecciones de datos una determinada estructura semántica, como por ejemplo un vector o una matriz.

```
T1 = (1,"Dos","Tres",True) #Tupla con elementos de distintos tipos
```

Creación de tuplas mediante la función tuple():

Otra forma de crear tuplas es mediante la función tuple().

- **tuple(c):** Crea una tupla con los elementos de la secuencia o colección c.

Se pueden indicar los elementos separados por comas, mediante una cadena, o mediante una colección de elementos iterable.

```
T1 = tuple() #T1 = ()
T2 = tuple(1,2,3) #T2 = (1,2,3)
```

Operaciones con tuplas:

El acceso a los elementos de una tupla se realiza del mismo modo que en las listas. También se pueden obtener subtuplas de la misma manera que las sublistas.

Las operaciones de listas que no modifican la lista también son aplicables a las tuplas.

Diccionarios:

Un diccionario es una colección de pares formados por una clave y un valor asociado a la clave.

Se construyen poniendo los pares entre llaves { } separados por comas, y separando la clave del valor con dos puntos :.

Se caracterizan por:

- No tienen orden.
- Pueden contener elementos de distintos tipos.
- Son mutables, es decir, pueden alterarse durante la ejecución de un programa.
- Las claves son únicas, es decir, no pueden repetirse en un mismo diccionario, y pueden ser de cualquier tipo de datos inmutable.

```
D = {
    'nombre': 'Educa2',
    'despacho': 1012,
    'email': 'educa2ucv@gmail.com'
}
```

Acceso a los elementos de un diccionario:

- **D[clave]** devuelve el valor del diccionario D asociado a la clave. Si en el diccionario no existe esa clave devuelve un error.
- **D.get(clave,valor)** devuelve el valor del diccionario D asociado a la clave. Si en el diccionario no existe esa clave devuelve valor, y si no se especifica un valor por defecto devuelve **None**.

```
D = {
    'nombre': 'Educa2',
    'despacho': 1012,
    'email': 'educa2ucv@gmail.com'
}

print(D['nombre'])
print(D.get('email'))
print(D.get('curso','Python'))
```

Operaciones que no modifican un diccionario:

- **len(D):** Devuelve el número de elementos del diccionario D.
- **min(D):** Devuelve la mínima clave del diccionario D siempre que las claves sean comparables.
- **max(D):** Devuelve la máxima clave del diccionario D siempre que las claves sean comparables.
- **sum(D):** Devuelve la suma de las claves del diccionario D, siempre que las claves se puedan sumar.
- **clave in D:** Devuelve True si la clave pertenece al diccionario D y False en caso contrario.
- **D.keys():** Devuelve un iterador sobre las claves de un diccionario.
- **D.values():** Devuelve un iterador sobre los valores de un diccionario.
- **D.items():** Devuelve un iterador sobre los pares clave-valor de un diccionario.

Operaciones que modifican un diccionario:

- **D[clave] = valor:** Añade al diccionario D el par formado por la clave clave y el valor.
- **D.update(D2):** Añade los pares del diccionario D2 al diccionario D.
- **D.pop(clave,alternativo):** Devuelve del valor asociado a la clave del diccionario d y lo elimina del diccionario. Si la clave no está devuelve el valor alternativo.
- **D.popitem():** Devuelve la tupla formada por la clave y el valor del último par añadido al diccionario D y lo elimina del diccionario.
- **del D[clave]:** Elimina del diccionario D el par con la clave.
- **D.clear():** Elimina todos los pares del diccionario D de manera que se queda vacío.

Copia de diccionarios:

Existen dos formas de copiar diccionarios:

- **Copia por referencia, D1 = D2:** Asocia la variable D1 el mismo diccionario que tiene asociado la variable D2, es decir, ambas variables apuntan a la misma dirección de memoria. Cualquier cambio que hagamos a través de D1 o D2 afectará al mismo diccionario.
- **Copia por valor, D1 = dict(D2):** Crea una copia del diccionario asociado a D2 en una dirección de memoria diferente y se la asocia a D1. Las variables apuntan a direcciones de memoria diferentes que contienen los mismos datos. Cualquier cambio que hagamos a través de D1 no afectará al diccionario de D2 y viceversa.