Introducción

¡Hola, volvimos! Seguro que están familiarizados con los tipos de datos básicos que hemos trabajado en capítulos anteriores, como los números y las cadenas. ¿Pero qué pasa si quieres almacenar un montón de números o cadenas? ¿Vas a declarar una variable para cada uno de ellos? ¡Por supuesto que no! Ahí es donde entran los arreglos. ¿Y qué son los arreglos? Bueno, estoy aquí para contarte todo sobre ellos.

Imagínate que eres el dueño de una tienda y tienes varios productos. Necesitas una forma de llevar un registro de los productos que tienes en stock. Podrías tener una variable para cada producto, pero eso rápidamente se vuelve inmanejable. En lugar de eso, puedes usar un arreglo para guardar los nombres de los productos en un solo lugar. ¡Vamos a explorar cómo hacerlo!

¿Qué es un arreglo unidimensional?

En términos muy sencillos, un arreglo es una colección de elementos que son *del mismo tipo de datos*. Piensa en él como una estantería donde cada estante tiene un valor que tú decides.

```
productos = ["manzanas", "bananas", "cerezas", "duraznos", "peras"]
```

Aquí, productos es un arreglo (en Python, lo llamamos "lista") que contiene varias cadenas.

Nota: Los clásicos arrays descritos en lenguajes como C, Java, entre otros no existen en Python como tal. Una forma elegante de superar esta limitación es utilizar las llamadas listas in Python, que tienen una función similar. A diferencia de los contenedores de Java, las listas de Python pueden contener diferentes tipos de valores.

Acceder a un elemento específico

Para acceder a los elementos de un arreglo, usamos su índice, que es la posición del elemento en el arreglo. Recuerda que en Python, los índices comienzan en 0.

```
primer_producto = productos[0]
print(primer_producto)
```

Añadir elementos a un array en Python

En Python, puedes añadir elementos a una lista usando el método append(). Este método añade el elemento al final de la lista.

```
productos.append("kiwis")
print(productos)
```

Eliminar elementos del array con pop ( ) o remove ( )

Para eliminar elementos de una lista, puedes usar el método pop() o remove(). pop() elimina el elemento en la posición especificada, mientras que remove() elimina el primer elemento con el valor especificado.

```
productos.pop(1)  # Elimina el elemento en el índice 1
productos.remove("cerezas")  # Elimina la primera aparición de
"cerezas"
print(productos)
```

Lista de diferentes métodos para arrays en Python

Python ofrece muchos métodos útiles para trabajar con listas. Aquí hay una tabla con algunos de los más comunes:

Método	Descripción
`append()`	Añade un elemento al final de la lista.
`clear()`	Este método elimina todos los elementos de la lista.
`copy()`	`copy()` produce una copia de toda la lista.
`count()`	Este método da como resultado el número exacto de elementos con un valor determinado.
`extend()`	`extend()` añade todos los elementos de una lista al final de una lista en Python.
`index()`	Muestra el número de índice del primer elemento con un valor determinado.
`insert()`	Añade un elemento en una posición determinada.
`len()`	Con `len()` se determina la longitud de un array en Python.
`pop()`	Con `pop()` se elimina un elemento en una posición determinada.
`remove()`	`remove()` elimina el primer elemento con un

Método	Descripción
	valor determinado.
`reverse()`	Utiliza este método para invertir el orden de los elementos de tu lista en Python.
`sort()`	Con `sort()` puedes ordenar tu lista.

Desafío 1: Gestión del stock de la verdulería

¡Bienvenido al desafío de la verdulería! Has sido nombrado el nuevo gerente de una tienda de frutas, verduras y hortalizas. Tu tarea será manejar el stock de la tienda utilizando un arreglo. Aquí hay una serie de tareas que tendrás que completar.

Inventario inicial

Aquí está el inventario inicial de la tienda:

```
inventario = ["manzanas", "bananas", "zanahorias", "espinacas", "brocoli", "cebolla", "kiwis"]
```

Pregunta 1: ¿Cuántos tipos de productos hay en el inventario inicial?

Pregunta 2: ¿Qué producto está en la tercera posición del inventario?

Gestión del inventario

Un cliente viene y compra todas las bananas.

Pregunta 3: ¿Cómo actualizarías el inventario después de la venta?

Ahora recibes un envío de nuevos productos: "frutillas", "apio" y "papas".

Pregunta 4: ¿Cómo añadirías estos productos al inventario?

Pregunta 5: ¿Cómo verificas si las "papas" están ahora en el inventario?

Un agricultor viene con una rara fruta llamada "dragonfruit", pero sólo tienes espacio para 7 productos en tu tienda.

Pregunta 6: ¿Cómo decidirías qué producto sacar para hacer espacio para el "dragonfruit"?

Ordenando el inventario

Decides que sería más fácil gestionar el inventario si estuviera ordenado alfabéticamente.

Pregunta 7: ¿Cómo ordenarías el inventario?

Un nuevo empleado viene y necesita una copia del inventario para poder reponer los estantes.

Pregunta 8: ¿Cómo proporcionarías una copia del inventario al nuevo empleado, asegurándote de que si el empleado hace cambios en su copia, el inventario original no se vea afectado?

Esperamos que disfrutes de este desafío y que te ayude a entender mejor cómo trabajar con arreglos en Python. ¡Buena suerte, gerente de la tienda!

Desafío 2: Ordenar el inventario de libros

Como encargado de la biblioteca, necesitas organizar los libros de acuerdo con sus códigos de identificación en orden decreciente, sin modificar la lista original. Se recomienda usar la función sorted(). ¿Por qué es importante no modificar la lista original? ¿Por qué no puedo usar el método sort sobre la lista original?

Desafío 3: Caracteres ASCII

Eres un desarrollador creando una herramienta educativa que muestra los caracteres ASCII correspondientes a una lista de números. Se recomienda usar la función chr() para convertir los números en caracteres.

Desafío 4: Control de notas de estudiantes

Como profesor, necesitas manejar las notas de tus estudiantes. Permite ingresar todas las notas de los estudiantes y realiza varias operaciones con esos datos.

Pregunta 1: Calcula el promedio de las notas de la clase. ¿Cómo lo harías?

Pregunta 2: Encuentra la nota más baja y la más alta. ¿Cómo lo harías?

Pregunta 3: Identifica la nota que más se repite. ¿Cómo lo harías?

Nota: La función Counter es útil porque simplifica el conteo de elementos en una lista, permitiendo identificar rápidamente la frecuencia de cada nota.

Pregunta 4 (Plus): Realiza un gráfico de barras con las notas. ¿Cómo lo harías?

Nota: La biblioteca matplotlib.pyplot es útil porque proporciona funciones fáciles de usar para crear visualizaciones de datos, como gráficos de barras, que pueden ayudar a visualizar la distribución de las notas de manera clara y comprensible.

```
#Calcular el promedio de las notas de la clase.
#Realizar una lista de las calificaciones.
notas= [7,6,9,7,8,9,8,10,8,6,10]
#Realizar los cálculos, sumando las calificaciones y dividiendo entre
la cantidad de estudiantes.
promedio= sum(notas) / len (notas)
```

```
#Imprimir el promedio con el resultado oobtenido.
print("El promedio de la clase es", promedio)
```

```
El promedio de la clase es 8.0
```

Pregunta 1: Calcula el promedio de las notas de la clase. ¿Cómo lo harías? Primero realicé la lista de notas, definida por el uso de corchetes [], para calcular el promedio utilicé la suma (sum) para que se agrupe todas las calificaciones y luego la división entre las cantidades. Al imprimir el promedio se obtuvo que el mismo es de 8.

```
#Encuentra la nota más baja  y más alta, ¿cómo lo harías?
#Realizar la lista de las notas []
notas=[7,6,9,7,8,9,8,10,6,10]
#Encontrar la nota mínima
nota_mínima=min(notas)
#Encontrar la nota máxima
nota_máxima=max(notas)
#Imprimir los resultados
print("la nota más baja", nota_mínima)
print("la nota más alta", nota_máxima)
```

```
la nota más baja 6
la nota más alta 10
```

Pregunta 2: Al igual que la anterior, realicé la lista de notas. Luego solicité encontrar las notas usando las funciones (min) y (max) para expresar la mínima y máxima. Al imprimir los resultados se obtuvo que la mínima es 6 y la máxima 10.

Problema 1: Análisis de ventas mensuales usando el módulo array

Eres el gerente de una tienda de ropa y necesitas analizar las ventas mensuales de un producto específico durante el último año. Usarás el módulo array para almacenar y manipular los datos de ventas de cada mes. El objetivo es calcular la venta total anual, el promedio de ventas mensuales, y encontrar los meses con la venta más baja y más alta.

Datos Iniciales

Las ventas mensuales del producto específico son las siguientes (en unidades):

```
import array

ventas_mensuales = array.array('i', [120, 135, 150, 145, 160, 155,
130, 140, 125, 170, 160, 180])
```

Tareas:

Calcula la venta total anual.

```
total_anual = sum(ventas_mensuales)
print(f"Venta total anual: {total_anual} unidades")
```

Calcula el promedio de ventas mensuales.

```
promedio_mensual = total_anual / len(ventas_mensuales)
print(f"Promedio de ventas mensuales: {promedio_mensual:.2f}
unidades")
```

Encuentra el mes con la venta más baja y más alta.

```
mes_baja = ventas_mensuales.index(min(ventas_mensuales)) + 1
mes_alta = ventas_mensuales.index(max(ventas_mensuales)) + 1
print(f"Mes con la venta más baja: {mes_baja} (Venta:
{min(ventas_mensuales)} unidades)")
print(f"Mes con la venta más alta: {mes_alta} (Venta:
{max(ventas_mensuales)} unidades)")
```

Problema 2: Análisis de temperaturas diarias usando NumPy

Como meteorólogo, necesitas analizar las temperaturas diarias registradas en tu ciudad durante el último mes. Usarás la librería NumPy para trabajar eficientemente con los datos de temperaturas. El objetivo es calcular la temperatura media, la temperatura máxima y mínima, y visualizar las temperaturas en un gráfico de línea.

Datos Iniciales

Las temperaturas diarias registradas en el último mes (30 días) son las siguientes:

```
import numpy as np

temperaturas_diarias = np.array([22.5, 21.3, 23.1, 25.0, 24.5, 22.1,
23.7, 24.8, 25.3, 26.1,
                                 23.4, 24.0, 22.9, 21.5, 23.0, 24.3,
25.0, 24.7, 23.1, 22.4,
                                 23.5, 24.1, 25.4, 26.0, 24.8, 23.9,
22.7, 23.3, 24.5, 25.1])
```

Tareas:

Calcula la temperatura media del mes.

```
temperatura_media = np.mean(temperaturas_diarias)
print(f"Temperatura media del mes: {temperatura_media:.2f}°C")
```

Encuentra la temperatura máxima y mínima del mes.

```
temperatura_maxima = np.max(temperaturas_diarias)
temperatura_minima = np.min(temperaturas_diarias)
print(f"Temperatura máxima del mes: {temperatura_maxima}°C")
print(f"Temperatura mínima del mes: {temperatura_minima}°C")
```

Genera un gráfico de línea de las temperaturas diarias.

```
import matplotlib.pyplot as plt

dias = np.arange(1, 31)
plt.plot(dias, temperaturas_diarias, marker='o')
plt.xlabel('Día')
plt.ylabel('Temperatura (°C)')
plt.title('Temperaturas Diarias del Último Mes')
plt.grid(True)
plt.show()
```