# PROGRAMACIÓN Y BASE DE DATOS


**Servicio Nacional de Derechos Intelectuales (Senadi)**
**GYE - 0097866**
SERVICIO NACIONAL DE DERECHOS INTELECTUALES

**Autor**
© Ediciones Espinosa

**Dirección Editorial**
© Ediciones Espinosa

**Diseño e Ilustración**
© Ediciones Espinosa

**Nueva Edición**
Año 2025

**ISBN**
978-9942-35-693-2

**Guayaquil - Ecuador**

EDICIONES ESPINOSA

**Lo mejor de enseñar es aprender**

# ÍNDICE

## UNIDAD 1 – PROGRAMACIÓN ORIENTADA A OBJETOS (POO)



## UNIDAD 2 – ESTRUCTURAS DE CONTROL

## UNIDAD 3 – ESTRUCTURAS DE CONTROL

# Unidad 1

## Programación Orientada a Objetos (POO)

## Programación Orientada a Objetos (POO)

La Programación Orientada a Objetos (**POO**) se define como un paradigma de programación, lo que implica que constituye un modelo o estilo de programación que proporciona directrices sobre su aplicación.

**Ejemplo**: imagina que quieres construir una casa. En lugar de pensar en cada cemento, bloque, clavo, etc; por separado, la POO te permite pensar en objetos más grandes, como paredes, puertas y ventanas. Cada objeto tiene sus propias características y comportamientos, y puedes combinarlos para crear la casa completa.

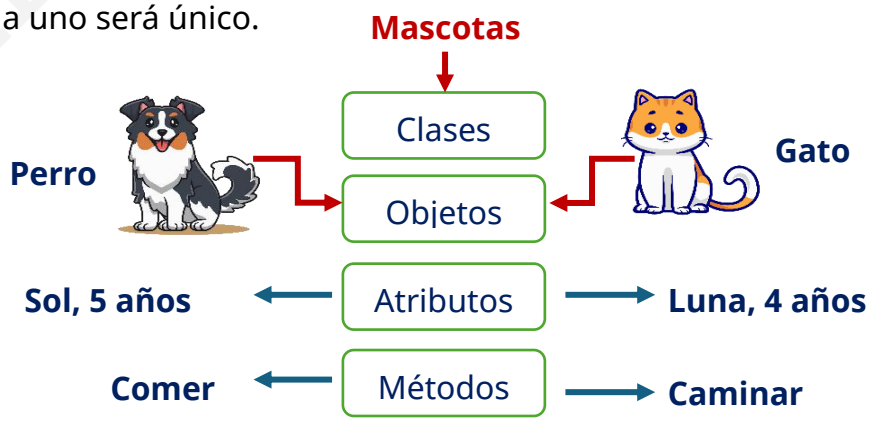
En programación, un objeto es como una "cosa" que tiene datos (características) y acciones (comportamientos). Por ejemplo, un objeto "vehículo" podría tener datos como "**color**", "**marca**" y "**velocidad**", y acciones como "***acelerar***", "***frenar***" y "***girar***".

### Clases, objetos e instancias

Básicamente, en POO se crea **clases** y luego haces **objetos** a partir de esas clases, que son como el plano, y los objetos son las versiones reales de ese plano. La clave aquí es entender la diferencia entre una **clase** y un **objeto**.

Una clase es como un molde genérico que define cómo van a ser los objetos de un cierto tipo. Por ejemplo, puedes tener una clase llamada "**mascotas**" con atributos tipo "nombre" y "edad" (los detalles de la mascota) y con comportamientos tipo "caminar" o "comer" (lo que la mascota puede hacer). Estos comportamientos se implementan como métodos, o sea, funciones de la clase.

Ahora, un objeto sería la mascota real que creas a partir de esa clase. O sea, tiene su edad y puede envejecer, por lo que necesitas un método para eso. Así que, al final, una clase es una definición general de un montón de objetos que van a compartir los mismos atributos y comportamientos, pero cada uno será único.



Con la clase se pueden crear instancias de un objeto, cada uno de ellos con sus atributos definidos de forma independiente. Con esto podríamos crear un perro llamado Sol de 5 años y otra mascota de tipo gato llamado Luna de 4 años. Los dos están definidos por la clase mascota, pero son dos instancias distintas. Por lo tanto, llamar a sus métodos puede tener resultados diferentes. Los dos comparten la lógica, pero cada uno tiene su estado de forma independiente.

## Beneficios de Programación Orientada a Objetos

- Reutilización del código.
- Evita la duplicación de código.
- Convierte cosas complejas en estructuras simples reproducibles.
- Al estar la clase bien estructurada permite la corrección de errores en varios lugares del código.
- La abstracción nos permite construir sistemas más complejos y de una forma más sencilla y organizada.
- Protege la información a través de la encapsulación, dado que solo se puede acceder a los datos del objeto a través de propiedades y métodos privados.
- Permite trabajar en equipo gracias al encapsulamiento porque minimiza la posibilidad de duplicar funciones cuando varias personas trabajan sobre un mismo objeto al mismo tiempo.

## Origen de la Programación Orientada a Objetos (POO)

La Programación Orientada a Objetos (POO) surgió con Simula 67, un lenguaje diseñado por Ole-Johan Dahl y Kristen Nygaard en el Centro de Cómputo Noruego en Oslo, para realizar simulaciones complejas. La necesidad de gestionar las interacciones entre diferentes tipos de naves llevó a la creación del concepto de clases de objetos, donde cada clase definía sus propios datos y comportamientos, sentando las bases de la POO.

A mediados de los años 80, la POO se consolidó como el paradigma dominante gracias al éxito de C++, una extensión orientada a objetos de C. Este paradigma también demostró ser muy efectivo en el desarrollo de **interfaces gráficas de usuario (GUI)**, que requerían programación dirigida por eventos.

Durante ese tiempo, muchos lenguajes como: Ada, BASIC, Lisp y Pascal incorporaron características de POO, pero a menudo enfrentaron problemas de compatibilidad. Para superar estos desafíos, surgieron nuevos lenguajes que combinaban POO con características imperativas de forma más segura, como Eiffel.

A medida que internet se expandía, Java ganó popularidad debido a su máquina virtual, que facilitaba el desarrollo de aplicaciones distribuidas. En este contexto, Python también se destacó como un lenguaje moderno que adoptó completamente la POO, permitiendo una sintaxis simple y flexible para la creación de clases, objetos y herencia, lo que lo convirtió en uno de los lenguajes más utilizados en la actualidad.

# TRABAJO AUTÓNOMO 1

Selecciona la respuesta correcta con base a lo estudiado en clase.

1. Si tienes una clase llamada 'vehículo', ¿qué tipo de datos podrías incluir como atributos?
   A) Aceleración y freno.
   B) Ruedas y motor.
   **C) Color, marca y velocidad.**
   D) Conductor y pasajeros.

2. ¿Cómo se relacionan las instancias de objetos creados a partir de la misma clase?
   A) No pueden tener métodos diferentes.
   B) No pueden ser creadas al mismo tiempo.
   **C) Comparten la misma lógica pero tienen estados independientes.**
   D) Son idénticas en todos los aspectos.

3. ¿Cuál es un beneficio de usar la Programación Orientada a Objetos?
   A) Complica la estructura del código.
   B) Aumenta la duplicación de código.
   **C) Facilita la corrección de errores en el código.**
   D) Elimina la necesidad de clases.

4. ¿Qué significa encapsulación en la Programación Orientada a Objetos?
   A) Es la eliminación de atributos de una clase.
   **B) Es el acceso a datos a través de métodos privados.**
   C) Es la creación de múltiples clases.
   D) Es la creación de instancias de un objeto.

5. ¿Cuál fue uno de los primeros lenguajes en implementar la Programación Orientada a Objetos?
   A) C++.
   B) Python.
   **C) Simula 67.**
   D) Java.

## Elementos de la programación orientada a objetos

### 01 Clases y objetos

- **Clase:** Es como un molde o plantilla que define las características y comportamientos de un tipo de objeto. Por ejemplo, la clase "**Vehículo**" define las características (color, marca, modelo) y comportamientos (acelerar, frenar, girar) de todos los **Vehículos**.
- **Objeto:** Es una instancia específica de una clase. Es como un **Vehículo** concreto que existe en el mundo real. Por ejemplo, un **Vehículo** rojo marca "Toyota" modelo "Corolla" es un objeto de la clase **Vehículo**.

### 02 Atributos y Métodos

- **Atributos**: Son las características o propiedades de un objeto. Por ejemplo, el color, la marca y el modelo son atributos del objeto **Vehículo**.
- **Métodos**: Son las acciones o comportamientos que un objeto (vehículo) puede realizar. Por ejemplo, acelerar, frenar y girar son métodos del objeto **Vehículo**.

### 03 Encapsulamiento

Es la capacidad de proteger los datos de un objeto, evitando que sean modificados por otros objetos de forma directa. Es como tener una caja fuerte para guardar tus objetos valiosos. Solo se puede acceder a los datos a través de métodos específicos, lo que garantiza la integridad de la información.

### 04 Herencia

Es la capacidad de crear nuevas clases a partir de clases existentes, heredando sus atributos y métodos. Es como tener un hijo que hereda los ojos y el color del cabello de sus padres. La clase Vehículo Deportivo podría heredar de la clase **Vehículo** y añadir atributos y métodos específicos, como turbo y Acelerar Muy Rápido.

### 05 Polimorfismo

Es la capacidad de un objeto de tomar diferentes formas o comportamientos, dependiendo del contexto. Es como un **camaleón** que puede cambiar de color para adaptarse a su entorno. Un método "**mover**" podría comportarse de forma diferente si se aplica a un objeto "Vehículo" o a un objeto "Barco".

### 06 Abstracción

Es la capacidad de representar las características esenciales de un objeto, ignorando los detalles innecesarios. Es como ver un **mapa** de una ciudad: te muestra las calles principales y los puntos de interés, pero no te muestra cada edificio individual. La abstracción nos permite trabajar con objetos de forma más sencilla, sin tener que preocuparnos por los detalles técnicos.

## Características de la POO

La POO se basa en cuatro pilares fundamentales:

- **Abstracción**: Es la capacidad de representar las **características** esenciales de un objeto, ignorando los detalles innecesarios. Es como ver un mapa de una ciudad: te muestra las calles principales y los puntos de interés, pero no te muestra cada edificio individual.

- **Encapsulamiento**: Es la capacidad de proteger los datos de un **objeto**, evitando que sean modificados por otros objetos de forma directa. Es como tener una caja fuerte para guardar tus objetos valiosos.

- **Herencia**: Es la capacidad de crear nuevos objetos a partir de objetos existentes, **heredando** sus datos y acciones. Es como tener un hijo que hereda los ojos y el pelo de sus padres.

- **Polimorfismo**: Es la capacidad de un objeto de tomar diferentes formas o **comportamientos**, dependiendo del contexto. Es como un camaleón que puede cambiar de color para adaptarse a su entorno.

## Propiedades de la POO

Además de sus características, la **POO** tiene propiedades que la hacen muy útil:

- **Modularidad**: La **POO** permite dividir un programa en módulos más pequeños e independientes, lo que facilita su desarrollo y mantenimiento. Es como construir una casa por partes: primero las paredes, luego el techo, luego las ventanas, etc.

- **Reusabilidad**: Los objetos se pueden reutilizar en diferentes partes del programa o en otros programas, lo que ahorra tiempo y esfuerzo. Es como usar una puerta que ya tienes para construir una nueva casa.

- **Extensibilidad**: La **POO** facilita la adición de nuevas características a un programa sin tener que modificar el código existente. Es como agregar una nueva habitación a una casa sin tener que derribar paredes.

# AULA INVERTIDA

Escanea el código QR para ver el video y responde cinco preguntas que te ayudarán para la siguiente clase.

1. ¿Cómo ayuda la encapsulación en la programación orientada a objetos?
   A) Elimina la necesidad de herencia.
   B) Facilita la duplicación de código.
   **C) Oculta información innecesaria para el usuario.**
   D) Permite que todos los datos sean públicos.

2. ¿Cuál es un beneficio del polimorfismo en la programación orientada a objetos?
   A) Complica la estructura del programa.
   **B) Permite que un objeto adopte múltiples formas.**
   C) Evita la reutilización de código.
   D) Reduce la legibilidad del código.

3. ¿Por qué es importante la fiabilidad en la programación orientada a objetos?
   **A) Facilita el análisis de cada módulo.**
   B) Elimina la necesidad de documentación.
   C) Permite la creación de programas más complejos.
   D) Reduce la necesidad de pruebas.

4. ¿Qué desventaja se menciona sobre la programación orientada a objetos?
   **A) Necesita una extensa documentación.**
   B) No permite la creación de módulos.
   C) Es más fácil de entender para los nuevos programadores.
   D) Requiere menos tiempo de desarrollo.

5. ¿Cómo afecta la programación orientada a objetos al desarrollo de software?
   **A) Facilita el desarrollo y la comprensión del código.**
   B) Elimina la necesidad de pruebas.
   C) Lo hace más complicado y menos eficiente.
   D) Aumenta el tiempo de desarrollo.

## Ventajas de la programación orientada a objetos.

**Reutilización de código**:

1. Gracias a la herencia, las clases existentes pueden servir como base para nuevas clases, lo que ahorra tiempo y esfuerzo.

**Mantenimiento fácil**:

2. El encapsulamiento ayuda a aislar cambios en el código, porque los objetos controlan cómo se accede a sus datos. Si es necesario modificar algo, los cambios pueden realizarse en una clase sin afectar al resto del sistema.

**Flexibilidad y extensibilidad**:

3. El polimorfismo y la herencia permiten que los programas sean flexibles y puedan ser extendidos con nuevas funcionalidades sin afectar el código existente.

**Modularidad**:

4. El código se organiza en clases, lo que facilita la división del trabajo y permite que diferentes equipos trabajen en distintas partes del sistema sin interferir entre ellos.

**Mejor gestión de proyectos grandes**:

5. POO es ideal para proyectos de software a gran escala, dado que los objetos pueden ser modelados para representar entidades del mundo real, facilitando el diseño y desarrollo.

**Fomenta la prueba y el debugging (solucionar errores en el código fuente)**:

6. El diseño modular facilita las pruebas unitarias de los objetos individualmente, mejorando la calidad del software y facilitando la detección de errores.

## Lenguajes de programación orientados a objetos

**Python:** Uno de los lenguajes más populares y versátiles. Soporta plenamente la POO y es conocido por su facilidad de uso y su sintaxis limpia.

**Java:** Un lenguaje puramente orientado a objetos (excepto por los tipos primitivos) que es ampliamente utilizado en aplicaciones empresariales, desarrollo web, y sistemas Android.

**C++:** Un lenguaje que combina características de programación estructurada y POO. Es muy usado en aplicaciones de alto rendimiento, como videojuegos y sistemas operativos.

**C#:** Un lenguaje desarrollado por Microsoft para su plataforma .NET, muy utilizado en aplicaciones de escritorio, desarrollo web y videojuegos (con Unity).

**Ruby:** Un lenguaje de programación dinámico que pone un fuerte énfasis en la simplicidad y la productividad. Utiliza una sintaxis elegante que facilita la lectura y escritura de código.

**PHP** (a partir de la versión 5): Aunque PHP comenzó como un lenguaje de scripting más orientado a lo procedimental, desde la versión 5 soporta plenamente la POO. Es muy utilizado en el desarrollo web.

**Swift:** Desarrollado por Apple para la creación de aplicaciones para iOS y macOS. Es un lenguaje orientado a objetos modernos, con una sintaxis clara y eficiente.


**Objective-C:** Un lenguaje orientado a objetos que fue utilizado por Apple antes de la adopción de Swift, todavía en uso en algunas aplicaciones para iOS y macOS.

# METACOGNICIÓN

¿Cómo la reutilización de código en la POO optimiza el desarrollo de software y qué ejemplos prácticos podrías mencionar?

¿De qué manera el encapsulamiento contribuye al mantenimiento del código y cómo podrías aplicarlo en un proyecto personal?

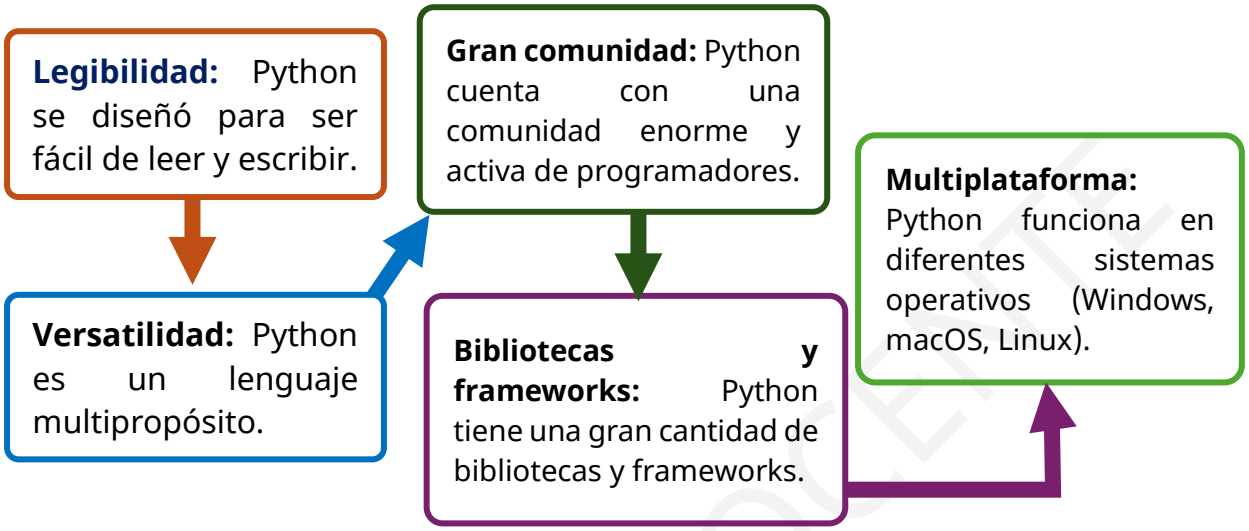
¿Cómo el polimorfismo y la modularidad facilitan la escalabilidad y la gestión de proyectos grandes en comparación con otros paradigmas de programación?

**Califica tus logros siendo 1 la calificación más baja y 4 la más alta.**

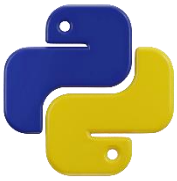
| INDICADORES | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Explica con ejemplos cómo la herencia permite la reutilización de código en la programación orientada a objetos. | | | | |
| Demuestra cómo el encapsulamiento mejora el mantenimiento y la seguridad del código en un proyecto práctico. | | | | |
| Identifica y compara diferentes lenguajes de programación orientados a objetos según sus características y usos principales. | | | | |
| Describe cómo la modularidad y la flexibilidad en la POO facilitan el desarrollo y mantenimiento de software a gran escala. | | | | |

## Lenguaje de programación orientada a objetos en Python

Python es un lenguaje de programación de alto nivel que se utiliza para crear todo tipo de aplicaciones, desde simples scripts hasta complejas aplicaciones web, software de escritorio, herramientas de análisis de datos y es muy popular por:




## Python



Python, cuyo nombre se inspira en la serie de televisión británica "Monty Python's Flying Circus", es un lenguaje de programación de alto nivel ampliamente reconocido por su diseño que prioriza la legibilidad del código.

Python es un lenguaje de programación de alto nivel muy popular porque está diseñado para que el código sea fácil de leer. Su idea principal es que el código sea claro y que programar sea más sencillo. Python también es muy versátil, porque permite programar de diferentes maneras, como con objetos, de forma imperativa y, aunque menos común, de forma funcional.

## Python se utiliza para:

01 **Desarrollo web:** Creación de sitios web y aplicaciones web robustas y escalables.

02 **Automatización de tareas:** Creación de scripts para automatizar tareas repetitivas y aumentar la productividad.

03 **Educación:** Python es un lenguaje popular para enseñar programación a principiantes debido a su facilidad de aprendizaje.

04 **Ciencia de datos:** Análisis de datos, visualización, aprendizaje automático e inteligencia artificial.

05 **Desarrollo de software:** Creación de aplicaciones de escritorio y software para diversas industrias.

## OnlineGDB

Es una plataforma gratuita y abierta que funciona como un laboratorio de programación en línea. Con esta herramienta, puedes escribir, compilar, ejecutar y depurar código en varios lenguajes como Python, Java, C, C++ y más. Es perfecta tanto para novatos como para programadores experimentados, debido a que te ofrece todo lo que necesitas para programar y lo puedes usar desde cualquier lugar con acceso a internet.

### Funcionamiento de OnlineGDB

Funciona en la nube, lo que significa que todo el trabajo se hace en servidores remotos. Solo necesitas un navegador web y conexión a internet para usarlo. Cuando escribes tu código, OnlineGDB lo traduce a un lenguaje que la computadora entiende y lo ejecuta en sus servidores. Después, te muestra los resultados en tu navegador.

### Utilización:

1. **Accede al sitio web:** Abre tu navegador y ve a **www.onlinegdb.com**
2. **Elige tu lenguaje:** Selecciona el lenguaje de programación que vas a utilizar en el menú desplegable.
3. **Escribe tu código:** En el editor de código, escribe tu programa.
4. **Compila y ejecuta:** Haz clic en el botón "Run" (Ejecutar) para compilar y ejecutar tu código.
5. **Observa los resultados:** Los resultados de tu programa se mostrarán en la consola.

# TRABAJO AUTÓNOMO 2

Selecciona la respuesta correcta con base a lo estudiado en clase.

1. ¿Por qué es importante la legibilidad del código en Python?
   A) Reduce la necesidad de comentarios.
   B) Aumenta la complejidad del código.
   C) Limita el uso de bibliotecas externas.
   **D) Facilita la colaboración entre programadores.**

2. ¿Qué paradigma de programación NO es comúnmente asociado con Python?
   A) Programación orientada a objetos.
   B) Programación funcional.
   C) Programación imperativa.
   **D) Programación de ensamblador.**

3. ¿Qué permite hacer la plataforma OnlineGDB?
   A) Instalar software en la computadora.
   **B) Escribir, compilar y ejecutar código en varios lenguajes.**
   C) Crear aplicaciones de escritorio sin conexión a internet.
   D) Programar solo en Python.

4. ¿Qué se necesita para utilizar OnlineGDB?
   A) Un software específico instalado.
   **B) Un navegador web y conexión a internet.**
   C) Un conocimiento avanzado de programación.
   D) Un dispositivo móvil exclusivamente.

5. ¿Cuál es una ventaja de usar OnlineGDB en comparación con la programación tradicional?
   A) Solo se puede usar en un solo lenguaje.
   B) Requiere instalación de software.
   C) No permite colaboración entre programadores.
   **D) Es accesible desde cualquier lugar con internet.**

## Tipos de datos

En **Python**, como en cualquier lenguaje de programación, los tipos de datos son esenciales, definen la naturaleza de los valores que el programa puede manipular y pueden ser:

- **Variables.**
- **Constantes.**
- **Booleanos.**

## Variables

Son los que almacenan valores, además que pueden cambiar durante la ejecución del programa, ya sea porque se los ingresa o se los calcula. En Python, se crean simplemente asignándoles un valor (ej: **edad = 25**).

## Constantes

Son los valores que no cambian durante la ejecución del programa. Aunque **Python** no tiene un concepto estricto de constantes (valores que no se pueden modificar), por convención se usan nombres en mayúsculas para indicar que un valor no debería cambiar (ej: **PI = 3.14159**).

## Clases de variables:

**Alfanuméricos - cadena de caracteres (str).-** Almacena todo tipo de caracteres. Por ejemplo: Nombres, Apellidos, Dirección, Estado Civil, Cargo, es decir, toda una cadena de caracteres.

**Numéricos.-** Almacena sólo números.

- **int,** *son para cantidades enteras sin decimal,* ***ejemplo***: edad, cantidad, días, meses, años, número de hijos, cantidades pequeñas, pero enteras, etc.
- **float,** *son para cantidades enteras con decimales,* ***ejemplo****:* estatura, Iva, subtotal, resultado de una división, cantidades que contengan números con decimales o números enteros grandes.

**Booleanos:** Admite valores lógicos de .T. (verdadero) o .F. (falso). Por ejemplo: varón, activo, status, etc.

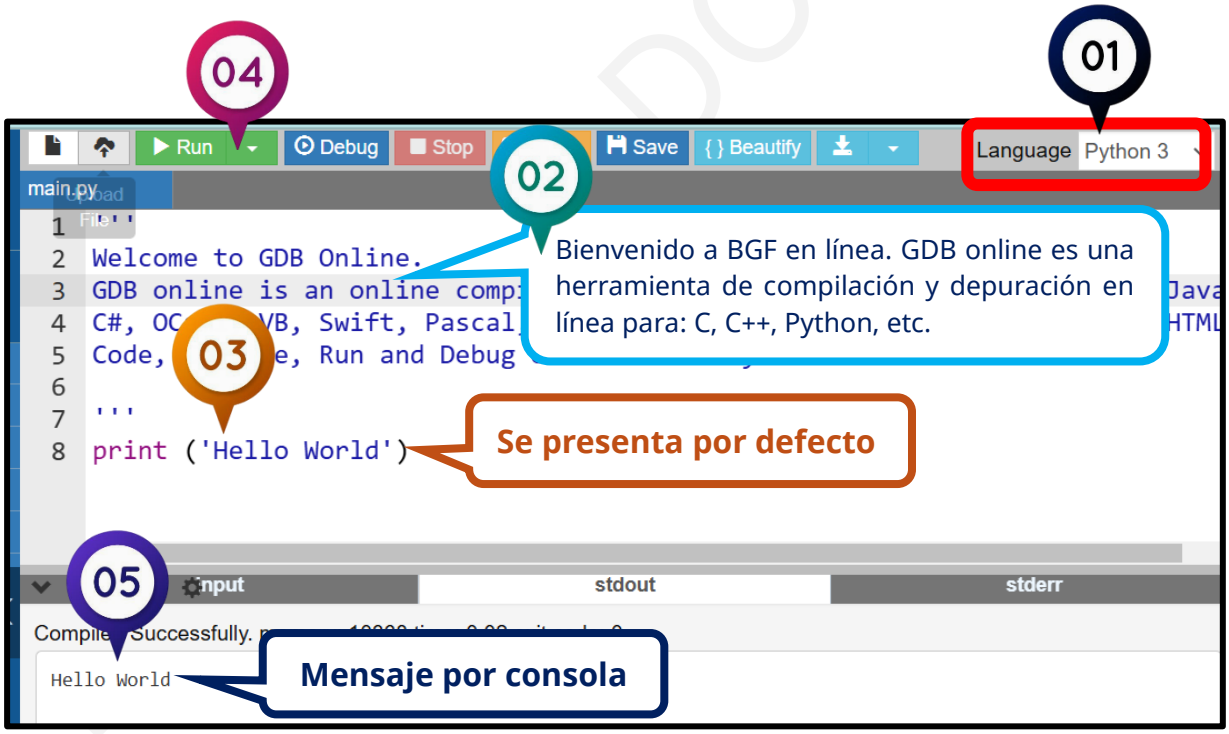
## Tipos de variables

Los principales tipos de variables son:

| Tipo | Rango |
|---|---|
| str | Cadena de caracteres |
| int | Números enteros |
| float | Números enteros con decimales |

Abre tu navegador web (Google Chrome, Firefox o Edge). Escribe la dirección: **www.onlinegdb.com** en la barra de direcciones.


Lo primero que se debe hacer es seleccionar es **Python.**




**Detalle de la ilustración:**

**01, Seleccionar el lenguaje de programación: Python 3.**

**02, Mensaje de bienvenida del programa.**

**03, Se presenta por defecto la siguiente instrucción: print (`'Hello World'`)**

**04, Dé clic en ejecutar.**

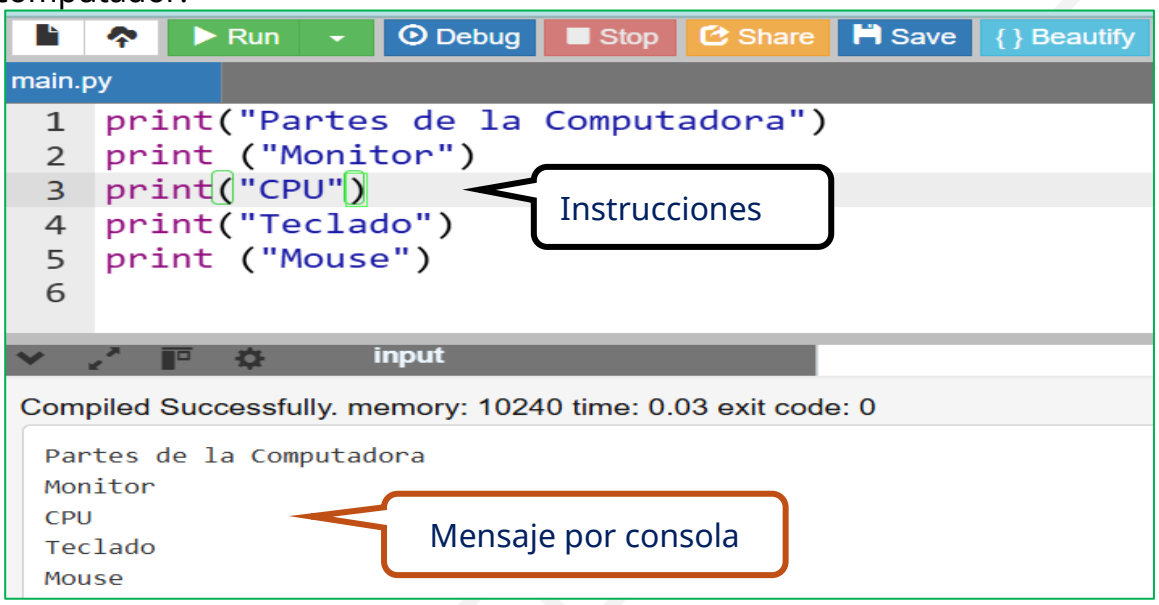
**05, Visualiza un mensaje por consola: `Hello, World`**

**Función print.** Esta función presenta, visualiza, muestra o imprime una expresión. Una expresión puede ser un mensaje, una variable, una función o cualquier combinación de ellas. Si la expresión es un **mensaje** debe ir entre paréntesis y comillas dobles, ejemplo: **("Hello, World")**

**Por cada print que se utilice se presenta una expresión.**

**Nota**: Todas las instrucciones deben ser escritas en letras **minúsculas**.

**Ejercicio 01,** el siguiente programa presenta por consola las partes del computador:




**Ejercicio 02,** a continuación otro programa que presenta los 5 lenguajes de programación más utilizados: