Parte 1. Fundamentos de la programación

Unai Pérez-Goya
Área de Lenguajes y Sistemas Informáticos

Curso 2024/2025

Parte 1.

Fundamentos de la programación

Sesión V: Estructuras de datos avanzadas (list) y for (each)

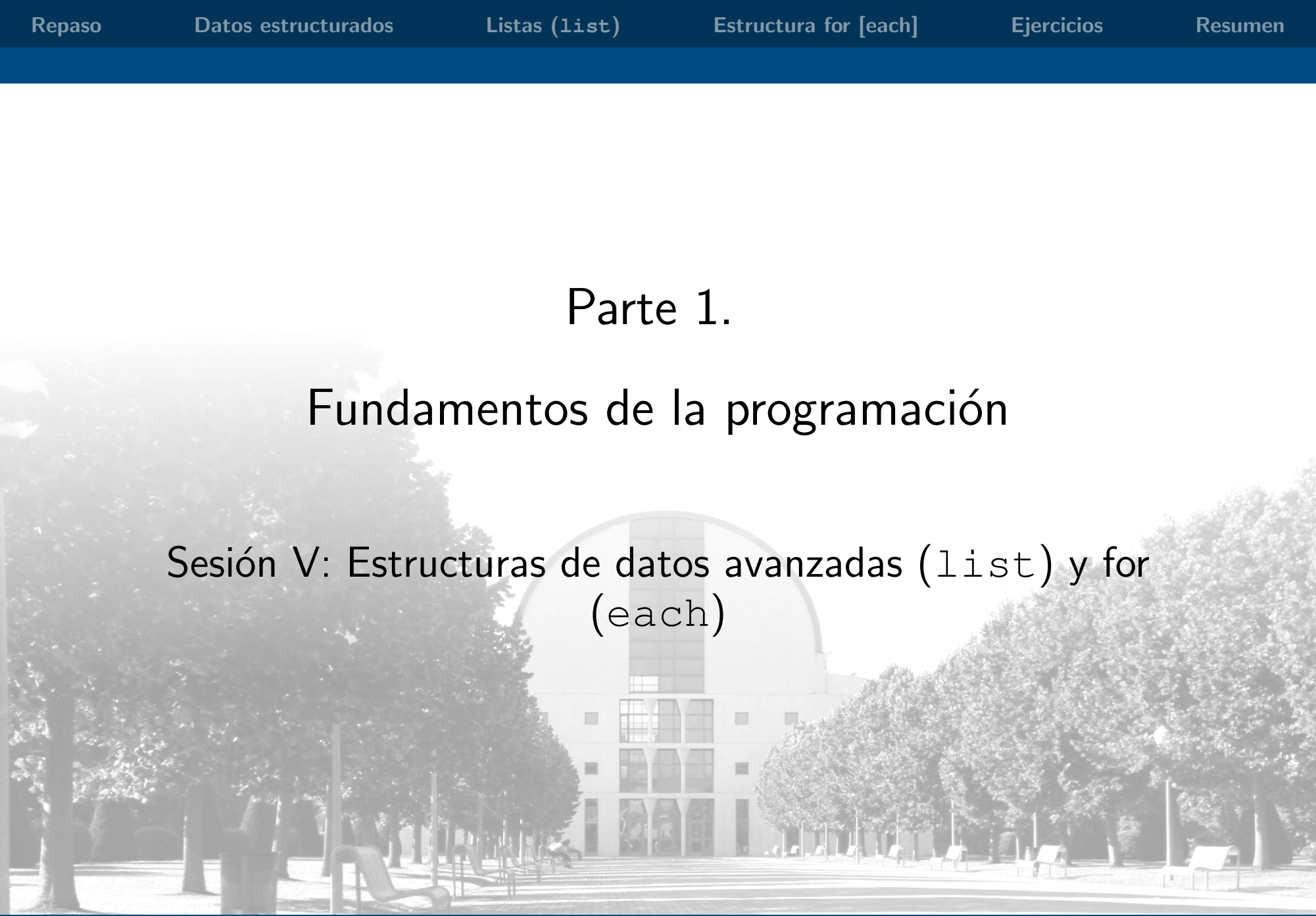


Tabla de contenidos

Repaso

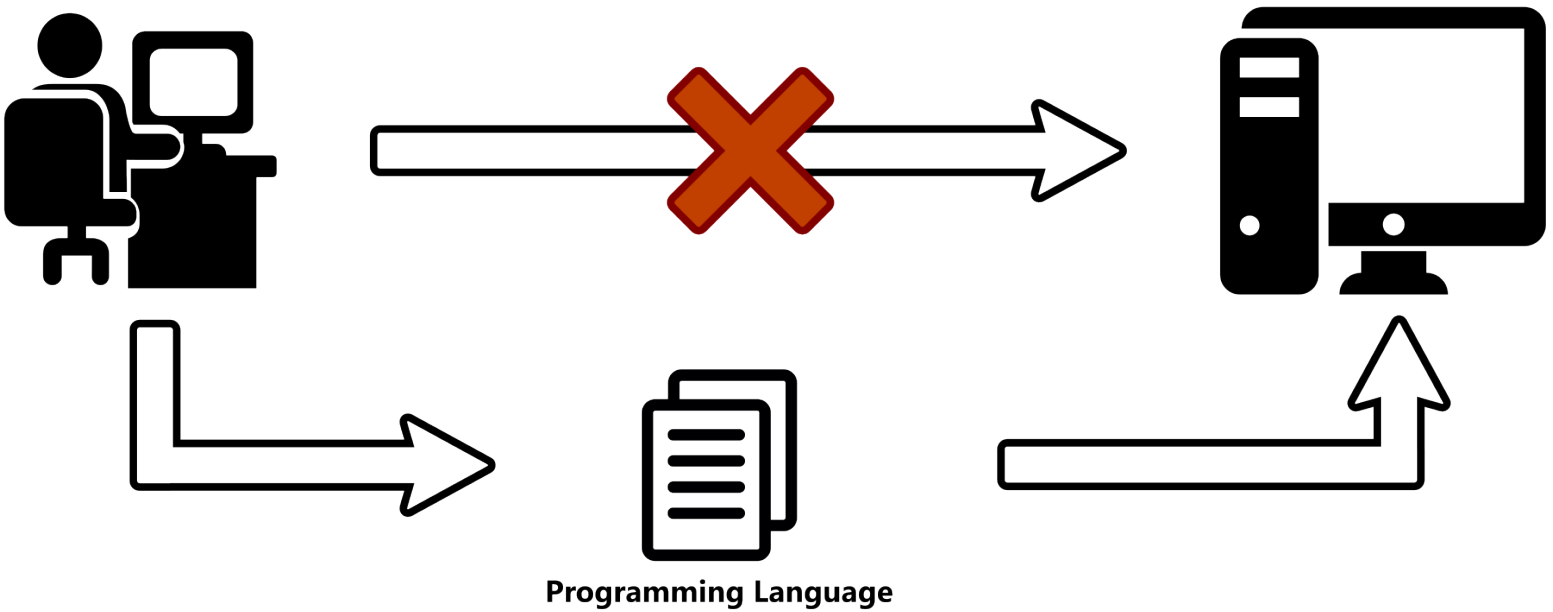
Sesión I

Lista de contenidos

1. Introducción a Python. Historia y características;
2. Lenguajes de programación. Concepto y uso;
3. Lenguajes interpretados *vs.* compilados. Programas ejecutables. Máquina intérprete;
4. Entorno de desarrollo integrado (IDE)

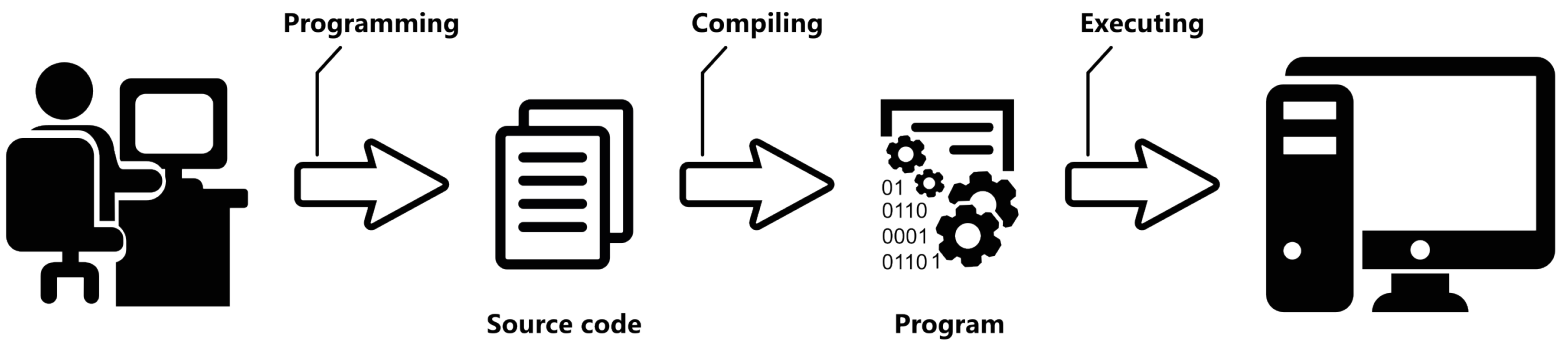
Sesión I

Lenguajes de programción



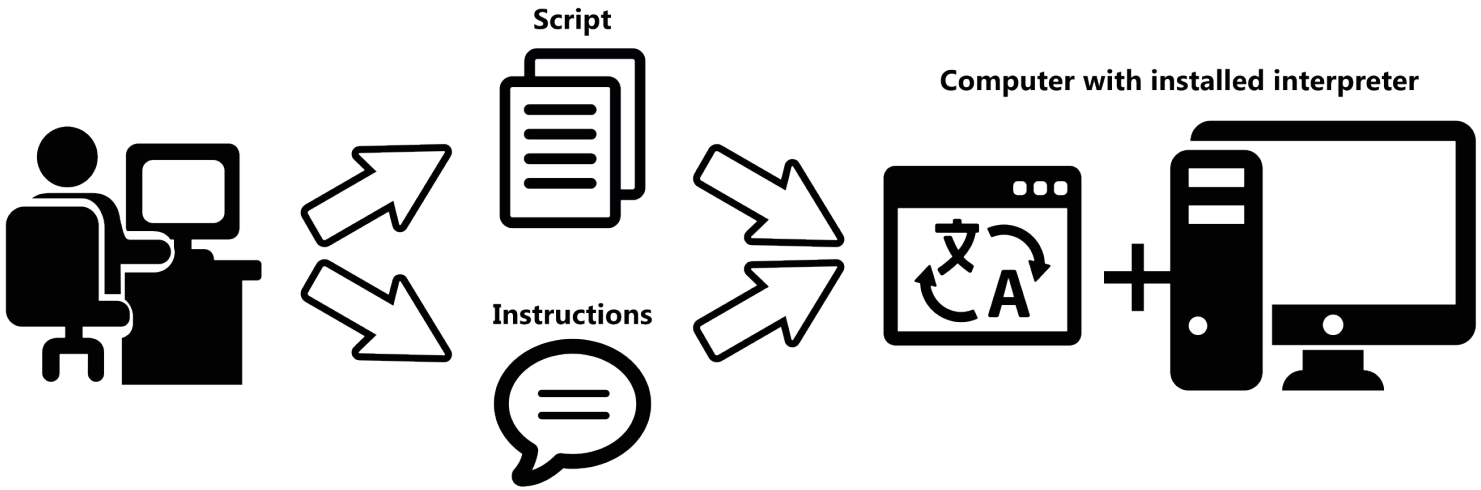
Sesión I

Proceso de compilación



Sesión I

Compilado vs. Interpretado



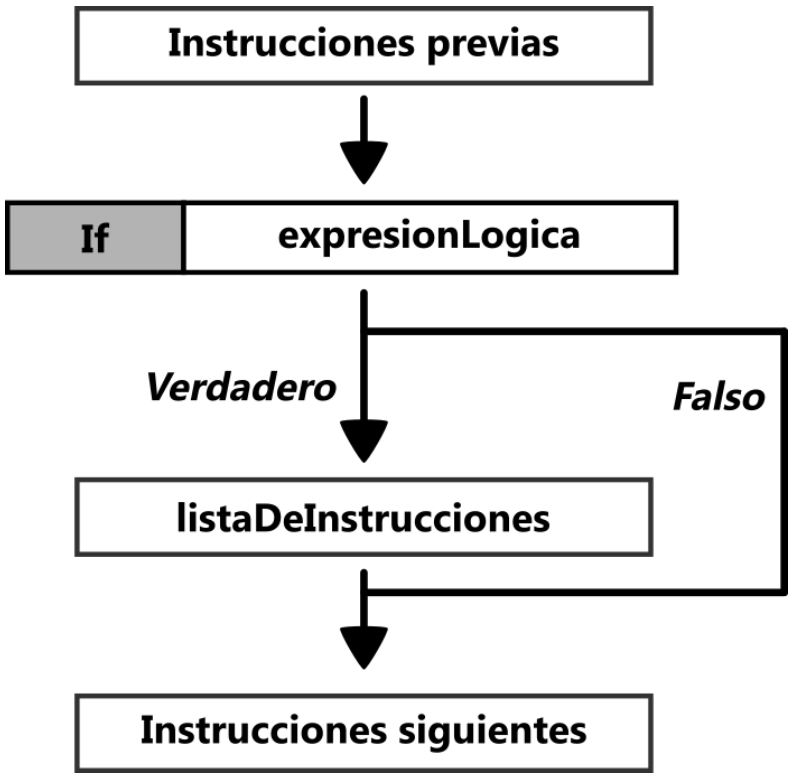
Contenido de la sesión II

- Variables y uso básico en python
- Declaración y asignación;
- Tipos
  - Numéricos
  - Texto
  - Booleanos
- Constantes

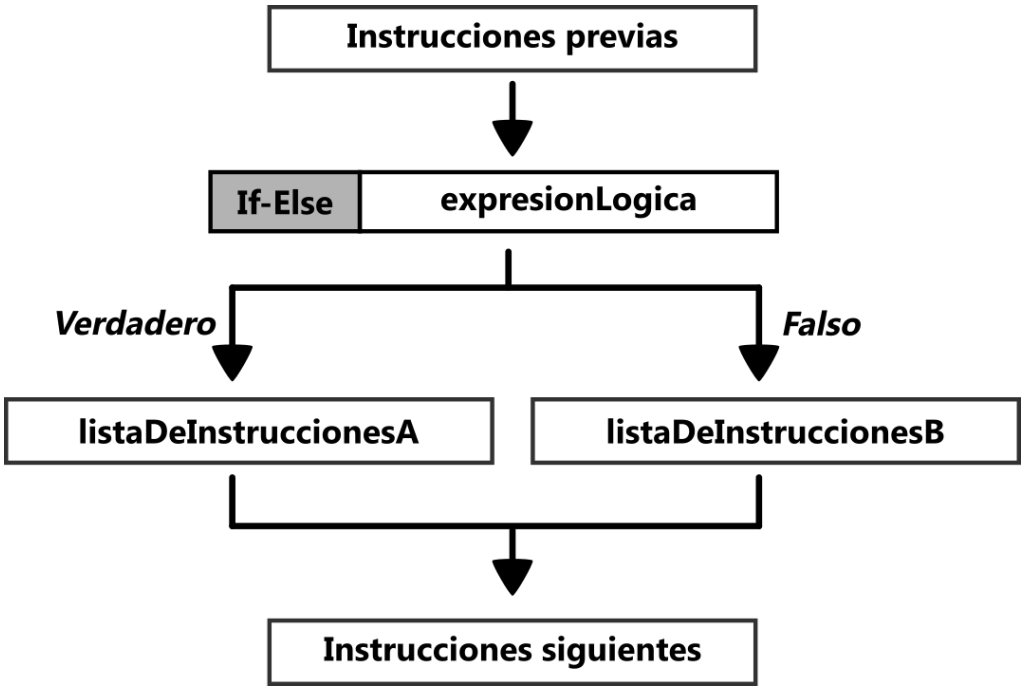
Sesión III

- Estructura de los programas
- Estructuras de control
  1. Estructura `if`
  2. Estructura `if-else`
- Funciones, capsulación y ámbito de la variable
- Ejercicios.

Sesión III


Instrucciones previas
If
expresionLogica
Verdadero
Falso
listaDeInstrucciones
Instrucciones siguientes

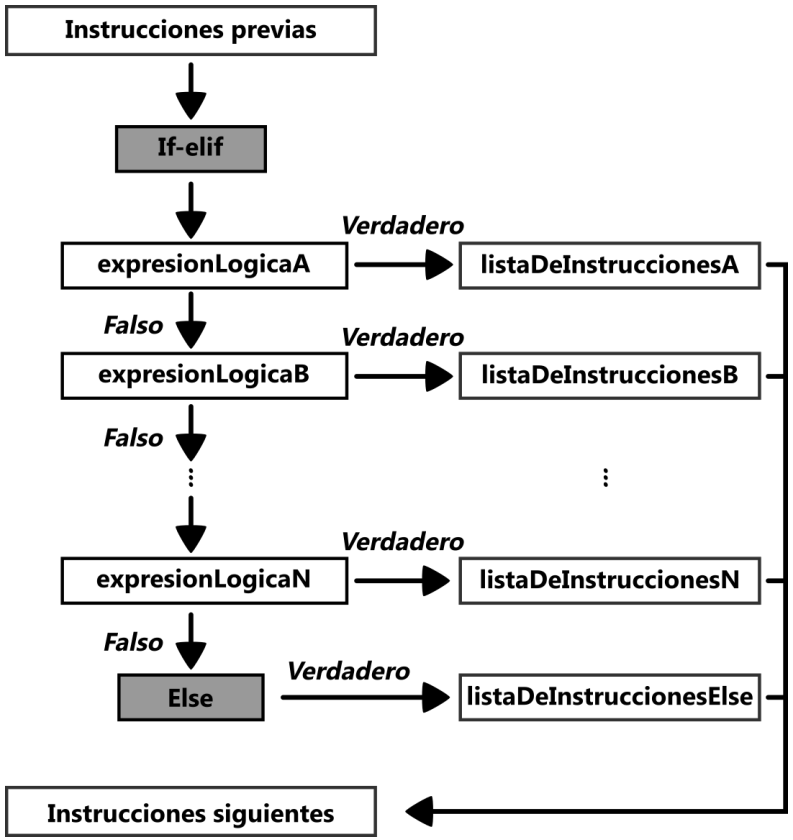
Sesión III


Instrucciones previas
If-Else
expresionLogica
Verdadero
Falso
listaDeInstruccionesA
listaDeInstruccionesB
Instrucciones siguientes

Sesión IV

- Interacción con el usuario;
- Estructura `if-elif`.
- Estructura `if-elif-else`.
- Funciones sobre variables de tipo `str`

Sesión IV


Instrucciones previas
If-elif
Verdadero
expresionLogicaA
listaDeInstruccionesA
Falso
Verdadero
expresionLogicaB
listaDeInstruccionesB
Falso
Verdadero
expresionLogicaN
listaDeInstruccionesN
Falso
Else
Verdadero
listaDeInstruccionesElse
Instrucciones siguientes

Datos estructurados

Datos escalares

- Hasta ahora todas las variables guardan un único valor.
  - Estos son datos escalares.
- Si queremos gestionar cinco números enteros, tenemos que asignar cinco variables.
  - Por ejemplo, `a`, `b`, `c`, `d` y `e`.
- Es bastante incómodo.
- Mejor dicho, es muy incómodo.

Datos escalares

- La potencia de los ordenadores consiste en procesar grandes cantidades de datos...
- ...sería bueno gestionar/asignar variables específicas para grandes cantidades de datos.
  - Ejemplo: *Quiero leer 10 números enteros*.
  - Ejemplo: *Quiero asignar una matriz del elemento 5x5*.
- Para hacerlo esto se crearon datos estructurados.

Datos estructurados

Datos estructurados

- Un tipo de dato estructurado es aquel que permite almacenar más de un dato.
- *Existen diferentes interpretaciones para los datos que deben mantener el orden*:
  - Vector como secuencia de números.El orden será la posición de la secuencia.
  - Matriz como tabla de números, donde el orden de los elementos está dado por filas y columnas.
  - ...
- Aprenderemos varios tipos de datos con diferentes tipos de estructuras.
  - La idea común es almacenar/acceder a más de un dato en una misma estructura.

Datos estructurados

- Listas de valores (`list`);
- Grupos de valores (`set`);
- Diccionarios (`dict`);

  +
- Matrices;

Listas (list)

El concepto de lista

- De los datos estructurados es el más básico, al menos conceptualmente
- Una lista es una secuencia de elementos
  - Cada elemento tendrá una posición en la lista.
  - Puede haber dos elemento iguales, es decir, se pueden repetir.
  - Los elementos pueden ser eliminados de la lista, además se pueden añadir nuevos elementos en cualquier posición
  - La logitud de la lista viene dada por el número de elementos
- La lista puede tener tantos elementos como se desee, variará dinámicamente en función de las necesidades del programador.

Listas (list): declaración

- Para asignar una lista vacía de Python se debe utilizar la función `list ()`.
- Se puede asignar con los valores iniciales igual que las variables
- Es decir:
  - Asignar número entero:`valor = 5`;
  - Asignar lista vacía:`lista = list ()`;
  - Asignar una lista de números enteros:`lista = [5]`;
  - Asignar lista de números reales:`lista = [-5.3,2.5]`;
  - ...
- Al ejecutar `type (lista)`, aparecerá por pantalla `<class 'list'>` .

Listas (`list`): Seleccionar elementos

- Las listas son secuencias de elementos (sin huecos entre elementos);
- Cada elemento puede identificarse por su posición en la lista;
- El primer elemento de la lista estará en posición en cero (0);
- Si una lista contiene *n* elementos el último elemento de la lista será `n - 1`;
- Ejemplo (`lista = [5,1,5,67,243, -4]`):
  - El primer elemento `lista [0]`, tendrá el valor `5`.
  - `lista [1]` el segundo, con el valor `1`...
  - El último elemento, `lista[5]` tendrá el valor`-4`.

Listas (list): Seleccionar elementos

- ¿Qué ocurre si intentamos seleccionar un valor que no existe? (Ejemplo (`lista=[5,1,5,67,243,-4]`):)
  - Primer elemento `lista[6]` ⟶ `IndexError`
  - Índice negativo `lista[-1]` ⟶ `-4`
  - Índice negativo `lista[-6]` ⟶ `5`
  - Índice negativo `lista[-7]` ⟶ `IndexError`
- La selección de un elemento de la lista se puede utilizar como una variable escalar `duplicar = lista[0]*5`
  - `duplicar` ⟶ `25`
  - Pero... `lista[0]` ⟶ `5`

Listas (list): selección parcial

- Ademas de poder seleccionar uno a uno los elementos de la lista, también se pueden crear sublistas
- La selección se puede realizar con el carácter `:`;
- Selección `variable[x:y]`
  - Donde, `x` es la posizion inicial de la selección
  - Donde, `y` es la posizion inicial de la selección (no entrará en la sección)
- Por jemplo: (con la lista `lista=[5,1,5,67,243,-4]`)
  - `lista[:]` $\longrightarrow$ `[5,1,5,67,243,-4]`.
  - `lista[2:])` $\longrightarrow$ `[5,67,243,-4]`.
  - `lista[:2])` $\longrightarrow$ `[5,1]`.
  - `lista[1:2])` $\longrightarrow$ `[1]`.
  - `lista[1:1])` $\longrightarrow$ `[]`.

Lista (list) Asignacion

Asignación en lista

- Cada elemento de la lista se puede usar como un valor...
- Cuando un elemento de la lista existe, este se puede reasignar directamente

!! ¡Cuidado! Si el elemento no existe, aparecerá un error

- Por ejemplo: (con la lista `lista=[5,1,5,67,243,-4]`)
  - `a[1]=9` ⟶ `[5,9,5,67,243,-4]`.
  - `a[-1]=9` ⟶ `[5,9,5,67,243,9]`.
  - `a[6]=9` ⟶ `IndexError`.
- Entonces... ¿Cómo añado un nuevo elemento a la lista?

Ejemplo de `list`

Añadir un elemento

- Para añadir elementos a una lista se pueden utilizar diferentes funciones
- Ejemplo: (dada la lista `lista=[5,1,5,67,243,-4]` y ejecutando secuencialmente las siguientes instrucciones)
  - Añadir un elemento en un posición de la lista:
    - `lista.insert(3,2)` $\longrightarrow$ `[5, 1, 5, 2, 67, 243, -4]`.
  - Añadir un elemento o varios al final de la lista:
    - `lista.append(1)` $\longrightarrow$ `[5, 1, 5, 2, 67, 243, -4,1]`.
    - `lista.extend([9,10])` $\longrightarrow$ `[5, 1, 5, 2, 67, 243, -4,9,10]`.

Ejemplos de listas

Eliminar elementos

- Se pueden realizar diferentes operaciones sobre la lista
- Ejemplo: (dada la lista `lista=[5,1,5,67,243,-4]` y ejecutando secuencialmente las siguientes instrucciones)
  - Eliminar un elemento por valor
    - `lista.remove(67)` $\longrightarrow$ `[5,1,5,,243,-4]` $\longrightarrow$ `[5,1,5,243,-4]`.
  - Eliminar un elemento por posición
    - `del lista[1]` $\longrightarrow$ `lista=[5,,5,243,-4]` $\longrightarrow$ `lista=[5,5,243,-4]`.

Ejemplos de listas

Ejemplo: insert vs. append y eliminar

- **Ejemplo 1** Crea un programa que declare una lista con valores y utilice las funciones `append`, `remove` e `insert`. ¿Qué hace cada función?

Ejemplos de listas

Ejemplo: listas

```
1  # lista declarar
2  lista = [1,2,3,5,7,11]
3  print('a)',lista)
4
5  # anadir un valor a lista
6  lista.append(13)
7  print('b)',lista)
8
9  # listako balio bat ezbatu
10 lista.remove(13)
11 print('c)',lista)
12
13 # listako 3. posizioan balioa gehitu
14 lista.insert(3,5)
15 print('d)',lista)
```

Código: Funciones de lista

Ejemplos de listas

Ejemplo: sumar dos listas

- **Ejemplo 2** Crear un programa que declare dos listas que almacene sus valores en una tercera lista.

Ejemplos de listas

Ejemplo: sumar dos listas

```
1 lista1 = [1,2,3,5,7,11]
2 lista2 = [11, 12, 13, 15, 17, 21]
3
4 lista_nueva = [lista1, lista2]
5 lista_sumada = lista1+lista2
6 lista_extends = lista1.extend(lista2)
```

Código: suma dos listas.

Ejemplos de listas

Ejemplo: comparaciones entre listas

- **Ejemplo 4** ¿Se pueden comparar listas? ¿Cómo se comparan las listas en python?

Ejemplos de listas

Ejemplo: comparaciones entre listas

```
1  # variables
2  lista1 = [1, 2, 3, 5, 7, 11]
3  lista2 = [1, 2, 3, 5, 7, 11]
4  lista3 = [1, 2, 3, 5, 7]
5  lista4 = [11, 12, 13, 15, 17, 21]
6  lista5 = [1.0, 2.0, 3.0, 5.0, 7.0, 11.0]
7
8  # mostrar en pantalla las comparaciones
9  print(lista1==lista2)
10 print(lista1==lista3)
11 print(lista1==lista4)
12 print(lista1==lista5)
```

Código: comparaciones entre listas.

Ejemplos de listas

Ejemplo: crear una lista con carácteres

- **Ejemplo 5** Crea una lista que se genere a partir de una cadena de carácteres (`stre`) y modifica los datos de la lista. ¿Cuáles son las funciones que acepta esta lista? ¿Se admiten diferentes tipos de datos en ese carácter lista?

Ejemplos de listas

Ejemplo: Comparación entre listas

```
1  # variables
2  lista = list('Hola')
3
4  # mostrar los cambios
5  print(lista)
6  print(lista[0])
7  lista.insert(3,'E')
8  print(lista)
9  lista.append('Fin')
10 print(lista)
```

Código: Zerrenden arteko konparaketa.

Estructura for [each]

Estructura `for` [each]

- Cuando se utiliza una lista, sus elementos se pueden utilizar uno a uno.
  - Solo se ejecuta en un elemento y una sola vez
  - Mediante `if` podemos tomar decisiones para aplicar un función
  - Pero no podemos aplicar una función `n` veces
- Entonces, ¿cómo podemos utilizar los datos de una lista si tienen un número de elementos flexibles?

for egitura

- Para visualizar los elementos de una lista de Python (todos) uno por uno se puede utilizar la estructura `for`.
- La estructura `for` en python, funciona en realidad como un `foreach` de otros lenguajes
  - Se puede utilizar sobre cualquier collección (esto lo veremos más adelante)
  - Realiza tantas iteraciones como elementos tiene la colección
- ¿Qué es una iteración?
  - Iteración es la repetición de un fragmento de código dentro de un programa de ordenador.

`for` egitura

- Sintaxis de la estructura `for`:

☞
```
for x in secuencia:
    instrucciones
```

- ...donde `secuencia` puede ser cualquier colección.
- ...donde `x` es el nombre asignado por el programador
  - En cada iteración tomará un valor de la lista
  - En orden... iteración 1, el valode del elemnto 0, 2 iterazioan, el valor del elemento 1...

for en python (foreach)

Ejemplo 1: recorrer una lista con for

- **Ejemplo 1** Utiliza `for` para recorrer todos los elementos de una lista

Ejemplos de listas

Ejemplo 1: bi lista batu

```
1 # variables
2 lista = [1,1,2,3,5,8,13]
3 # Mostrar los elementos
4 for elemento in lista:
5     print(elemento)
```

Código: Mostrar los elementos de una lista 1 a 1.

for en python (foreach)

Ejemplo 2: recorrer una lista con for

- **Ejemplo 2** Utiliza `for` para mostrar una sublista, de una lista mayor.

for en python (foreach)

Ejemplo 2: recorrer una lista con for

```
1 # aldagaiak
2 lista = [1,1,2,3,5,8,13]
3 # logica del programa
4 for elemento in lista[2:5]:
5     print(elemento)
```

Código: recorrer lista con for.

for en python (foreach)

Ejemplo 2: recorrer una lista con for

- **Ejemplo 2** Crea un programa que duplique todos los elementos de una lista.

for en python (foreach)

Ejemplo 2: utilizando for sobre listas

```
1  # funciones
2  def duplicar_lista(lista):
3      aux_lista = list()
4      for elemento in zer:
5          aux_lista.append(lista*2)
6      return aux_lista
7
8  # variables
9  lista = [1,1,2,3,5,8,13]
10 # logica del programa
11 lista_duplicada = duplicar_lista(lista)
```

Código: Duplicar los valores de una lista.

Ejercicios

Ejercicios

- **Ejercicio 1** Escribe la función que une todos los elementos de una lista.

Ejercicios

- **Ejercicio 2** Escribe la función que identifica el mayor de los elementos de una lista. La función se llamará `maximo`.

Ejercicios

- **Ejercicio 3** Escribe la función que identifica el menor de los elementos de una lista. La función se llamará `minimo`.

Ejercicios

- **Ejercicio 4** Escribe la función que calcula la media de los elementos que hay en una lista. El nombre de la función será `promedio`.

Ejercicios

- **Ejercicio 5** Escribe el programa en el que se pide al usuario 10 números y se visualiza la estadística descriptiva del conjunto de elementos. Las estadísticas que mostrará serán, máximo, mínimo, media y suma de todos los elementos.

Resumen

Resumen de contenidos

- Datos estructurados
- Uso de `list`
- Recorrer listas con `for [each]`
- Ejercicios