

«Talento Tech»

Iniciación a la

# Programación con Python

Clase 01



# Clase N° 1 | Conceptos básicos

## Temario:

- Introducción al curso: duración, objetivos, metodología, evaluación.
  - Lenguajes de programación
  - Concepto de algoritmo. Entrada-Proceso-Salida.
  - Estrategias para la resolución de problemas.
  - Introducción a Python: Origen, características del lenguaje, campo de aplicación.
  - Instalación y configuración del entorno de desarrollo (Visual Studio Code + Python).
  - Interfaz de Visual Studio Code.
  - "Hola Mundo" en Python.
- 

## Objetivos de la clase

El principal objetivo de esta primera clase es proporcionar a los estudiantes una introducción al curso y al entorno de programación, sentando las bases conceptuales y prácticas para el aprendizaje de Python. Durante la clase, se busca que las y los estudiantes comprendan qué es la programación y su importancia en el contexto actual, además de familiarizarse con el concepto de algoritmos y el modelo Entrada-Proceso-Salida como herramientas fundamentales para resolver problemas.

Adicionalmente, aprenderán a configurar su entorno de trabajo, instalando Python y Visual Studio Code, dos herramientas esenciales que utilizarán a lo largo del curso. Como parte de esta introducción, escribirán y ejecutarán su primer programa, el clásico "Hola, Mundo", para asegurarse de que todo esté correctamente instalado y funcionando. Este ejercicio inicial tiene como objetivo brindarles una primera experiencia práctica con Python, fomentando su confianza y entusiasmo por el aprendizaje.

# ¡Talento Lab te está esperando!



Imaginá que recibís una invitación para participar en el proceso de selección de Talento Lab una startup ubicada en Buenos Aires.

¿Cuál sería el reto? Completar una pasantía de aprendizaje que pondrá a prueba todas tus habilidades y aprendizaje.

A partir de este momento un equipo de expertos te guiarán en este emocionante viaje.

## ¡Te damos la bienvenida a TechLab!

En **TechLab**, nuestra empresa de desarrollo de software, convertimos ideas en herramientas digitales innovadoras y confiables, ofreciendo servicios de desarrollo backend con un enfoque en la calidad y eficiencia. Nuestro compromiso es desarrollar soluciones que optimicen procesos y potencien negocios, combinando creatividad, tecnología de punta y un compromiso absoluto con la satisfacción de nuestras y nuestros clientes. Tu éxito es nuestra prioridad, y nuestras soluciones están diseñadas para marcar la diferencia.

### Te presentamos al equipo de TechLab:



**Luis**  
Desarrollador Senior



**Elena**  
Desarrolladora Junior



**Diego**  
Analista de Datos



**Sofía**  
Especialista en UX/UI



**Mariana**  
Gerente de Proyectos

## ¡Tu primer día en TechLab!



En **TechLab**, recibimos un nuevo proyecto de un cliente que requiere el desarrollo de una aplicación en Python capaz de gestionar información de clientes, productos y pedidos. La solución deberá incluir funcionalidades para registrar, consultar, actualizar y eliminar datos, utilizando una base de datos SQLite integrada con el programa.

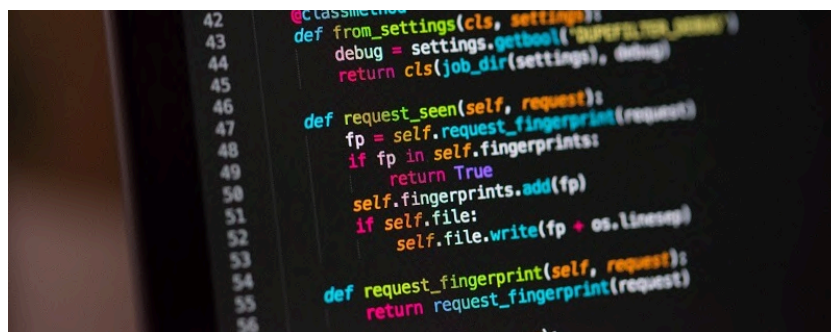
¡Vamos de a poco!

Antes de encarar este nuevo proyecto debemos capacitarte con los conceptos básicos de programación. Leé con atención lo que sigue...



# ¿Qué es la programación?

La programación es un proceso fascinante y esencial en el mundo de la tecnología, definida como la práctica de escribir instrucciones para que una computadora realice tareas específicas. Aunque a primera vista parece ser simplemente la escritura de un código, la programación es mucho más. Representa una manera única de pensar y abordar problemas, transformando ideas abstractas en aplicaciones concretas y funcionales. La programación no sólo se trata de conocer lenguajes de programación y sintaxis; implica también un enfoque lógico y creativo para solucionar problemas, optimizar procesos y crear nuevas posibilidades dentro del mundo digital. Es un campo dinámico que requiere una constante actualización de conocimientos y adaptación a nuevas tecnologías y paradigmas.



```
42 @classmethod
43 def from_settings(cls, settings):
44     debug = settings.getbool('SUPERFINGER_DEBUG')
45     return cls(job_dir(settings), debug)
46
47 def request_seen(self, request):
48     fp = self.request_fingerprint(request)
49     if fp in self.fingerprints:
50         return True
51     self.fingerprints.add(fp)
52     if self.file:
53         self.file.write(fp + os.linesep)
54
55 def request_fingerprint(self, request):
56     return request_fingerprint(request)
```

Código fuente de un programa informático.

El **objetivo principal** de la programación es desarrollar programas que lleven a cabo tareas específicas. Estas tareas pueden variar desde simples operaciones matemáticas hasta la gestión de complejos sistemas operativos y aplicaciones avanzadas.

## Lenguajes de programación

Sin embargo, las computadoras no entienden el lenguaje humano sino que debemos redactar las instrucciones que queramos que realicen de una forma especial. ¿Cómo? Mediante los **lenguajes de programación**. Estos son herramientas fundamentales en el desarrollo de software, actuando como puente entre las y los programadores y las computadoras. Estos lenguajes permiten escribir instrucciones de manera comprensible, que luego son convertidas en código que la computadora puede ejecutar. Existen numerosos lenguajes de programación, cada uno con sus propias reglas de sintaxis y aplicaciones particulares. Estos lenguajes son diseñados para facilitar la creación de programas eficientes y efectivos, abarcando desde operaciones básicas hasta complejas aplicaciones.

Cada lenguaje de programación tiene sus propias fortalezas y se elige en función de las necesidades específicas del proyecto en desarrollo.

Algunos de los lenguajes más populares son:



**Python** es reconocido por su simplicidad y versatilidad, lo que lo convierte en un lenguaje ideal para aquellas personas que se inician en la programación. Su sintaxis clara y directa, junto con una extensa biblioteca de recursos, facilita el desarrollo rápido de aplicaciones complejas. Podés utilizar Python en una amplia gama de aplicaciones, desde el desarrollo web hasta la ciencia de datos y la inteligencia artificial, ofreciendo una gran flexibilidad para abordar diversos proyectos.



**Java**, por otro lado, es un lenguaje ampliamente utilizado en el desarrollo de aplicaciones empresariales y sistemas móviles, especialmente en la plataforma Android. Es valorado por su robustez y portabilidad, así como por su capacidad para manejar grandes cantidades de datos, lo que lo hace ideal para proyectos a gran escala. La fiabilidad y el rendimiento de Java lo convierten en una opción preferida para aplicaciones que requieren una base sólida y escalable.



Los lenguajes **C** y **C++** son conocidos por su eficiencia y su capacidad para interactuar de manera cercana con el hardware. Estos lenguajes se utilizan comúnmente en el desarrollo de sistemas operativos, juegos y otras aplicaciones donde el rendimiento es crítico. La gestión detallada de los recursos del sistema que permiten C y C++ es esencial en situaciones donde la eficiencia y la velocidad son prioritarias.

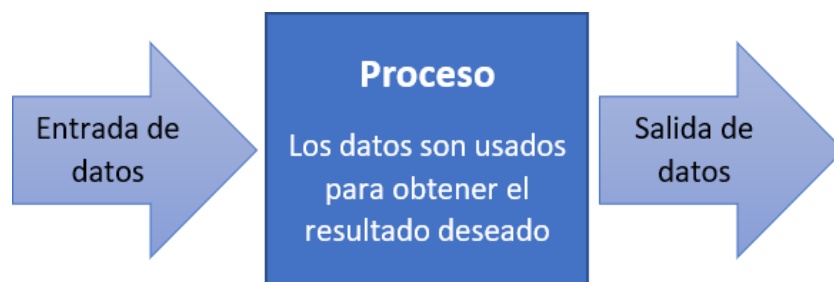
Existen cientos de lenguajes de programación más, algunos muy populares como **JavaScript** o los mencionados anteriormente y otros prácticamente desconocidos. Pero todos cumplen la misma función: permitir la creación de programas que serán ejecutados por una computadora.

## Algoritmos

¿Cómo diseñaremos las instrucciones a seguir por la computadora? Por medio de un **algoritmo**: se trata básicamente de una serie de pasos lógicos y ordenados que permiten resolver un problema o realizar una tarea específica. Podés imaginarlo como una receta de cocina: tenés ingredientes, seguís una secuencia de instrucciones y al final obtenés un plato preparado.

El modelo de **Entrada-Proceso-Salida** nos ayuda a estructurar ese algoritmo de una manera más clara:

1. **Entrada:** Son los datos o información que necesitamos para empezar. Por ejemplo, si queremos calcular el área de un rectángulo, las entradas serían el ancho y el alto.
2. **Proceso:** Es donde ocurre el "trabajo". Aquí aplicamos operaciones o cálculos sobre los datos de entrada. Siguiendo el ejemplo, multiplicarías el ancho por el alto para obtener el área.
3. **Salida:** Es el resultado final después de procesar los datos. En este caso, el área calculada del rectángulo.



Entender este modelo es fundamental porque nos permite planificar y estructurar nuestros programas de manera lógica y eficiente. Cada vez que nos enfrentamos a un problema, podemos descomponerlo en estas tres partes para encontrar la solución más adecuada.

Por ejemplo, cuando en **TechLab** necesitamos procesar datos de clientes, primero identificamos qué información necesitamos (Entrada), luego definimos cómo vamos a procesar esa información (Proceso) y finalmente determinamos qué resultados o reportes queremos obtener (Salida).

En varias ocasiones en TechLab debemos generar aplicaciones que sólo deben poder utilizar personas mayores de edad. ¿Podés imaginar cómo se vería el algoritmo de un programa que resuelva este problema? Veamos cómo aplicamos el modelo de Entrada-Proceso-Salida en este caso.

#### **Entrada:**

- La usuaria o usuario ingresa su edad en años.

#### **Proceso:**

- El programa compara la edad ingresada con el número 18.
- Si la edad es mayor o igual a 18, se considera mayor de edad.
- Si la edad es menor a 18, se considera menor de edad.

**Salida:**

- El programa muestra un mensaje indicando si la persona es mayor o menor de edad.

Ahora, detallando los pasos del algoritmo:

1. Pedir que se ingrese su edad.
2. Capturar y almacenar el valor ingresado.
3. Comparar la edad con 18:
  - Si la edad es mayor o igual a 18:
    - Mostrar el mensaje "Sos mayor de edad."
  - De lo contrario:
    - Mostrar el mensaje "Sos menor de edad."

Este ejemplo es sencillo pero nos sirve para que veas cómo estructuramos un algoritmo. Descomponemos el problema en pasos lógicos que el programa puede seguir para llegar a una solución. Además, este tipo de lógica es muy común en desarrollo de software, ya que frecuentemente necesitamos tomar decisiones basadas en la información que recibimos del usuario o usuaria.

## Características de los algoritmos

- 1. Claridad y precisión:**
  - Un algoritmo debe ser detallado y comprensible, evitando ambigüedades y asegurando la correcta ejecución de cada paso.
- 2. Finitud:**
  - Debe concluir después de un número determinado de pasos, evitando bucles infinitos o procesos sin fin.
- 3. Definición y estructura:**
  - Cada paso y su secuencia deben estar claramente definidos y bien organizados.
- 4. Orden:**
  - El orden de los pasos es crucial, ya que determina la forma en que se ejecutan las instrucciones para alcanzar el resultado deseado de manera eficiente.
- 5. Efectividad:**
  - Un algoritmo debe ser capaz de resolver el problema planteado en un tiempo razonable y con un uso adecuado de recursos.



## 6. Generalidad:

- Un buen algoritmo debe aplicarse a un rango de problemas similares, no solo a un caso específico.

Estas características hacen que los algoritmos sean herramientas poderosas en diversas disciplinas, permitiendo la estructuración lógica de ideas y el desarrollo de soluciones efectivas en programación, análisis de datos, inteligencia artificial y más. Continuaremos trabajando pero, más adelante también, trabajaremos con Luis y el equipo de desarrollo senior de TechLab para ver ejemplos más específicos en torno a la aplicación de los algoritmos... ¡Vamos bien! Continuemos un poco más.

## Representación de algoritmos.

La representación de algoritmos se puede realizar de diversas maneras: cada una adecuada para diferentes etapas del desarrollo o para diferentes públicos. Las tres formas más comunes son los diagramas de flujo, el pseudocódigo y el código fuente.

**Diagramas de flujo:** Son representaciones gráficas de los pasos de un algoritmo. Utilizan símbolos estándar para representar diferentes tipos de instrucciones o acciones, como operaciones de entrada/salida, decisiones y procesos. Son especialmente útiles para visualizar el flujo lógico de un algoritmo y son ideales para explicar procesos complejos de manera sencilla y clara. En el siguiente ejemplo vemos cómo cocinar un huevo se grafica por medio del diagrama:



Diagrama de flujo.

**Pseudocódigo:** Es una forma de escribir algoritmos que utiliza un lenguaje cercano al natural con algunas estructuras de lenguajes de programación. No sigue la sintaxis estricta

de ningún lenguaje específico, pero ayuda a planificar y estructurar el algoritmo de manera clara y organizada. El *pseudocódigo* es un paso intermedio útil entre la formulación del problema y la escritura del código fuente que luego interpretará el computador.

```
1  Algoritmo PAR_IMPAR
2      Escribir 'Por favor, teclea un número'
3      Leer numero
4      Si numero MOD 2=0 Entonces
5          Escribir 'El número, ',numero,' es par'
6      SiNo
7          Escribir 'El número, ',numero,' es impar'
8      FinSi
9  FinAlgoritmo
```

Algoritmo escrito en pseudocódigo



**¡En TechLab te sugerimos que empieces por acá!** Podés experimentar escribiendo algoritmos en sitios como [Gobstones](#)

**Código fuente:** Es la representación de un algoritmo en un lenguaje de programación específico. Aquí, el algoritmo se escribe siguiendo las reglas sintácticas y semánticas del lenguaje elegido, como PSeInt, Python, Java o C++, entre otros. El código fuente es lo que finalmente se ejecuta en una computadora.

```
107 @routes.route('/login', methods=['GET', 'POST'])
108 def login():
109     if request.method == 'POST':
110         nombre = request.form['username']
111         password = request.form['password']
112         user = Usuario.query.filter_by(nombre=nombre).first()
113         if user and check_password_hash(user.password, password):
114             session['username'] = user.nombre
115             session['correo'] = user.correo
116             session['id_usuario'] = user.id_usuario
117             return redirect(url_for('seleccionar_fecha'))
118         else:
119             return render_template('login.html', error='Usuari
120     return render_template('login.html')
121
```

Código fuente escrito en Python.

Cada una de estas representaciones tiene su propósito y su público objetivo. Mientras que los diagramas de flujo y el pseudocódigo son excelentes para la enseñanza y la planificación, el código fuente es necesario para la implementación efectiva de los algoritmos en aplicaciones reales.

# Introducción a Python



**Python** es un lenguaje de programación interpretado de alto nivel, caracterizado por su énfasis en la simplicidad y legibilidad del código. Creado a finales de 1989 por Guido van Rossum en el Centro para las Matemáticas y la Informática (CWI) en los Países Bajos, Python se concibió como un sucesor de un lenguaje anterior llamado ABC, capaz de manejar excepciones y de interactuar con el sistema operativo Amoeba.

La filosofía detrás de Python se centra en dos principios fundamentales: **la legibilidad del código y la simplicidad tanto en la escritura como en la lectura**. Estos principios se reflejan en el diseño del lenguaje, que promueve estructuras de código claras y un estilo de programación que **prioriza la facilidad de comprensión sobre la brevedad del código**. Esta filosofía se encapsula en el "*Zen de Python*", una colección de 19 aforismos que guían el diseño del lenguaje.

## Características claves de Python

Python se destaca por una serie de características que lo hacen accesible y poderoso tanto para quienes recién comienzan a programar como para aquellas personas con más experiencia.

### 1. Sintaxis clara y legible:

- Python tiene una sintaxis que se asemeja al lenguaje natural, lo que facilita la escritura de código intuitivo y comprensible.

### 2. Tipado dinámico:

- Las variables (ya veremos qué son y cómo funcionan) en Python pueden cambiar de tipo automáticamente según el valor asignado, eliminando la necesidad de especificar el tipo de dato al declarar una variable.

### 3. Gestión de memoria automática:

- Python maneja automáticamente la reserva y liberación de memoria, lo que reduce el riesgo de errores como fugas de memoria y simplifica el trabajo del programador.

### 4. Extensa biblioteca estándar:

- Cuenta con una amplia gama de módulos y funciones predefinidas que permiten realizar tareas comunes sin necesidad de escribir el código desde cero, acelerando el desarrollo y garantizando soluciones robustas.

### 5. Lenguaje interpretado:

- El código de Python se ejecuta línea por línea, lo que facilita la depuración y permite detectar y corregir errores en tiempo real.

## 6. Multiplataforma:

- Los programas escritos en Python pueden ejecutarse en diferentes sistemas operativos sin necesidad de modificaciones, lo que le confiere versatilidad y accesibilidad.

Estas características hacen de Python un lenguaje altamente accesible y poderoso, tanto para principiantes como para programadoras y programadores experimentados.

**¡Es el lenguaje de programación más utilizado para el desarrollo de software en TechLab!**

## Evolución y popularidad.

Desde su creación, Python ha experimentado un crecimiento constante en popularidad, impulsado tanto por su facilidad de aprendizaje como por su poderosa funcionalidad. Su uso abarca una amplia gama de aplicaciones, desde desarrollo web y automatización de tareas hasta ciencia de datos, aprendizaje automático e inteligencia artificial. La adopción de Python en la comunidad científica y de investigación ha sido particularmente notable, gracias a su simplicidad y la disponibilidad de numerosas bibliotecas especializadas.

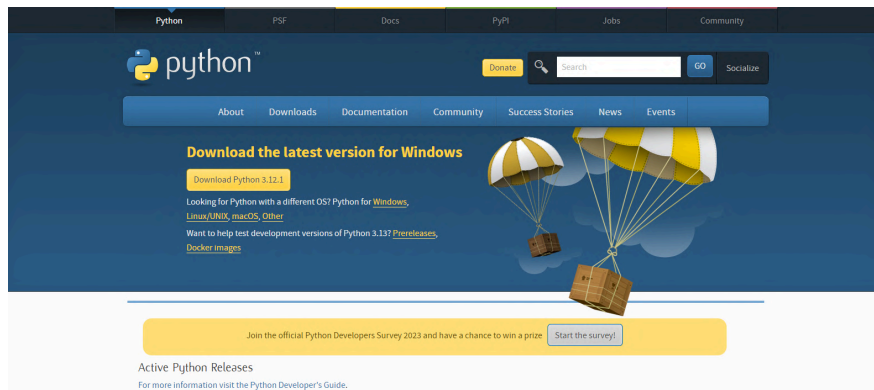
La comunidad de Python es una de las más activas y colaborativas en el mundo del software libre y de código abierto. Con una amplia gama de conferencias, talleres y foros dedicados a Python, las desarrolladoras y desarrolladores tienen muchas oportunidades para aprender y contribuir al ecosistema de Python.

Python se distingue por ser un lenguaje que combina eficiencia, claridad y potencia, ofreciendo un equilibrio ideal entre facilidad de aprendizaje y capacidad para realizar tareas complejas. Su diseño orientado a mejorar la experiencia tanto de personas novatas como experimentadas ha llevado a Python a ser uno de los lenguajes de programación más enseñados en las universidades y más empleados en la industria hoy en día.

# Instalación de Python

Tu primera tarea será instalar el intérprete Python desde su página oficial en tu computadora de TechLab. ¡Veamos cómo!

- Ingresamos a Downloads y hacemos clic en el botón Download Python.
- Asegúrate de descargar la última versión (o como mínimo la versión 3.8.x).



[Hacé clic acá para acceder a la página oficial de python](#)

Procedemos con la instalación realizando los pasos típicos para instalar aplicaciones en nuestro sistema operativo.

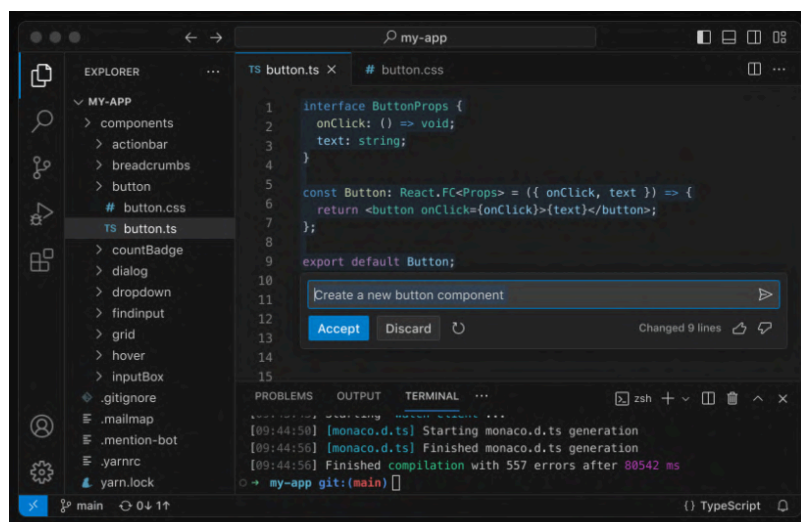


Si usás Windows, recordá instalar seleccionando la opción “*Add python.exe to PATH*”.

# Instalación de VSCode

Tu segunda tarea será instalar el entorno en el cual trabajarás en TechLab. ¿Sabías que para escribir código utilizaremos un editor de texto llamado **Visual Studio Code**?

Lo descargamos desde su página oficial y seguimos los pasos necesarios para su instalación, asegurándonos de crear un ícono de acceso en el escritorio o menú de nuestra computadora. ¡Qué emoción!



[Hacé clic acá para acceder a la página oficial de VSCode](#)

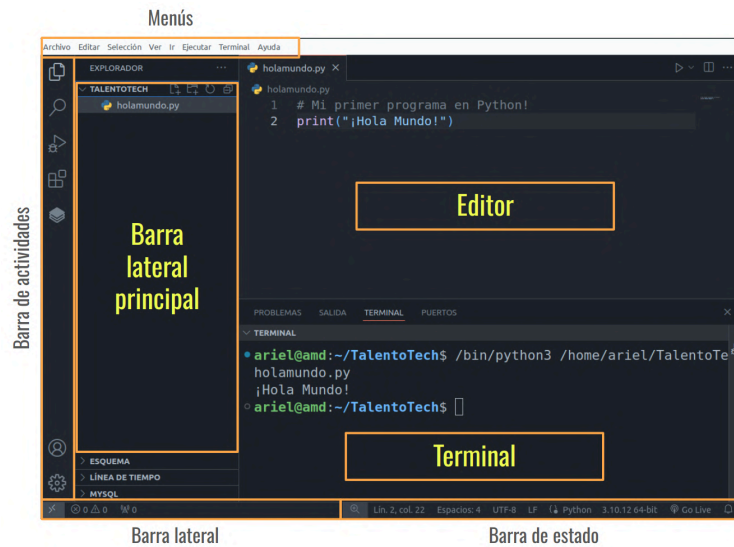
## La pantalla principal de VSCode

El diseño de **Visual Studio Code (VSCode)** está dividido en cinco áreas principales:

- **Editor:** Esta es la parte principal donde escribís tu código. Podés abrir varios archivos al mismo tiempo y verlos en diferentes secciones, tanto vertical como horizontalmente. Se parece bastante a la ventana de un navegador web y su sistema de pestañas. Podés borrarlas, moverlas, abrir nuevas, etcétera.
- **Barra lateral principal:** En esta sección encontrarás varias herramientas que te ayudan a trabajar en tu proyecto, como un explorador de archivos que te permite ver y acceder a todos los archivos de tu proyecto. Este explorador es similar al explorador de archivos que posee tu sistema operativo, pero está integrado en el editor.
- **Barra de estado:** En la parte inferior de la ventana, una barra de estado muestra información sobre tu proyecto y los archivos que estás editando, como el nombre del archivo y el estado de guardado. No la pierdas de vista: siempre hay información útil en esa zona de VSCode.
- **Barra lateral:** Está ubicada en el lado izquierdo. Te permite cambiar entre diferentes vistas y te muestra indicadores adicionales, como el número de cambios pendientes, si tenemos compartido algún puerto de comunicaciones, etcétera. Quizás no nos sea demasiado útil hasta dentro de un tiempo.
- **Terminal:** Este panel se encuentra en la parte inferior de la ventana, debajo del editor. Muestra información adicional como resultados de pruebas, errores y advertencias y una terminal integrada donde podés escribir comandos y ver el



resultado de la ejecución de tus programas Python. Este panel es móvil, podés ubicarlo a los lados izquierdo o derecho si preferís más espacio vertical en el editor.



## Nuestra primera línea de código en VSCode

**¡Tu primer programa en TechLab!** Te queremos contar que escribir una línea de código en VSCode es muy parecido a escribir texto en un procesador de textos, como Word o Google Docs. Primero, necesitás abrir VS Code y, dentro de él, abrir un archivo donde quieras escribir tu código.

No te olvides guardar el archivo luego de escribir tu código, es muy importante para no perder tu trabajo. En VSCode, puedes hacer esto haciendo clic en el icono de guardar (parece un disquete) en la parte superior de la ventana, o usando el atajo de teclado (**Ctrl+S** en Windows o **Cmd+S** en Mac). También puedes activar la opción "**Autoguardado**" que aparece en el menú "Archivo", y VSCode se encargará de guardar por vos cada modificación que hagas en el código.

Una vez que tu archivo esté abierto en VS Code, empezás a escribir código en el área principal del editor. Hacé clic en el área donde quieras escribir y comenzá a tipear.

## Primer programa en Python: "Hola, Mundo"

### ⚠ ¡ATENCIÓN!

Mariana, la jefa de proyecto, nos pidió que te ayudemos a escribir tu primer programa para acreditar lo que trabajamos hoy. Te dejamos una lista de los pasos a seguir para lograrlo. ¡Éxitos!

- **Creá un nuevo archivo:**
  - Seleccioná en **Archivo > Nuevo Archivo**.

- Guardá el archivo como `hola_mundo.py`.
- **Escribí el código exactamente de la siguiente manera:**

```
print("¡Hola, Mundo!")
```

- **Ejecutá el programa:**
  - Abrí la terminal integrada en VS Code (podés hacerlo con la tecla Ctrl+ñ o Ctrl+Shift+).
  - Asegurate de que la terminal esté en la carpeta correcta.
  - Ejecutá el comando:

```
python hola_mundo.py
```

Si usás Linux, debés escribir “python3” en lugar de “python”:

```
python3 hola_mundo.py
```

- **¡Compartile tu resultado a Mariana!**

```
¡Hola, Mundo!
```

---

## Reflexión final

¡Felicitaciones por completar tu primer día! Ya lograste configurar tu entorno de desarrollo, creado un proyecto y ejecutado tu primer programa. ¡No es poca cosa! Este es el primer paso fundamental para el éxito de nuestro proyecto en TechLab, donde pronto comenzaremos a construir soluciones más completas y útiles para satisfacer las necesidades de nuestras y nuestros clientes. Clase a clase desarrollaremos más y más herramientas que te ayuden a enfrentar los desafíos que vendrán. ¡Eso es todo por hoy!

---

## Materiales y recursos adicionales:

UNIR Formación Profesional: [Algoritmo: ¿qué es, para qué sirve, ejemplos de algoritmos y cómo funciona](#)

Cinco Noticias: [Algoritmos cotidianos](#)

Nate Gentile: [¿Cómo funciona una computadora y qué hace cada parte?](#)

GCFaprendeLibre: [¿Qué es software y qué es hardware?](#)

Sitio oficial de VSC: [Visual Studio Code Online](#) y [Documentación de Visual Studio Code](#)

Programiz.com: [Python Online Compiler](#)

El Blog de Aitana: [Atajos de teclado en Visual Studio Code](#)

---

## Preguntas para reflexionar:

1. ¿Qué importancia creés que tienen los algoritmos en la resolución de problemas cotidianos y cómo los podrías aplicar en situaciones fuera del ámbito de la programación? ¿Qué ejemplos se te ocurren?
  2. ¿Por qué considerás que Python es un lenguaje adecuado para iniciarte en la programación? ¿Qué características de Python te resultaron más interesantes?
  3. ¿Qué desafíos enfrentaste al configurar tu entorno de desarrollo y cómo los superaste? ¡Contanos!
- 

## Ejercicio práctico:

Preparar tu computadora para trabajar con Python,, asegurando que todas las herramientas estén correctamente instaladas y configuradas.

1. **Instalación del IDE para la próxima clase:**
  - **Objetivo:** Asegurarte de que tenés instalado y configurado correctamente tu IDE (Visual Studio Code).
  - **Tareas:**
    - Completá la instalación del IDE.
    - Abrí el IDE y familiarízate con su interfaz.
    - Asegurate que Python está correctamente integrado en el IDE.
2. **Salida de texto por consola:**
  - **Objetivo:** Practicar la creación y ejecución de programas en Python.
  - **Tareas:**

- Creá un nuevo archivo llamado personalizado.py y escribí un programa que imprima una frase diferente a "Hola Mundo", por ejemplo: "¡Bienvenidos al mundo de Python!".
- Ejecutá el programa y verificá que el mensaje se muestre correctamente en la consola

**Código ejemplo:**

```
print("¡Bienvenidos al mundo de Python!")
```

---

## Próximos pasos

En la próxima capacitación de **TechLab**, vamos a dar un paso más en el desarrollo de nuestras habilidades en Python. Aprenderemos cómo usar variables para almacenar información, cómo interactuar con la usuaria o usuario mediante funciones como **input()** y **print()** y cómo realizar operaciones aritméticas simples. Estos conceptos serán fundamentales para empezar a construir programas más dinámicos y útiles.

Además, comenzaremos a trabajar con el modelo Entrada-Proceso-Salida de una manera más práctica, aplicándolo para crear programas. Este será el primer paso para empezar a manejar la información de manera estructurada, sentando las bases para lo que desarrollaremos en los siguientes encuentros.

¡Nos vemos pronto para continuar avanzando!



**Buenos Aires**  
*aprende*  
Agencia de Habilidades para el Futuro

**BA** Buenos  
Aires  
Ciudad