

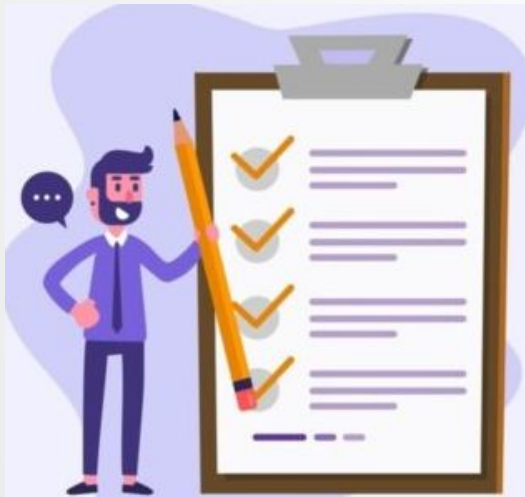


# Python

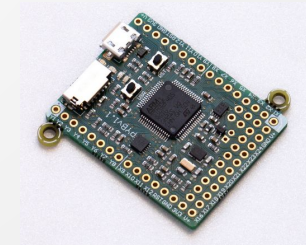
## Código interpretado

# Programación

- Un programa de computadoras es un conjunto de instrucciones paso a paso que le indican a una computadora como realizar una tarea dada.
- Las instrucciones se deben escribir en un lenguaje que nuestra computadora entienda



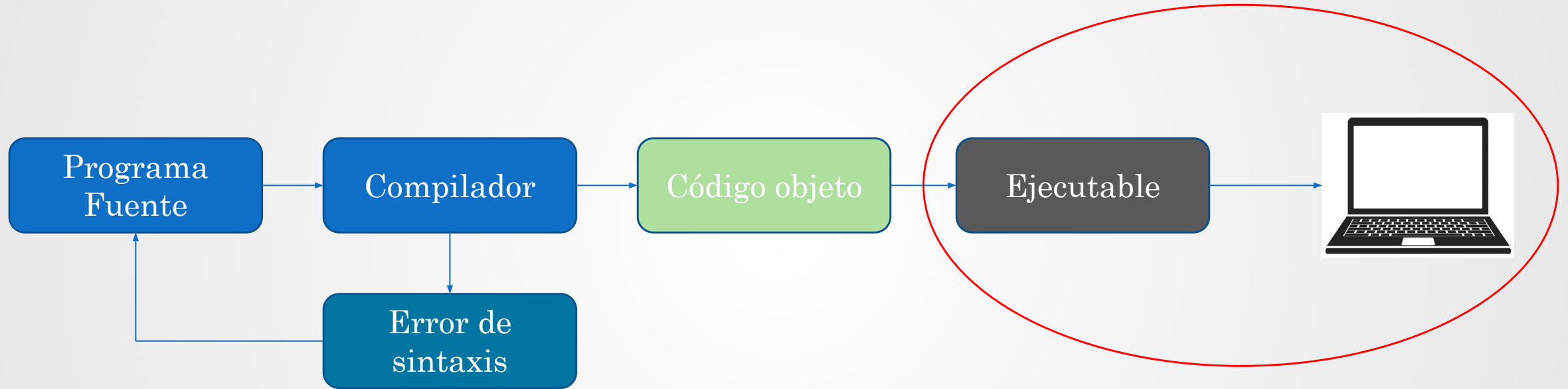
Traductor





# Lenguajes compilados

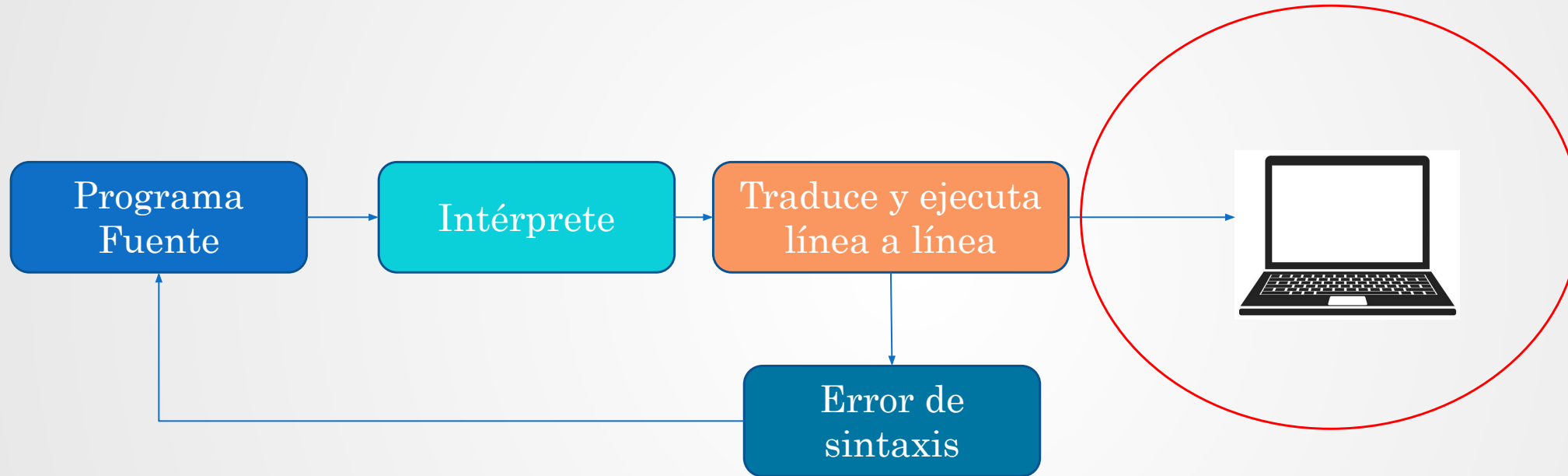
Secuencia de un lenguaje compilado: Ej: C, C++, Java, etc





# Lenguajes interpretados

Secuencia de un lenguaje interpretado: Ej: Python, Ruby, Javascript, etc





## Ventajas y desventajas

- En general, el ciclo de desarrollo (el tiempo entre el momento en que escribes el código y lo pruebas) es mas rápido en un **lenguaje interpretado**.
- Los lenguajes **compilados** solo generan código ejecutable para el sistema operativo donde fue compilado.
- Sin embargo, un lenguaje compilado es más rápido que uno interpretado. Esto se debe a que cuando se ejecuta ya se encuentra el código de máquina generado.
- En general, un lenguaje **compilado** está optimizado para el momento de la ejecución. Por otro lado, un lenguaje interpretado está optimizado para hacerle la vida más fácil al programador.
- Hoy en día la brecha entre estas ramas es cada vez menor. Lenguaje compilados como Go y Rust se inclinan cada vez más en la productividad y felicidad del programador, mientras que los lenguajes interpretados son cada vez más rápidos en ejecución.



# Hola Mundo

El primer programa es el clásico *Hola Mundo*, o sea, vamos a imprimir en pantalla la frase “Hola Mundo”

Creamos un nuevo archivo, en este caso, ejemplo.py. Lo abrimos con el IDE Spyder y escribimos lo siguiente:

```
print (“Hola Mundo”)
```



# Hola Mundo

The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, and Help. Below the menu is a toolbar with various icons for file operations and execution. The main editor window displays a file named 'ejemplo.py' with the following code:

```
1 print ("Hola Mundo")
```

The IPython console on the right shows the execution of the code:

```
In [2]: runfile('C:/Users/pl691d/Google Drive/UNM/Taller Python/ejemplo.py', wdir='C:/Users/pl691d/Google Drive/UNM/Taller Python')
Hola Mundo

In [3]:
```

Observe que en Python no existen los famosos “;” que si podemos encontrar en otros lenguajes, como por ejemplo C, Java o JavaScript.

Por defecto, los programas escritos en Python tienen la extensión .py



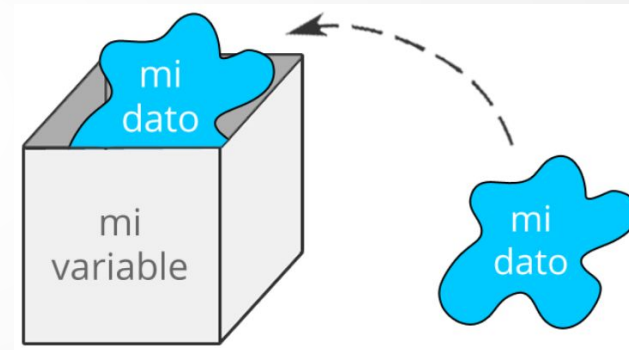
# Variables

Se define como variable al **espacio reservado de la memoria que almacena un dato**, que como su propio nombre indica, puede cambiar de valor en tiempo de ejecución.

Python es un lenguaje de programación de tipado dinámico. Las variables se comprueban en tiempo de ejecución.

**Declaración de una variable:**

**Nombre\_de\_la\_variable** = *dato*



**Una buena práctica consiste en crear variables con nombres intuitivos referentes al dato que almacenan.**





# Tipos de datos

- **Numéricos:** Pueden ser enteros (int) o reales (float).
- **Booleanos:** Se utilizan para representar verdadero (True) o falso (False).
- **Cadenas (string):** Es una secuencia de caracteres para formar una palabra o frase. Se delimita entre comillas simples o dobles.
- **Listas:** Es una colección de objetos: datos numéricos, cadenas, etc. Se delimita utilizando [ ] y sus elementos se separan por comas.
- **Tuplas:** Es como una lista, pero contiene una colección de objetos de distinto tipo.
- **Diccionario:** Se compone de dos partes: una llave (key) y un valor (value). La llave y el valor se separan con : y sus elementos con comas.



# Variables y tipos de datos

```
numero = 2019  
  
decimal = 3.14  
  
booleano = True  
  
cadena = 'Hola desde Python'  
  
lista = ['x', 'y', 'z']  
  
tupla = (3, 'hola', False)  
  
diccionario = {'dia': 'Lunes', 'fecha': 1}
```



# Imprimir contenido de las variables

```
Spyder (Python 3.6)
```

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\pl691d\Google Drive\UNM\Taller Python\ej... x

ejemplo.py x

```
1 a = 10
2 b = 15.5
3 c = "soy un string"
4
5 print (a)
6 print (b)
7 print (c)
8 print (a + b)
```

IPython console x

Console 1/A x

```
In [9]: runfile('C:/Users/pl691d/Google Drive/UNM/
Taller Python/ejemplo.py', wdir='C:/Users/pl691d/
Google Drive/UNM/Taller Python')
10
15.5
soy un string
25.5

In [10]:
```



## Operadores aritméticos

Operación	Símbolo	Resultado
Suma	+	Realiza una suma entre números o concatenación entre string
Resta	-	Realiza la resta entre números
División	/	División real
División	//	División entera (ej: 6 // 4 produce el valor 1)
Multiplicación	*	Multiplicación
Potencia	**	Exponenciación (ej: x**3 significa $x^3$ )
Resto	%	Resto entre una división entera



# Imprimir contenido de las variables

The screenshot shows the Spyder Python IDE interface. The left pane contains a Python script named `ejemplo.py` with the following code:

```
1 dato1 = "Hola"
2 dato2 = "soy un string"
3
4 print (dato1)
5 print (dato2)
6
7 dato3 = dato1 + dato2
8 print (dato3)
9
10 dato3 = dato1 + " " + dato2
11 print (dato3)
```

The right pane shows the IPython console output for two execution steps:

```
In [17]: runfile('C:/Users/pl691d/Google Drive/UNM/Taller Python/ejemplo.py', wdir='C:/Users/pl691d/Google Drive/UNM/Taller Python')
Hola
soy un string
Holasoy un string
Hola soy un string

In [18]:
```





# Imprimir contenido de las variables

The screenshot shows the Spyder Python IDE interface. The left pane displays a file named `ejemplo.py` with the following code:

```
1 a = 2
2 b = "soy un string"
3
4 print (b + a)
```

The right pane shows the IPython console output, which includes the traceback of the error:

```
File "C:\Anaconda3\lib\site-packages\spyder\utils\site\sitecustomize.py", line 705, in runfile
    execfile(filename, namespace)

File "C:\Anaconda3\lib\site-packages\spyder\utils\site\sitecustomize.py", line 102, in execfile
    exec(compile(f.read(), filename, 'exec'), namespace)

File "C:/Users/pl691d/Google Drive/UNM/Taller Python/ejemplo.py", line 4, in <module>
    print (b + a)

TypeError: must be str, not int
```

A red arrow points from the text below to the `print (b + a)` line in the code editor.

**Error de formato. No se puede concatenar un string con un entero. Debemos castearlo primero**



# Imprimir contenido de las variables

The screenshot shows the Spyder Python IDE interface. The editor window on the left contains a file named `ejemplo.py` with the following code:

```
1 a = 2
2 b = "soy un string"
3
4 print (b + str(a))
```

A red arrow points from the text "Casteo de un entero a string" below to the `str(a)` part of the fourth line. The IPython console on the right shows the output of running the script:

```
In [22]: runfile('C:/Users/pl691d/Google Drive/UNM/Taller Python/ejemplo.py', wdir='C:/Users/pl691d/Google Drive/UNM/Taller Python')
soy un string2

In [23]:
```

**Casteo de un entero a string**



# Imprimir con formato

```
edad = 36
nombre = "Pablo"

print ("Mi nombre es %s y tengo %d años." %(nombre, edad))
```

Esto imprime en pantalla:

***Mi nombre es Pablo y tengo 36 años.***

## Indicadores de formato

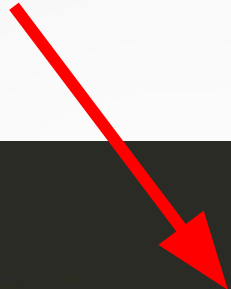
%d	<u>Int</u>
%f	<u>Float</u>
%s	Cadena





# Imprimir con formato

También podemos utilizar el método **format** para mostrar el contenido de las variables



```
1 edad = 37
2 nombre = 'Pablo'
3
4 print('Mi nombre es {} y tengo {} años'.format(nombre, edad))
5
```

```
In [2]: runfile('C:/Users/pl691d/Google Drive/Python/codigos va
Python/codigos varios')
Mi nombre es Pablo y tengo 37 años

In [3]:
```



## Ingreso de datos por Teclado

La función que realiza esta tarea es: **input()**. La computadora recibe algo a través del teclado y por defecto lo interpreta como texto (string).

Si lo que introduce el usuario corresponde a un número, tendremos que indicarle a la función que tiene que transformarlo a número.

Para transformar el dato a un tipo entero, debemos utilizar la función **int()**. Para transformar el dato a float (formato decimal), utilizar **float ()**.

```
Ejemplo.py ✕
algo = input("Introduce algo por el teclado: ")
numero = int(input("Introduce un número: "))
decimal = float(input("Introduce un decimal: "))
```



# Comentarios

En programación, los comentarios son añadidos con el propósito de hacer legible el código que estamos programando y son ignorados por el interprete de programación.

```
# Comentario de una sola línea

"""
Comentario en Python con
múltiples líneas en el
código
"""
```



# Ejercicios

1. Escribir un programa que pregunte al usuario.
  1. Su nombre, y luego lo salude.
  2. Dos números, y luego muestre el producto.
2. Escribir un programa que realice lo siguiente:
  1. Ingresar la base y la altura de un rectángulo y calcular el perímetro.
  2. Calcular la superficie de una circunferencia dado su radio.
  3. Dados los catetos de un triángulo rectángulo, calcular su hipotenusa.