TALENTOFUTURO

Docentes:
Renzo Stanley
Pablo Cruz

# Clase 2: Estructuras de Datos Avanzadas

*Apoyado por:*
CORFO

Módulo 2: Programación Avanzada en Python.

# Clase de hoy

01

## Listas

Colecciones de elementos

02

## Diccionarios

Gestión de pares clave-valor

# 2A Listas

## Bloque A

# Qué veremos en Bloque A

- Listas.
- Operaciones sobre listas.
- Iteración sobre listas.
- Otras operaciones genéricas aplicables a listas.

# Listas

- Las **listas** en Python son un tipo de estructuras de datos que permiten almacenar múltiples ítems en una sola variable.

- Algunas de las ventajas son que estas permiten almacenar **distintos tipos** (prácticamente todos los permitidos por Python) y son **mutables**, es decir, su contenido puede cambiar.

- Las listas son **iterables**, es decir, podemos usar un ciclo como "`for`" para iterar sobre estas.

- Además, las listas son objetos que proveen una serie de operaciones que extienden la funcionalidad sobre estas.

# Listas

- Una lista en Python se define por medio de los corchetes cuadrados [ ] con cada elemento separado por coma.

- Cada elemento de una lista se determina según su posición, partiendo en **cero**. Por ejemplo, para acceder al primer elemento de la lista se escribe `lista[0]`.

- La lista vacía es muy importante ya que esta permite crear un elemento iterable sobre el cual se puede incorporar datos.

- Operaciones típicas como agregar elemento, ordenar elementos, borrar elementos, buscar elemento, son soportados por métodos nativos de esta estructura de datos.

# Listas

- Definición de una lista:

```
productos = [“zapatillas”, “chaquetas”, “camisas”]
```

- Agregar un elemento:

```
productos.append(“pantalones”)
```

- Agregar un elemento en posición específica:

```
productos.insert(posición, elemento)
```

- Eliminar el primer elemento que coincide con el criterio:

```
productos.remove(elemento_a_eliminar)
```

# Listas

- Ordenar la lista:

```
productos.sort()
```

- Invertir el orden de la lista:

```
productos.reverse()
```

- Eliminar un elemento en posición específica y retornarlo:

```
productos.pop(posición)
```

# Recorridos en listas

- Las listas son **iterables**, es decir, existe un iterador que puede ser usado por un ciclo para sistematizar el recorrido de los elementos.

- El ciclo que permite recorrer directamente las listas con el iterador se llama "**`for`**".

```
for elemento in lista:
        print(elemento)
```

- Recordemos que las **listas pueden contener otras listas como elementos**, y el ciclo for sólo itera en la lista en un solo nivel.

# Listas

## Listas y operaciones genéricas

- Existen otras funciones nativas en Python que actúan sobre las listas:
  - `min(lista)` retorna el mínimo de la lista.
  - `max(lista)` retorna el máximo de la lista.
  - `sum(lista)` retorna la suma de los elementos de la lista.
  - `len(lista)` retorna el largo de la lista (en un solo nivel).

- Estas funciones son especialmente útiles cuando deseamos, por ejemplo, calcular un promedio o determinar el rango en un conjunto de números.

# Ejemplo bloque A

**Ejemplo: Gestión de Inventario de Productos**

Considera un sistema de inventario simple para una tienda. Los productos están organizados en una lista, y el sistema debe ser capaz de realizar varias operaciones para gestionar el inventario.

1. Cree una lista inicial de productos fija.
2. Solicita al usuario que agregue tres nuevos productos.
3. Elimina un producto específico.
4. Ordena la lista en orden alfabético.
5. Muestra el primer y último producto en orden alfabético.

# Trabajo grupal – Bloque A

# Paralelo 1

| G1 | G2 | G3 | G4 |
|---|---|---|---|
| Mónica Carvajal Romero | Rodrigo Yanez Pilgrim | Ivan Santivañez | John Gonzalez |
| Patricio Hernández Pavez | Pablo Mahuzier | Francisco Zuloaga | Rodrigo Espinoza |
| Paula Rivera Oberg | Jhonatan Ortiz | Hamid Pinilla Saah | Pablo Silva |
| Valentina Loyola Rodó | Ariel Mora | Freddy Vega | Marlenne Allendes |
| Nicolas Gonzalez | María Ignacia Strobel | Hilda Silva | Jorge San Martin |
| Nicolás Aravena | christian donoso | Teresa Peña | Carlos Gonzalez |

| G5 | G6 | G7 | G8 |
|---|---|---|---|
| Paulina Valenzuela | Gabriela Torres | Miguel Mayorga | Felipe Mercado Lopez |
| Daniela Castro | Matias Araya | Isaac Vega | Francisco Yañez |
| Jorge Moraga Calvo | Alex Riquelme | Francisco Villaseca | Matias Faundes |
| Alejandro Gonzalez | Jorge Troncoso | Matías Macaya | christopher nuñez soto |
| Francisco Vega Contreras | Valentina Gutierrez | Gabriela Karina Lopez | Angelo Farias |
| Alvaro Torres | Jorge Estefane | Rodrigo Fuenzalida Vera | yuri urzua lebuy |

# Trabajo grupal – Ejercicio #1

## 1. Cálculo de promedio y rango

a. Crear una lista con los siguientes números:
`10, 20, 5, 60, 20, 10` (en este orden).

b. Determine el rango del conjunto de números.

c. Determine el promedio haciendo uso del ciclo `for`.

d. Determine el promedio haciendo uso de las funciones genéricas que aplican sobre la lista.

# Trabajo grupal – Ejercicio #2

## 2. Estudiantes y sus notas

Desarrolle un programa que permita almacenar y calcular el promedio de notas de cinco estudiantes en una escuela. Cada estudiante tiene tres calificaciones en un rango de 1 a 7. El sistema debe solicitar los datos de los estudiantes y sus notas al usuario y luego mostrar el promedio de calificaciones de cada estudiante.

a. Crear un programa que solicite el nombre de cada uno de los cinco estudiantes. Para cada estudiante, solicite tres notas.
b. Almacenar los datos de cada estudiante (nombre y lista de notas) en una lista de listas.
c. Calcular el promedio de notas para cada estudiante. Mostrar el nombre de cada estudiante junto con su promedio.

# Break!

# 2B Diccionarios

## Bloque B

# Qué veremos en Bloque B

- Diccionarios.
- Iteración sobre diccionarios.

# Diccionarios

- Los diccionarios son otra de las estructuras de dato nativas y más avanzadas que provee Python.
- Son colecciones de pares clave-valor.
- Se definen entre llaves { … } y cada para `clave-valor` se separa por coma.
- Las claves son la entrada a los datos: cuando llamamos a un diccionario y lo "evaluamos" en la clave, obtenemos los datos que están almacenados en dicha clave.
- Las claves pueden ser strings, números, o tuplas, mientras que los valores pueden ser cualquier tipo soportado por Python.

# Diccionarios

## Orden y repetición

- Un aspecto importante de los diccionarios en **versiones modernas** de Python es que estos son **ordenados**.

- Es decir, un diccionario cuando es definido mantiene su orden hasta el final (no significa que las claves se ordenen por algún criterio).

- Otro aspecto importante que es muy útil en la práctica es que los diccionarios NO admiten claves repetidas:

    - Si la clave no existe, es creada.
    - Si la clave ya existe, su valor es modificado (actualizado).

# Diccionarios

## Creando diccionarios y accediendo a sus valores

Creación de un diccionario:

```
datos = { “nombre”: “Juan”, “apellido”: “Pérez” }
```

Imprimir el nombre y apellido:

```
print(datos[“nombre”] + “ “ + datos[“apellido”])
```

Modificar el nombre (recordar que no se admiten claves repetidas):

```
datos[“nombre”] = “Claudia”
```

# Diccionarios

## Iteración sobre diccionarios

- Los diccionarios son **iterables.**
- Similar al caso de las listas, usamos un iterador para sistematizar el recorrido de los pares clave valor.
- A diferencia de las listas en que el iterador se acopla al índice, en el caso de los diccionarios el iterador se acopla a la clave.
- Ejemplo: Imprimir todos los valores de un diccionario:

```
for clave in datos:
        print(datos[clave])
```

# Diccionarios

## El operador “in”

- Si los diccionarios no admiten claves repetidas, resulta conveniente contar con un “atajo” para determinar si una clave ya existe o no.
- Para esto tenemos el operador “in” (también es aplicable a las listas).
- El operador “in” retorna True si la clave existe en el diccionario, False en caso contrario.

```
if clave not in datos:
    print(“La clave”, clave, “no esta en el diccionario”)
```

# Ejemplo bloque B

**Ejemplo: Gestión de Inventario de Productos**

Considerando el enunciado anterior, realicemos lo siguiente:

1. Cree un diccionario llamado inventario que representará el inventario de la tienda. El inventario deberá contener **nombre**, **cantidad** y **precio** de tres productos.
2. Actualizar la cantidad de un producto solicitando los datos al usuario (nombre de producto a actualizar, y cantidad).
3. Mostrar el inventario completo recorriendo el diccionario.

# Trabajo grupal – Bloque B

# Paralelo 1

| G1 | G2 | G3 | G4 |
|---|---|---|---|
| Mónica Carvajal Romero | Rodrigo Yanez Pilgrim | Ivan Santivañez | John Gonzalez |
| Patricio Hernández Pavez | Pablo Mahuzier | Francisco Zuloaga | Rodrigo Espinoza |
| Paula Rivera Oberg | Jhonatan Ortiz | Hamid Pinilla Saah | Pablo Silva |
| Valentina Loyola Rodó | Ariel Mora | Freddy Vega | Marlenne Allendes |
| Nicolas Gonzalez | María Ignacia Strobel | Hilda Silva | Jorge San Martin |
| Nicolás Aravena | christian donoso | Teresa Peña | Carlos Gonzalez |

| G5 | G6 | G7 | G8 |
|---|---|---|---|
| Paulina Valenzuela | Gabriela Torres | Miguel Mayorga | Felipe Mercado Lopez |
| Daniela Castro | Matias Araya | Isaac Vega | Francisco Yañez |
| Jorge Moraga Calvo | Alex Riquelme | Francisco Villaseca | Matias Faundes |
| Alejandro Gonzalez | Jorge Troncoso | Matías Macaya | christopher nuñez soto |
| Francisco Vega Contreras | Valentina Gutierrez | Gabriela Karina Lopez | Angelo Farias |
| Alvaro Torres | Jorge Estefane | Rodrigo Fuenzalida Vera | yuri urzua lebuy |

# Trabajo grupal – Ejercicio #1

## 1. Registro de personas (Lista de diccionarios)

Desarrolla un programa que permita registrar el nombre y apellido de varias personas, almacenando cada registro en un diccionario y organizando todos los registros en una lista.

a. Solicita al usuario los datos de cada persona. Cada persona debe ser almacenada en un diccionario con las claves "nombre" y "apellido".
b. Guarda cada diccionario en una lista llamada personas. Cada vez que se ingresen los datos de una persona, crea un diccionario y agrégalo a la lista personas.
c. El programa debe continuar solicitando datos hasta que el usuario ingrese "FIN" como nombre.
d. Mostrar la lista de personas. Recorre la lista personas e imprime el nombre y apellido de cada persona registrada.

# Trabajo grupal – Ejercicio #2

## 2. Cursos y notas de una persona

- Incrementar la funcionalidad del programa anterior permitiendo que ahora cada persona cuente con cursos inscritos.
- Es decir, necesitamos que cada persona contenga un diccionario con claves correspondientes a cursos y cada clave con una lista con las notas de dicho curso.
- Todos los datos deben ser solicitados por la entrada estándar. Puede considerar ingresar un curso por persona. Cada curso tendrá 3 notas.
- Al finalizar, el programa debe imprimir el nombre y apellido, seguido del listado de cursos y los promedios obtenidos en cada curso.

# ¿Preguntas?

¡Hemos llegado al final de la clase!