Estructuras de Datos

INSTITUTO DE INNOVACIÓN Y TECNOLOGÍA APLICADA

IITA

Contenido

PYTHON

Pregunta Inicial:

¿Qué tipos de datos conocen hasta ahora?

INTRODUCCIÓN

¿Estructuras de de Datos?

Son formas de **organizar la información** dentro de un programa para que podamos manejarla de manera más fácil y eficiente. En Python, tenemos varias estructuras que nos ayudan a almacenar y trabajar con distintos tipos de datos.

INTRODUCCIÓN

Tipos de estructuras de datos

Listas (list)

Una lista es como una **fila de casilleros**, donde cada uno puede guardar cualquier tipo de dato (números, palabras, etc.). Además, puedes cambiar lo que guardas en cada casillero cuando quieras.

Tuplas (tuple)

Las tuplas son como las listas, pero **una vez que guardas algo en ellas, no puedes cambiar lo que guardaste.** Son útiles para datos que no deben modificarse.

Diccionarios (dict)

Un diccionario es como una **libreta de contactos,** donde puedes asociar **una clave a un valor.** Las claves deben ser únicas, y con ellas puedes encontrar los valores fácilmente.

¿Qué es una lista?

Las listas son una colección de **elementos ordenados**, lo que significa que cada elemento **tiene una posición** o índice dentro de la lista. Son **mutables**, lo que quiere decir que puedes cambiar los valores que contiene en cualquier momento, agregando, eliminando o modificando elementos.

```
# problema: si quiero preservar los datos, deberia definir tantas variables como alumnos hay
alumno_1 = "Pablo Sandoval"
alumno_2 = "nico martinelli"
"""
.
.
.
y a si los n alumnos
"""
print(alumno_1,alumno_2)
```

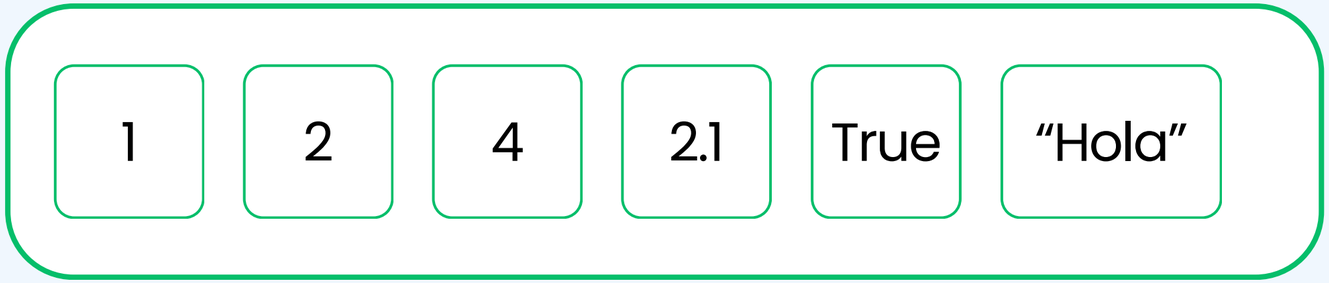
```
lista_alumnos = ['Pablo Sandoval', 'Nicolas Martinelli']
print(lista_alumnos)
```

Por ejemplo, si quiero que me presenten un listado de todos los estudiantes del curso de "**Python IITA**" incluyendo su nombre y apellido; podemos hacer todo en un solo dato y guardarlo en UNA sola variable.

..y como accedo a un elemento?

```
mi_lista = [1,2,4,2.1,True, "Hola"]
print(mi_lista[0])
```

```
1
```



.... y como obtengo el tamaño de la lista?

Utilizamos la función **len()** para obtenerla **longitud o el tamaño de la lista**

```
mi_lista = [1,2,4,2.1,Tru
print(len(mi_lista))
```

```
6
```

0	1	2	3	4	5	si leemos de izquierda a derecha
-6	-5	-4	-3	-2	-1	si leemos de derecha a izquierda

¿Qué es una porción de listas?

Una porción de lista en Python, también llamada rebanada o slice, es una **técnica** que permite **obtener una parte de una lista sin modificar la original.** Usamos los índices de la lista para seleccionar los elementos que queremos extraer.

```
mi_lista = ['A','E','I','O','U']
# mi_lista[valor_inicial:valor_final]
print(mi_lista[0:2]) # ['A','E']
print(mi_lista[1:3]) # ['E','I']
print(mi_lista[2:]) # ['I','O','U']
print(mi_lista[:3]) # ['A','E','I']
print(mi_lista[:]) # ['A','E','I','O','U']
print(mi_lista[:-1]) # ['A', 'E', 'I', 'O']
```

AYUDITA DEL PROFE NICO:

Si es dificil hacer el slicing mentalmente, imagina que hay una coma antes del primer elemento (la coma cero), entonces a partir de est0, todo elemento que quede **encerrado en las comas en el rango** que tengas sera nuestra nueva sub lista

PYTHON

¡Probemos!

append(elemento)

Añade un elemento al final de la lista

```
lista_ejemplo = ["A", "B", "C", "D", "E"]
lista_ejemplo.append("F")
print(lista_ejemplo)
#Salida:
["A", "B", "C", "D", "E","F"]
```

insert(indice,elemento)

inserta un elemento en una lista, especificando por índice la posición del mismo

```
lista_ejemplo = ["A", "B", "C", "D", "E"]
lista_ejemplo.insert(2,"Y")
print(lista_ejemplo)
#Salida:
["A", "B", "Y", "C", "D", "E"]
```

extend(otra_lista)

concatena listas. Debe llamarse con un único argumento: otra lista! (Es lo mismo que sumar listas)

```
lista_ejemplo = ["A", "B", "C", "D", "E"]
lista_ejemplo.extend(["casa","hotel"])
print(lista_ejemplo)
#Salida:
["A", "B", "C", "D", "E","casa","hotel"]
```

PYTHON

¡Probemos!

remove(elemento)

Elimina la primera aparición de un valor en la lista. CUIDADO, si no encuentra algo devuelve error

```
lista_ejemplo = ["A", "B", "C", "D", "E","B"]
lista_ejemplo.remove("B")
print(lista_ejemplo)
#Salida:
["A", "C", "D", "E", "B"]
```

pop(indice)

Remueve el ultimo elemento existente en la lista y lo devuelve. Si pasas un indice, remueve el elemento en esa posición

```
mi_lista = ['A','E','I','O','U']
ultimo_elemento = mi_lista.pop() # 'U'
print(ultimo_elemento)
primer_elemento = mi_lista.pop(0) # 'A'
print(primer_elemento)
print(mi_lista) # ['E', 'I', 'O']
```

A PRACTICAR

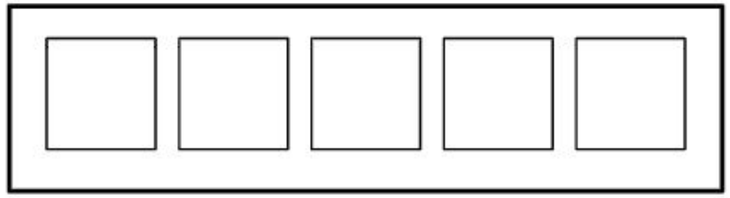
01 Meter los números del 1 al 35 en una lista y mostrarla en pantalla. Hacer lo mismo para un rango de números indicado por un usuario.

02 Pedir una palabra al usuario, meter los todos sus caracteres en una lista y mostrarla en pantalla.

03 Mostrar uno por uno los elementos de una lista

Ayuda Memoria

```
miLista = [1, 2, 4, 2.1 , "Hola"]
```



append() => añade elemento al final

insert() => inserta un elemento donde digas

remove() => elim. 1ra aparicion

pop() => remueve y retorna el ultimo elemento

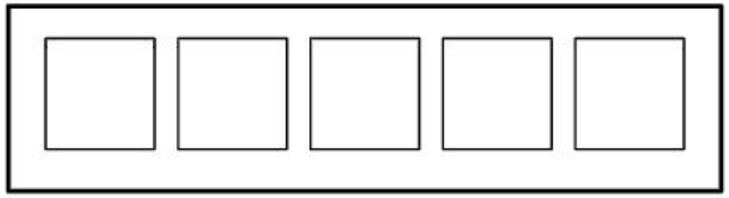
A PRACTICAR

01 Pedir un numero y cargar su tabla de multiplicar en una lista

02 **Pide una cadena (string) por teclado, mete los caracteres en una lista sin espacios.**

Ayuda Memoria

```
miLista = [1, 2, 4, 2.1 , "Hola"]
```



append() => añade elemento al final

insert() => inserta un elemento donde digas

remove() => elim. 1ra aparicion

pop() => remueve y retorna el ultimo elemento

¿Qué es una tupla?

Las tuplas son similares a las listas, pero con una diferencia importante: son **inmutables,** lo que significa que una vez que creas una tupla, no puedes cambiar sus valores. Son útiles cuando tienes datos que no deben ser modificados.

- Al igual que las listas, pueden contener diferentes tipos de datos.
- **No** puedes agregar, eliminar ni modificar elementos una vez creada la tupla.

```
# LISTA A TUPLA
numeros = [1,2,3,4,5]
tupla_numeros = tuple(numeros)
print(type(tupla_numeros))
print(tupla_numeros)
```

```
coordenadas = (10, 20)
```

```
lista_numeros = list(tupla_numeros)
print(type(lista_numeros))
print(lista_numeros)
```

A PRACTICAR

01 Crea una tupla con valores ya predefinidos del 1 al 10, pide un índice por teclado y muestra los valores de la tupla en esa posición.

02 Crea una tupla que contenga todos los meses de un año. Luego pedir al usuario un numero y mostrar que mes representa

03 Crea una tupla con números, pide un numero por teclado e indica cuantas veces se repite.

Ayuda Memoria

Se usa parentesis
tuple() => convierte a una tupla
list() => convierte a una lista

¿Qué es un diccionario?

Los diccionarios son una estructura que nos permite almacenar datos en pares de clave-valor. Esto significa que cada elemento tiene una "clave" única que se usa para acceder a su "valor" correspondiente. Son útiles cuando quieres relacionar datos de manera clara y directa.

- Las claves deben ser **únicas** (no puedes tener dos claves iguales), y pueden ser de **distintos tipos** (números, textos, etc.).
- Los valores pueden ser de cualquier tipo de dato, y puedes modificarlos.

```
persona = {"nombre": "Juan", "edad": 25, "ciudad": "Salta"}
print(persona["nombre"])  # Muestra 'Juan'
persona["edad"] = 26      # Cambia la edad a 26
```

A PRACTICAR

01 Crea un programa que pida al usuario el nombre de un mes (Enero, Febrero, etc.) y diga cuántos días tiene (por ejemplo, 30. Para simplificarlo vamos a suponer que febrero tiene 28 días.

02 Crear un programa que pida al usuario un nombre de un alumno, y luego muestre la lista de notas de ese alumno. Usar diccionarios

Ayuda Memoria

```
# estructuras de clave - valor
dicc_ejemplo = {
    'usuario': 'psandoval',
    'contraseña': '1234'
    }
print(dicc_ejemplo['usuario']) # psandoval
```

IITA

¡Muchas Gracias!

Por su atención