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 IV: Interacción con el usuario y estructura `if-elif-else`

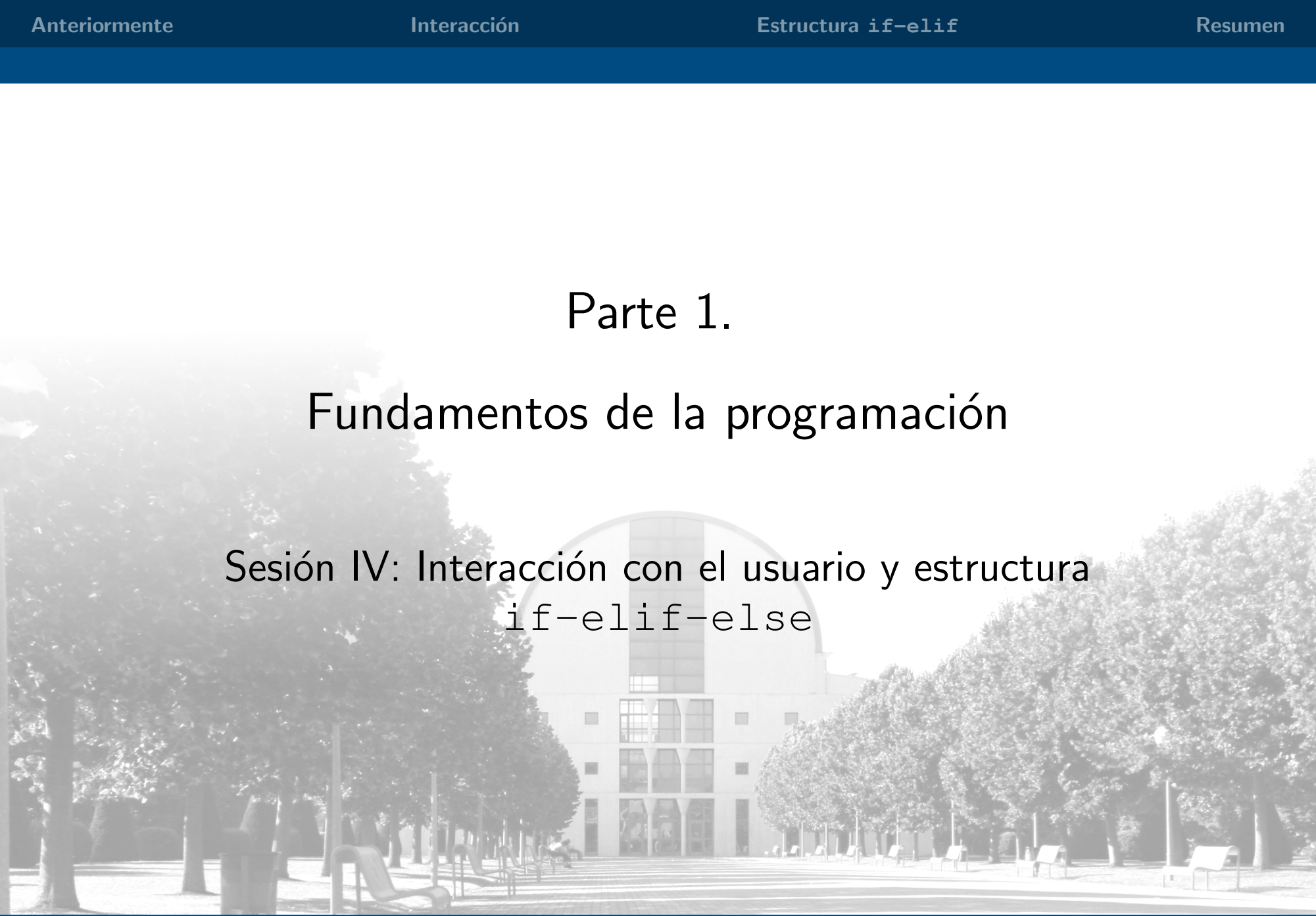


Tabla de contenidos

Anteriormente

Interacción con el usuario

Estructura `if-elif`

Resumen

Resumen de sesiones anteriores

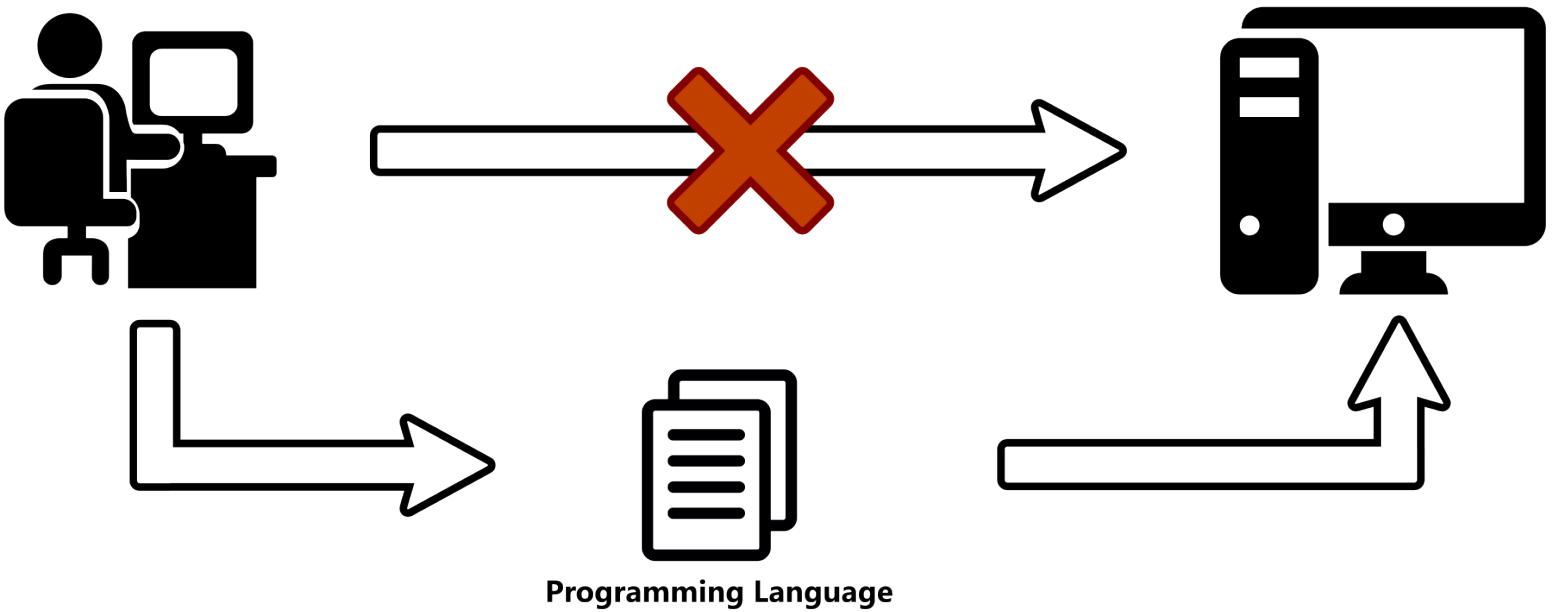
Contenido de la sesión II

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)

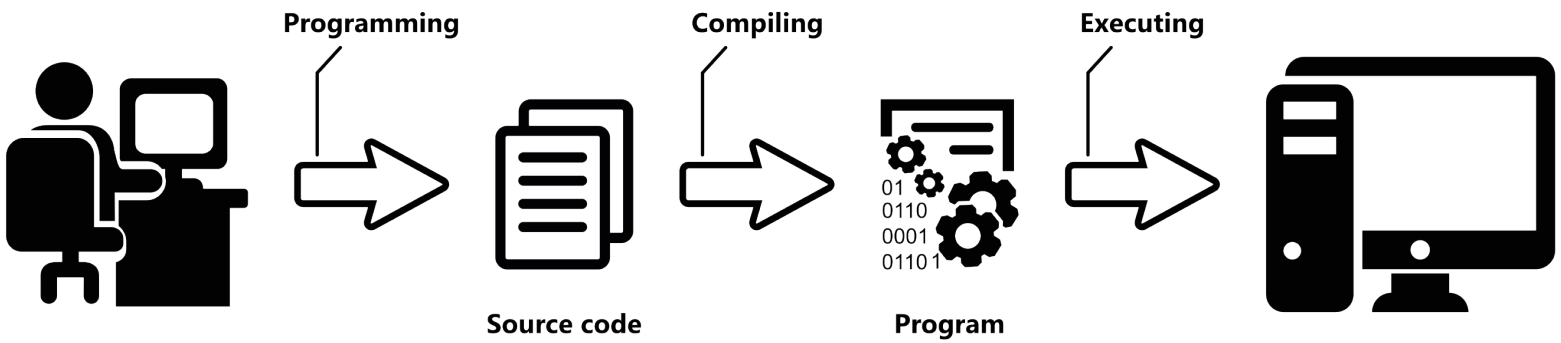
En sesiones anteriores

Lenguajes de programción



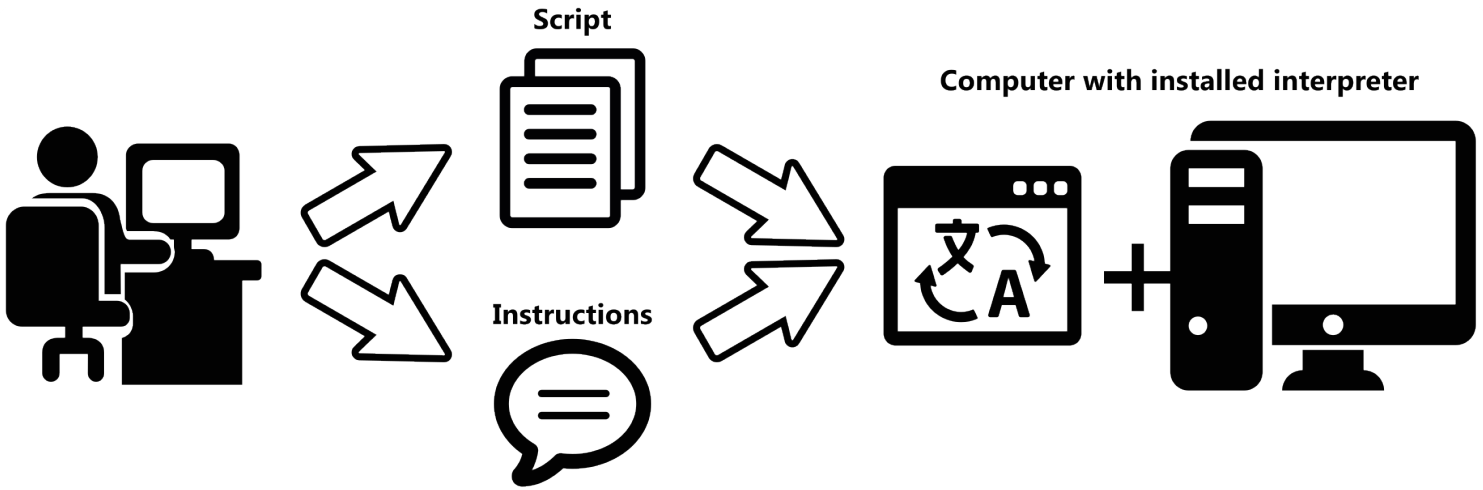
En sesiones anteriores

Proceso de compilación



En sesiones anteriores

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.

Interacción con el usuario

Funcionamiento de los programas

- Actualmente los ordenadores son utilizados por múltiples usuarios
  - Un usuario utiliza el ordenador
  - Puede instalar aplicaciones
  - Pero... no tiene por qué saber programar
- Hasta ahora, a las variables les hemos asignado un valor *fijo* y predefinido (por ejemplo `a = 5.0`)

Ejercicio 1

- **1. Ejercicio** Crear una calculadora que realice sumas y restas de dos números enteros.

Ejercicio 1: funciones necesarias para realizar el ejercicio

```
1 # funciones
2 def suma(num1, num2):
3     return num1,+num2
4
5 def resta(num1, num2):
6     return num1,-num2
```

Código: Funciones necesarias para crear la calculadora.

Ejercicio 1: unciones necesarias para realizar el ejercicio

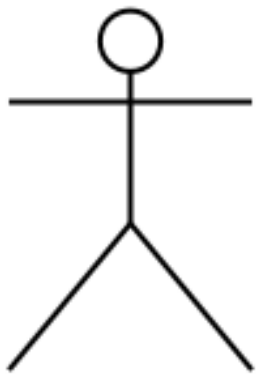
- ¿qué no es correcto en el siguiente código?

```
1  # variables
2  num1 = 5
3  num2 = 3
4  op = '+'
5
6  # logica del programa
7  if op='+':
8     resultado = suma(num1,num2)
9  else:
10    resultado = resta(num1,num2)
11
12 print(resultado)
```

Código: código necesario para el ejercicio 1.

Interacciones de uso en la calculadora

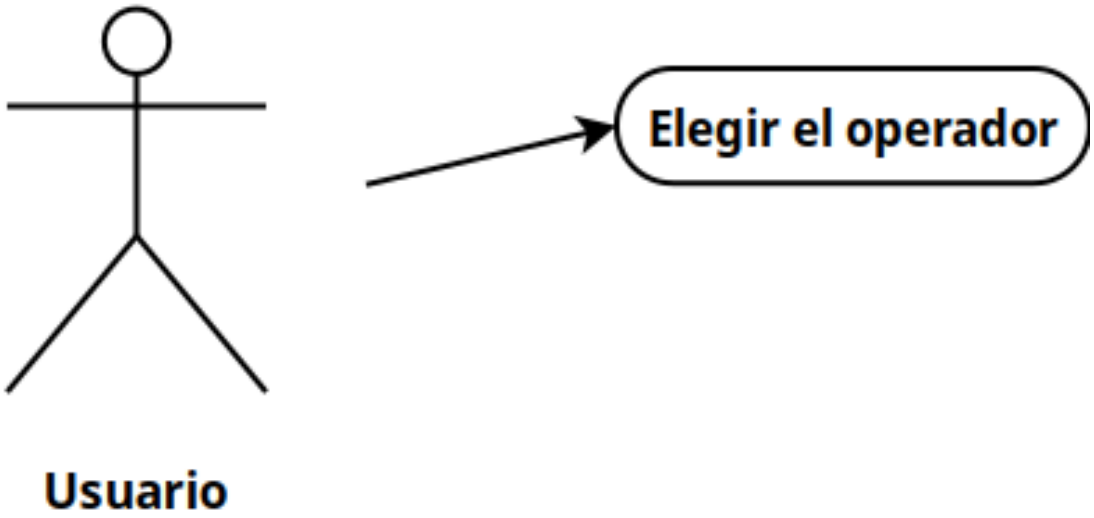
- Pero, los programas deben adaptarse a los usuarios
- Por eso, todo programador debe pensar en las acciones que deberán realizar los usuarios
- ¿Cuáles serían los datos que debe aportar el usuario? ¿Qué debe hacer para utilizar la calculadora?



Usuario

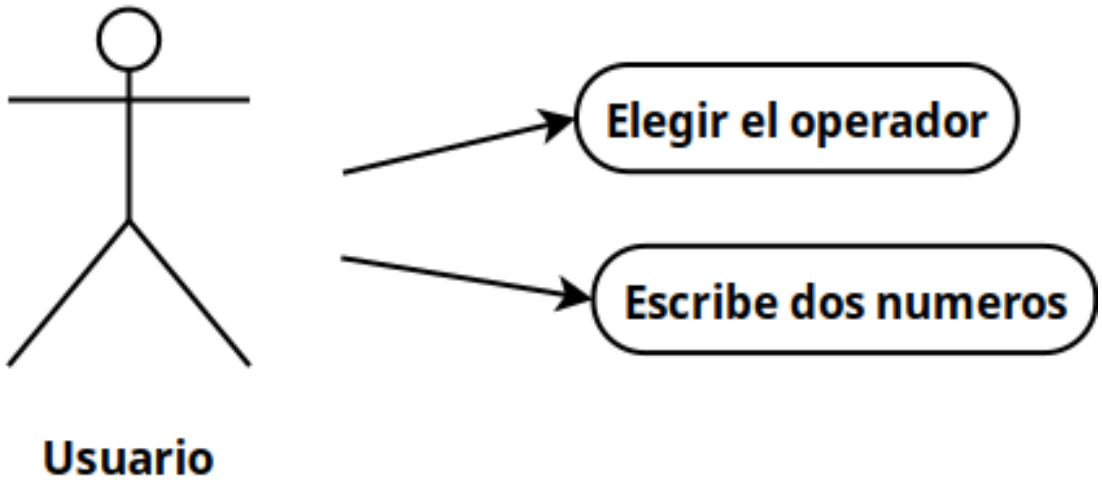
Interacciones de uso en la calculadora

- Pero, los programas deben adaptarse a los usuarios
- Por eso, todo programador debe pensar en las acciones que deberán realizar los usuarios
- ¿Cuáles serían los datos que debe aportar el usuario? ¿Qué debe hacer para utilizar la calculadora?



Interacciones de uso en la calculadora

- Pero, los programas deben adaptarse a los usuarios
- Por eso, todo programador debe pensar en las acciones que deberán realizar los usuarios
- ¿Cuáles serían los datos que debe aportar el usuario? ¿Qué debe hacer para utilizar la calculadora?



Leyendo información del teclado

- Lo más habitual en este nivel es leer desde el teclado.
- Cada vez que queramos leerlo desde el teclado, la aplicación se detendrá, solicitará información al usuario y esperará dicha interacción.

☞ `[nombre_variable] = input (mensaje)` El comando

- `input` mostrará en pantalla el contenido de `mensaje` y una vez introducida la información por el usuario la guardará en la variable `nombre_variable`.

Leyendo información del teclado

- Los valores obtenidos utilizando el comando `input` serán **SIEMPRE** de tipo texto, es decir, de tipo `str`.
- ¿Por qué?, porque el usuario puede escribir cualquier cosa desde el teclado texto, números o una combinación de las dos cosas.
- Entonces... Si queremos obtener un número del teclado ¿se pueden utilizar funciones para convertirlo al tipo deseado?

Ejercicio 2

- **Ejercicio 2** Cree una calculadora en python que realice las sumas y restas de números enteros. Solicite la información al usuario para realizar los cálculos.

Ejercicio 1: funciones básicas para crear ejercicio

```
1 # funciones
2 def suma(num1, num2):
3     return num1+num2
4
5 def resta(num1, num2):
6     return num1-num2
```

Código: funciones para crear la calculadora.

Ejercicio 1: funciones básicas para crear ejercicio

- ¿Qué es incorrecto en el siguiente código?

```
1  # variables
2  numero_1 = input('Escribe un numero: ')
3  numero_2 =  input('Escribe un numero: ')
4  operador =  input('Escribe un operador: ')
5
6  # logica del programa
7  if operador='+':
8     resultado = suma(numero_1,numero_2)
9  else:
10    resultado = resta(numero_1,numero_2)
11
12 print(resultado)
```

Código: código para que funcione la calculadora.

Pidiendo información

- Mira el código para pedir información línea a línea.

```
1 # variables
2 numero_1 = input('Escribe un numero')
```

Código: código para que funcione la calculadora.

- En la variable `numero_1`, aparecerán digitos de tipo texto. Por jemplo: `'12'` .
- Es decir, debemos cambiar el tipo de texto a numérico

```
1 # variables
2 numero_1 = input('Escribe un numero')
3 numero_1 = int(numero_1)
```

Código: código para que funcione la calculadora.

Pidiendo información

- Para el segundo número deberemos modificar también el tipo del valor

```
1 numero_2 = input('Escribe un numero')
2 numero_2 = int(numero_2)
```

Código: código para que funcione la calculadora.

- Cuando ejecute la función `input` de la primera línea el programa se quedará esperando la interacción del usuario.
- El usuario deberá escribir un número y pulsar el botón enter.
- Se deberá convertir la información leída por `input` en valor numérico.

Pidiendo información: Control de operador

- En la variable `operador` necesitaremos información sobre el operador
- Para hacer la calculadora solo necesitamos aceptar dos operaciones, así que...

```
operador = input('Escribe un operador: ')
if operador == '+':
  #Realizar suma
else:
  if operador == '-':
    # Realizar resta
  else:
    # no has introducido un operador valido
```

Código: código para que funcione la calculadora.

Ejercicio 3

✪ **3. Ejercicio** Cree una calculadora en Python que realice sumas, restas, multiplicaciones y divisiones entre números enteros. Solicite la información al usuario para realizar los cálculos.

Pidiendo información: Control de operador

```
1  operador = input('Escribe un operador: ')
2  if operador == '+':
3    #Realizar suma
4  else:
5    if operador == '-':
6      # Realizar resta
7    else:
8      if operador == '*':
9        # Realizar multiplicacion
10     else:
11       # no has introducido un operador valido
```

Código: código para que funcione la calculadora.

Estructura `if-elif`

Estructura `if-elif`

- La estructura `if` permite ejecutar (o no) una lista de instrucciones.
  - Decidir si hacer algo o no.
- La estructura `if-else` permite seleccionar una de dos listas de instrucciones.
  - Decidir cuál de los dos debe ser ejecutado.
- Pero, ¿tengo más de dos opciones?
- Las estructuras `if-else` se pueden anidar, pero la legibilidad se verá afectada

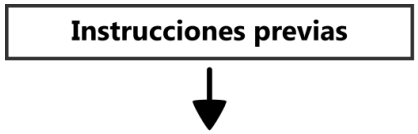
Estructura `if-elif`

- La estructura `if-elif-else` permite elegir una entre diferentes listas de instrucciones.

☞
```
if expresion_logica_A:
    instruciones_A
elif expresion_logica_B:
    instruciones_B
...
```

- Se ejecuta unicamente el bloque correspondiente a la primera expresión lógica que se cumpla.
  - Si ninguna se cumple se saltan todos los bloques contemplados en la estructura.

Estructura `if-elif`



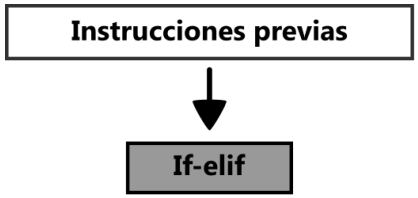
- [instrucciones anteriores]

```
if expresion_logica_A:
    instruciones_A
elif expresion_logica_B:
    instruciones_B
```

[siguientes instrucciones]

- ...

Estructura `if-elif`



- [instrucciones anteriores]

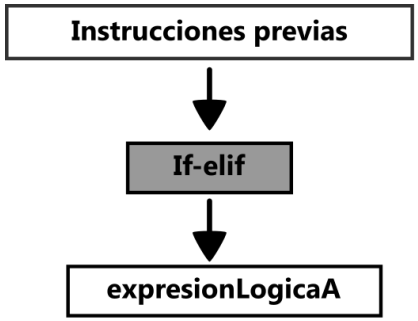
```
if expresion_logica_A:
    instruciones_A
elif expresion_logica_B:
    instruciones_B
```

[siguientes instrucciones]

- ...

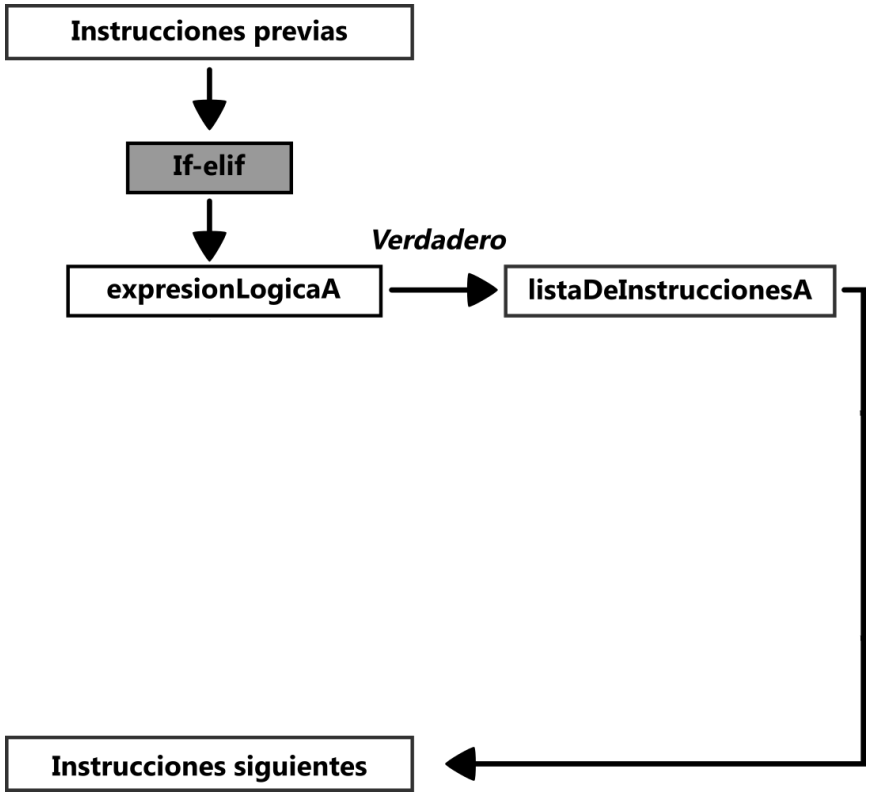
Estructura `if-elif`

```
[instrucciones
anteriores]
if expresion_logica_A:
    instruciones_A
 elif expresion_logica_B:
    instruciones_B
[siguientes
instrucciones]
...
```



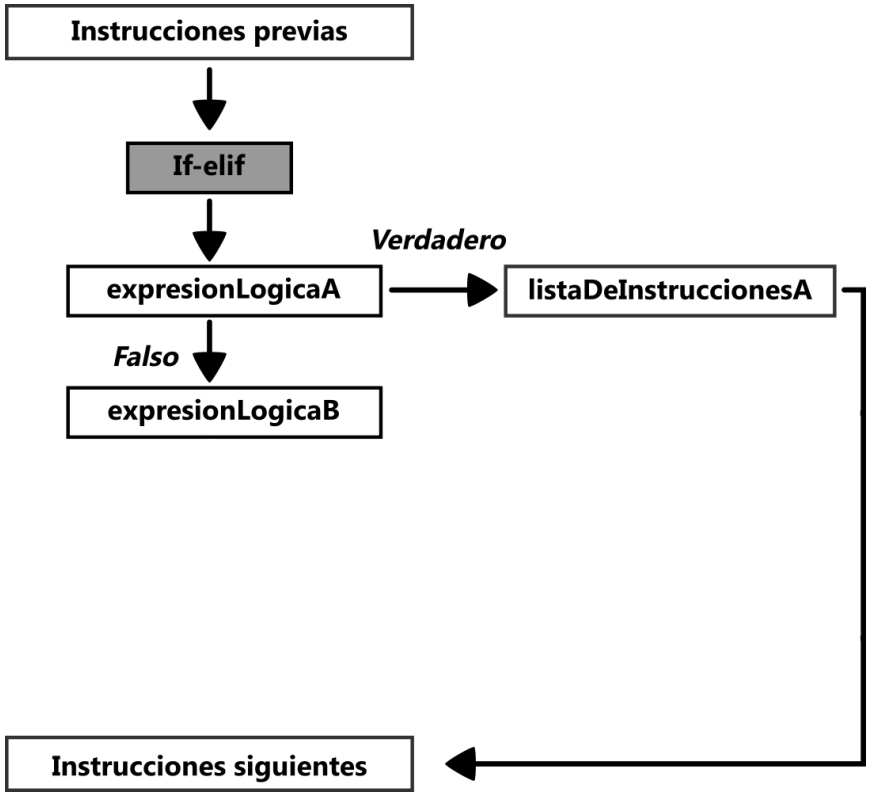
Estructura `if-elif`

```
[instrucciones
anteriores]
if expresion_logica_A:
    instruciones_A
 elif expresion_logica_B:
    instruciones_B
[siguientes
instrucciones]
...
```



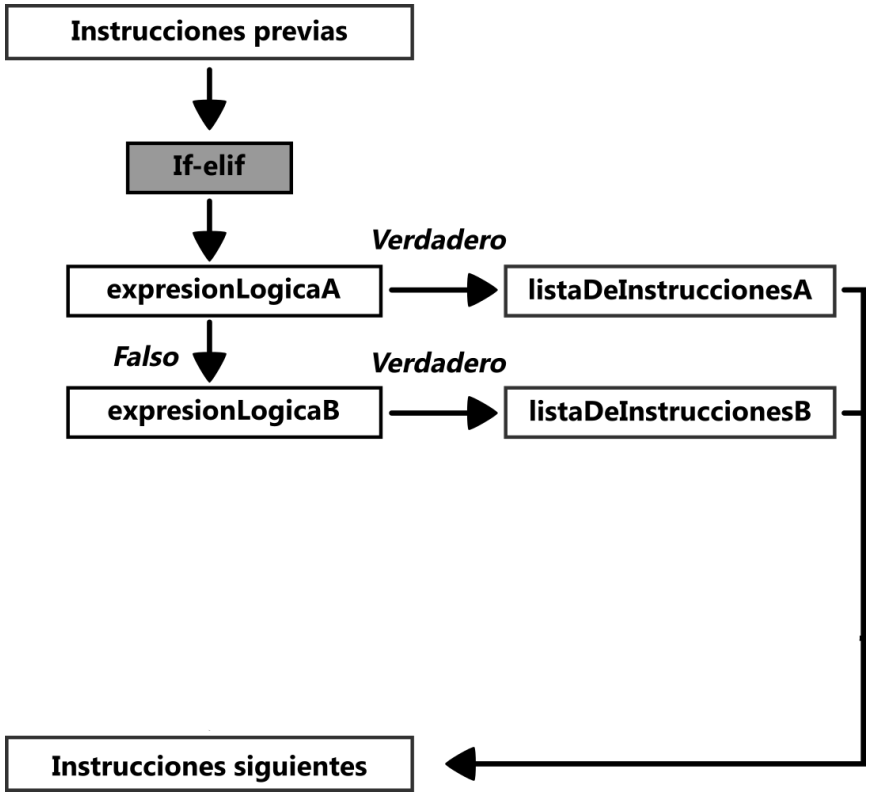
Estructura `if-elif`

```
[instrucciones
anteriores]
if expresion_logica_A:
    instruciones_A
 elif expresion_logica_B:
    instruciones_B
[siguientes
instrucciones]
...
```



Estructura `if-elif`

```
[instrucciones
anteriores]
if expresion_logica_A:
    instruciones_A
 elif expresion_logica_B:
    instruciones_B
[siguientes
instrucciones]
...
```



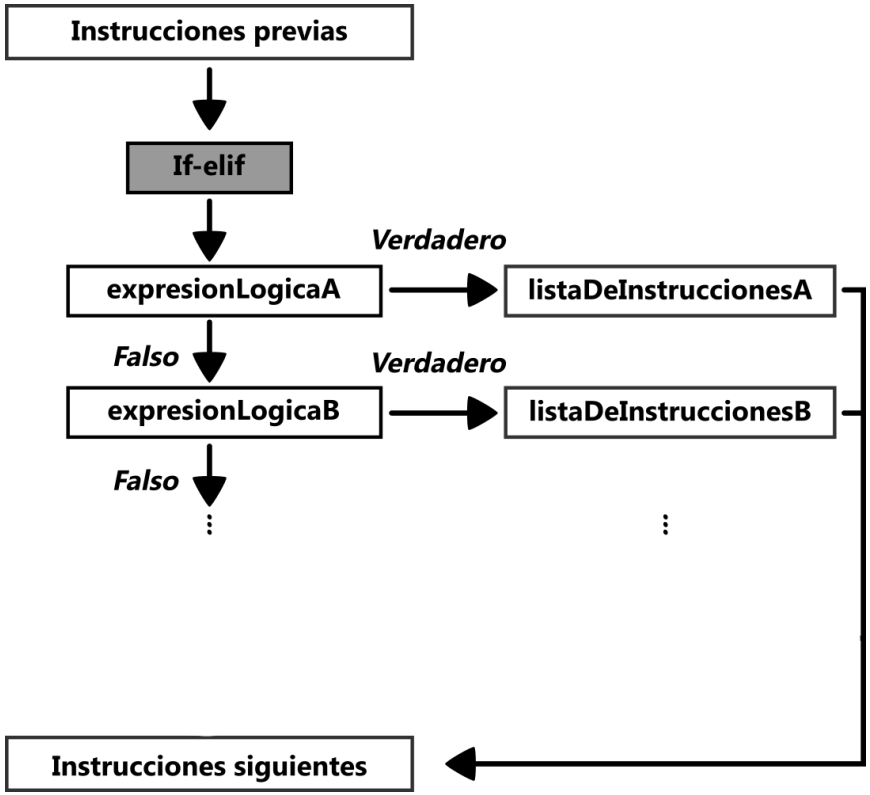
Estructura `if-elif`

- [instrucciones anteriores]

```
if expresion_logica_A:
    instruciones_A
elif expresion_logica_B:
    instruciones_B
```

[siguientes instrucciones]

- ...



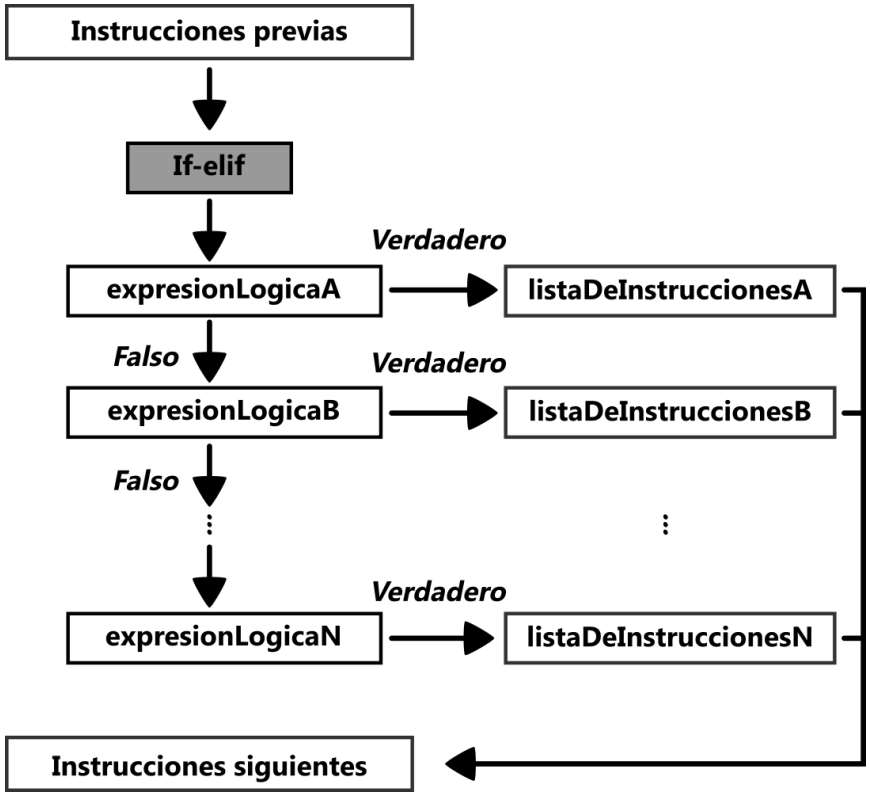
Estructura `if-elif`

- [instrucciones anteriores]

```
if expresion_logica_A:
    instruciones_A
elif expresion_logica_B:
    instruciones_B
```

[siguientes instrucciones]

- ...



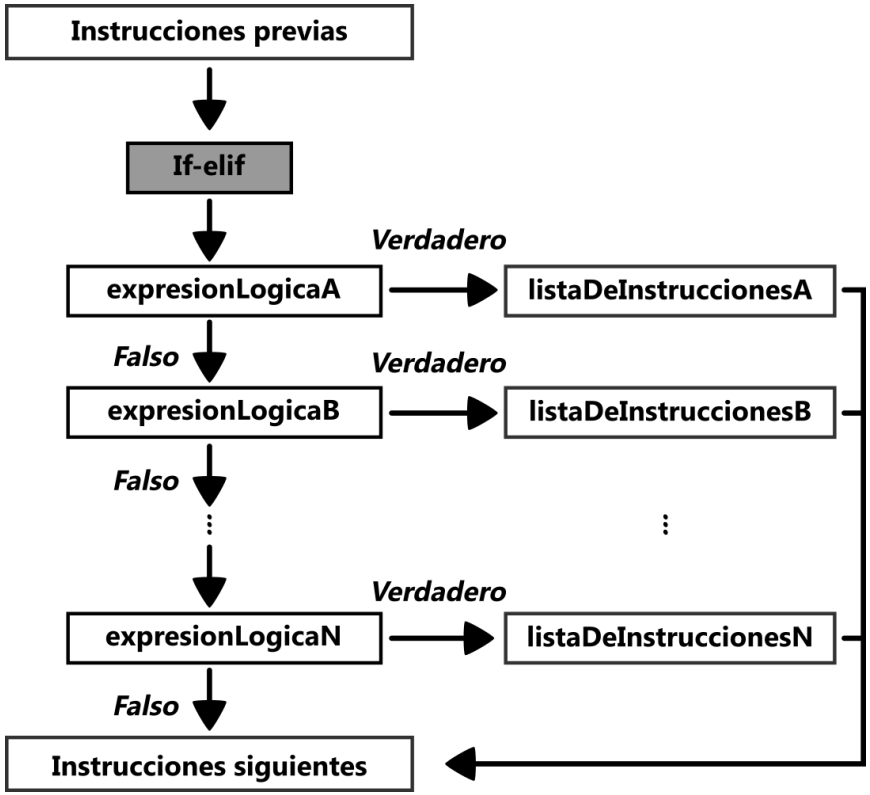
Estructura `if-elif`

- [instrucciones anteriores]

```
if expresion_logica_A:
    instruciones_A
elif expresion_logica_B:
    instruciones_B
```

[siguientes instrucciones]

- ...



Estructura if-elif-else

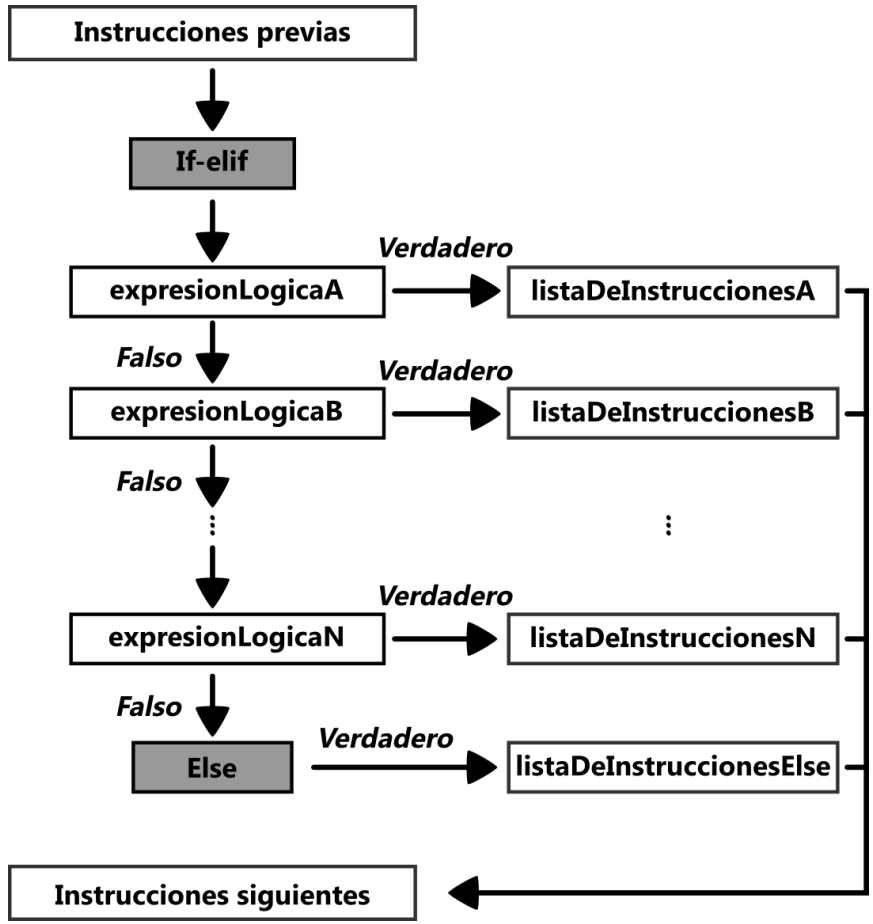
- if-elif-else permite seleccionar una instrucción de un grupo de instrucciones.

☞
```
if expresion_logica_A::
    instruciones_A
 elif expresion_logica_B::
    instruciones_A
...
else:
    instrucciones_else
```

- Se ejecuta unicamente el bloque correspondiente a la primera expresión lógica que se cumpla.
  - Si ninguna se cumple se saltan todos los bloques y se ejecuta el bloque instrucciones_else.

Estructura if-elif-else

```
if expresion_logica_A::
    instruciones_A
elif expresion_logica_B::
    instruciones_A
...
else:
    instrucciones_else
```



Estructura `if-elif-else`

P: Entonces, ¿qué se puede poner en `expresion_logica`?

R: Lo mismo que en la estructura `if` o`elif`, cualquier cosa que devuelva True/False.

P: Y, ¿qué se puede poner en el `listado de instrucciones`?

- Cualquier cosa, incluyendo las estructuras `if`, o`if-else`
- Eso sí, todas las líneas necesitan la misma indentación.

Ejercicio 4

- **3. Ejercicio** Cree una calculadora en Python que realice sumas, restas, multiplicaciones y divisiones entre números enteros. Solicite la información al usuario para realizar los cálculos. Utilice la estructura `if-elif-else` para una mejor legibilidad.

Pidiendo información: Control de operador

- En la variable `operador` necesitaremos información sobre el operador
- Para hacer la calculadora solo necesitamos aceptar dos operaciones, así que...

```
1 operador = input('Escribe un operador: ')
2 if operador == '+':
3   #Realizar suma
4 elif operador == '-':
5   # Realizar resta
6 elif operador == '*':
7   # Realizar multiplicacion
8 else:
9   # no has introducido un operador valido
```

Código: código para que funcione la calculadora.

Ejercicio 5

- **Ejercicio 5** Cree una calculadora en Python que realice sumas, restas, multiplicaciones y divisiones entre números enteros. Solicite la información al usuario para realizar los cálculos. **Asegurarse de que la información proporcionada por el usuario sea útil.**

Pidiendo información: Errores con los tipos de las variables

- Si el usuario escribe información que no es un número entero, por ejemplo `'6.5'`, estas líneas de código darán un error

```
1 numero_1 = int(numero_1)
```

Código: código para que funcione la calculadora.

- Entonces, ¿cómo evitar estos errores?
- Se puede utilizar `str.isdigit (variable)`
  - donde `variable` sea una variable de tipo `str`
  - Si la cadena de carácteres solo tiene dígitos devolverá `True`.

```
1 numero_1 = input('Escribe un numero')
2 if str.isdigit(numero_1):
3   numero_1 = int(numero_1)
4 else:
5   print('No he hecho nada')
```

Código: código para que funcione la calculadora.

Funciones para gestión de texto

- Preguntas
  - `str.isdecimal(texto)`
  - `str.isalpha(texto)`
  - `str.isalnum(texto)`
  - `str.isupper(texto)`
  - `str.islower(texto)`
- Otras funciones sobre texto
  - `str.count(texto,texto_a_contar)`
  - `str.upper(texto)`
  - `str.lower(texto)`

Resumen

Resumen de contenidos

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