



Python: Colecciones de Datos

Centro de Servicios y Gestión Empresarial
SENA Regional Antioquia

Conceptualización

Colecciones de Datos

En Python, las colecciones de datos son estructuras que permiten almacenar múltiples valores en una sola variable. Python ofrece cuatro tipos principales de colecciones:

Listas (list)

↓
Colección ordenada y modificable.

Tuplas (tuple)

↓
Colección ordenada e inmutable.

Conjuntos (set)

↓
Colección no ordenada y sin duplicados.

Diccionarios (dict)

↓
Colección de pares clave: valor.

Colecciones de Datos

En Python, el estilo recomendado para nombrar las colecciones es la convención **snake_case** (minúsculas con guion bajo) según la **Guía de Estilo PEP 8**. Se recomienda plural cuando la colección contiene varios elementos.

💡 Reglas Generales

- ◊ **Listas, tuplas y conjuntos** → Plural (usuarios, productos, nombres).
- ◊ **Diccionarios** → Singular (porque representa un solo objeto con varios atributos) (usuario, producto).

Listas

Listas

Las listas son ordenadas, permiten elementos duplicados y son mutables (pueden modificarse).

💡 Sintaxis

```
# Definir una lista

# Lista de Strings
frutas = ["manzana", "banana", "cereza", "banana"]

# Lista de enteros
numeros = [10, 20, 30, 40, 50]
```

Listas

❖ Métodos Principales

Método	Descripción	Ejemplo
<code>append(x)</code>	Agrega un elemento al final	<code>frutas.append("uva")</code>
<code>insert(i, x)</code>	Inserta x en la posición i	<code>frutas.insert(1, "mango")</code>
<code>remove(x)</code>	Elimina el primer elemento con valor x	<code>frutas.remove("banana")</code>
<code>pop(i)</code>	Elimina el elemento en la posición i	<code>frutas.pop(1)</code>
<code>sort()</code>	Ordena la lista	<code>frutas.sort()</code>
<code>reverse()</code>	Invierte el orden de la lista	<code>frutas.reverse()</code>
<code>count(x)</code>	Cuenta las veces que x aparece en la lista	<code>frutas.count("banana")</code>
<code>len(x)</code>	Contar la cantidad de elementos de la lista	<code>len(frutas)</code>
<code>[x:x]</code>	Rango de índices especificando	<code>frutas[2:4]</code>

Listas

🔨 Ejemplo:

```
# Definir la Lista de Frutas
frutas = ["manzana", "banana", "cereza"]

# Agregar elementos a la lista
frutas.append("uva") # Agregar uva

# Eliminar elementos de la lista
frutas.remove("banana") # Eliminar banana

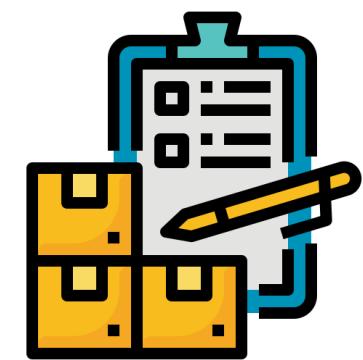
# Imprimir la lista
print(frutas) # ['manzana', 'cereza', 'uva']
```

Listas

💡 Ejercicio: Gestión de Inventario

Una tienda necesita gestionar su inventario de productos. La aplicación debe permitir al usuario gestionar las siguientes acciones:

- Ver la lista de productos disponibles: "*Laptop*", "*Mouse*", "*Teclado*", "*Monitor*"
- Agregar un nuevo producto.
- Eliminar un producto.
- Ordenar la lista de productos alfabéticamente.
- Salir del programa.



Listas

💡 Ejercicio: Gestión de Inventario

Listar elementos

```
for producto in productos:  
    print(f"- {producto}")
```

Validar elementos

```
nuevo_producto = input("Ingrese el nombre del producto a agregar:  
").capitalize()  
# Verificar si el producto ya está en la lista  
if nuevo_producto in productos:  
    print("El producto ya está en la lista.")  
else:  
    print("El producto no está en la lista.")
```

Listas

💡 Ejercicio: Gestión de Inventario

Validar y Agregar elementos

```
nuevo_producto = input("Ingrese el nombre del producto a
agregar: ").capitalize()
# Verificar si el producto ya está en la lista
if nuevo_producto in productos:
    print("El producto ya está en la lista.")
else:
    # Agregar el producto a la lista
    productos.append(nuevo_producto)
    print(f'{nuevo_producto} ha sido agregado al
inventario.')
```

Listas

📝 Ejercicio: Gestión de Inventario

Eliminar elementos

```
# Solicitar el nombre del producto
eliminar_producto = input("Ingrese el nombre del producto a eliminar:")
                            .capitalize()
# Verificar si el producto está en la lista
if eliminar_producto in productos:
    # Eliminar el producto de la lista
    productos.remove(eliminar_producto)
    print(f'{eliminar_producto} ha sido eliminado del inventario.')
else:
    print("El producto no está en la lista.")
```

Listas

📝 **Ejercicio:** Gestión de Inventario

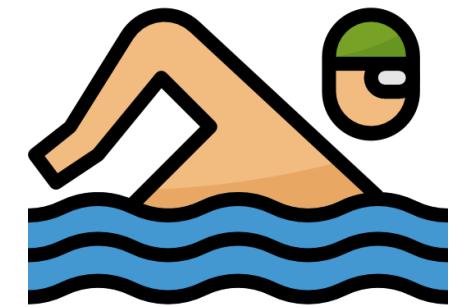
Ordenar elementos

```
productos.sort()  
print("Los productos han sido ordenados alfabéticamente.",  
      productos)
```

Listas

❖ **Ejercicio:** Natación

- Realice una aplicación que permita calcular:
 - El valor promedio, valor mínimo y valor máximo de un número indefinido de marcas (tiempos) de los nadadores olímpicos en la categoría de 50m pecho.
- La información debe ser almacenada en una lista
- Para salir del sistema utilice el valor de cero.



Listas

Método	Descripción	Ejemplo	Salida
<code>clear()</code>	Elimina todos los elementos de la lista.	<code>numeros = [1, 2, 3]; numeros.clear()</code>	[]
<code>copy()</code>	Devuelve una copia de la lista.	<code>copia = numeros.copy()</code>	[1, 2, 3]
<code>extend(iterable)</code>	Agrega elementos de otra lista o iterable al final de la lista actual.	<code>numeros.extend([4, 5])</code>	[1, 2, 3, 4, 5]
<code>zip(lista1, lista2)</code>	Combina dos listas en pares (tuplas).	<code>list(zip([1, 2, 3], ['a', 'b', 'c']))</code>	[(1, 'a'), (2, 'b'), (3, 'c')]
<code>index(valor)</code>	Devuelve el índice del primer elemento con el valor especificado.	<code>numeros.index(2)</code>	1
<code>max(lista)</code>	Devuelve el valor más grande de la lista.	<code>max([4, 7, 2, 9])</code>	9
<code>min(lista)</code>	Devuelve el valor más pequeño de la lista.	<code>min([4, 7, 2, 9])</code>	2
<code>sum(lista)</code>	Devuelve la suma de todos los elementos de la lista.	<code>sum([1, 2, 3, 4])</code>	10
<code>list(tupla)</code>	Convierte una tupla en una lista.	<code>list((10, 20, 30))</code>	[10, 20, 30]



G R A C I A S

Presentó: Alvaro Pérez Niño

Instructor Técnico

Correo: aperezn@sena.edu.co

<http://centrodesserviciosygestionempresarial.blogspot.com/>

Línea de atención al ciudadano: 01 8000 910270

Línea de atención al empresario: 01 8000 910682



www.sena.edu.co