pontia

www.pontia.tech

APRENDE PYTHON

PROGRAMACIÓN CON PYTHON : 10 FUNDAMENTOS BÁSICOS

ÍNDICE

ÍNDICE

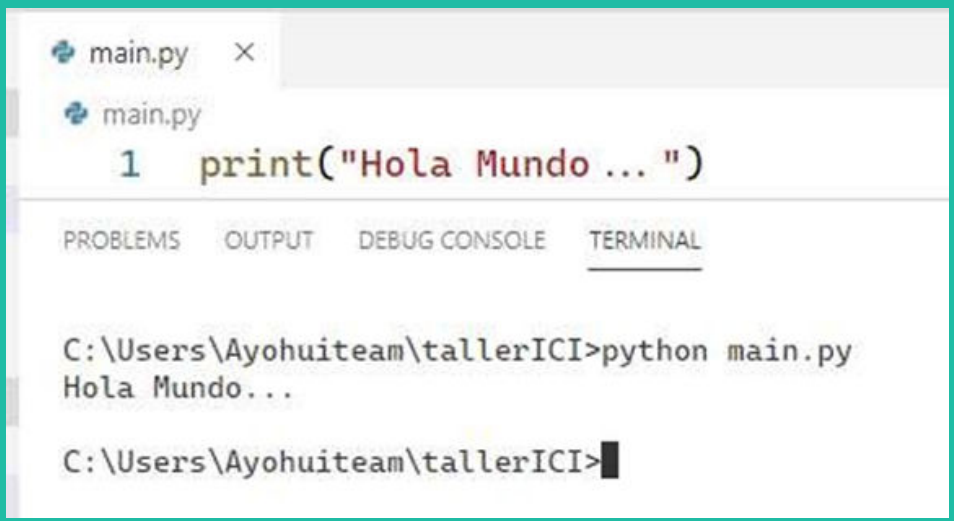
I. PROGRAMACIÓN CON PYTHON : 10 FUNDAMENTOS BÁSICOS

Con las **10** distintas versiones siguientes del **Hola Mundo...** que haremos a continuación, vamos a aprender varios conceptos importantes de **Python**, que posteriormente aplicaremos con problemas más complejos.

II. HOLA MUNDO VERSIÓN 1

Definitivamente la forma más sencilla de hacer un "Hola Mundo" es con una sola línea de código, pero vamos a ver porqué esta forma no siempre es la más adecuada.

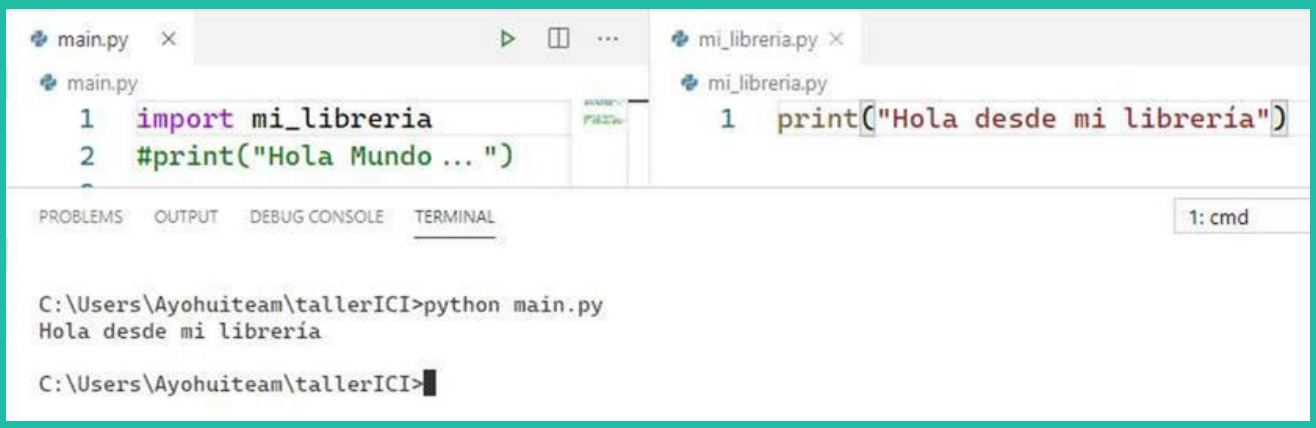
Pues bien, el programa quedaría de la siguiente forma:



III. HOLA MUNDO VERSIÓN 2

Como dijimos anteriormente, Python es un lenguaje interpretado iniciando la ejecución de arriba hacia abajo, ejecutando todo lo que se encuentre en su camino siempre y cuando no tenga errores.

Ahora vamos a utilizar la instrucción **import**, para separar nuestro hola mundo en una librería, para esto hacemos un nuevo programa que se llame **mi_libreria.py**, al cual le escribiremos la misma línea de **main.py** con el texto **"Hola desde mi librería..."**. Esta línea de código en el **main.py** la podemos comentar con el signo # o incluso borrar para que no se ejecute. Mira lo que pasa:



Como podrás observar, se ejecutó el código de **mi_libreria**. Pero si en **main.py**, necesito escribir código, mira lo que sucede en la siguiente versión al dejar la instrucción **print** del **main**.

IV. HOLA MUNDO VERSIÓN 3

Ahora se ejecutó el código de **mi_librería.py** y después el código de **main.py**. En la vida real, el desarrollo de software requiere de cientos de programas, y no es posible tener todo en un solo archivo, razón por la que es importante, desde los primeros pasos en la programación, aprender a estructurar de una manera adecuada nuestro código. Si quisieras probar **mi_librería** de manera separada podrías por supuesto ejecutarla:

De esta forma tienes el resultado solo de **mi_librería.py**. En la siguiente versión veremos cómo arreglar este problema, para que se ejecute solo el código de **main.py**

pontia

www.pontia.tech

¿Conoces nuestros programas en Data Science y Data Analytics?

MÁSTER EN DATA ANALYTICS

Aprenderás Matemáticas y Estadística, Python, SQL, Machine learning y Deep learning, Visualización de datos con Tableau y Big Data para convertirte un perfil clave en cualquier empresa tecnológica.

¡Más información!

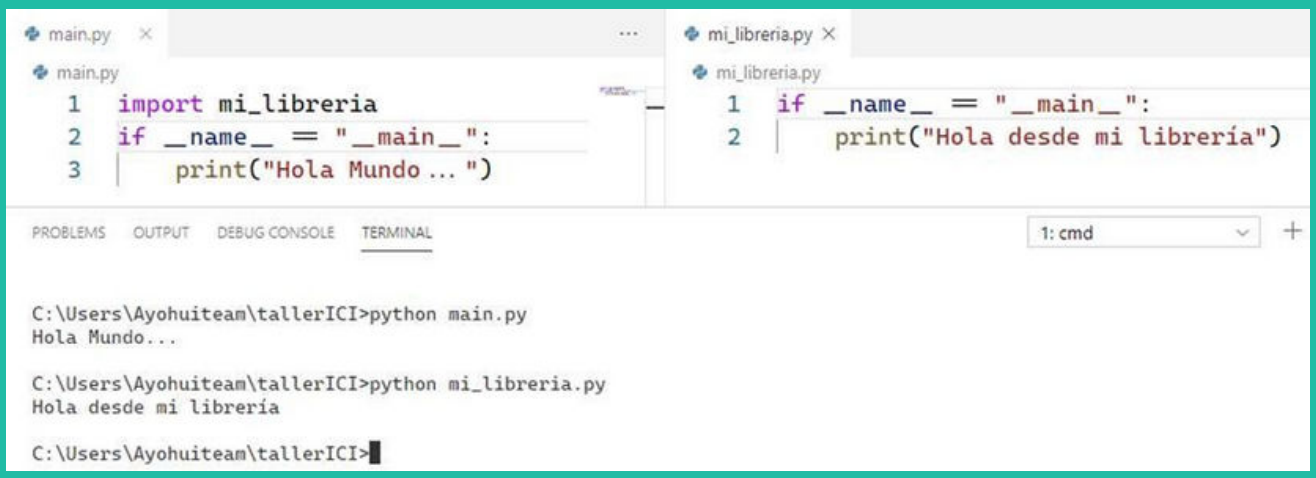
MÁSTER EN DATA SCIENCE

Te formarás en Matemáticas y Estadística, Python, SQL, Excel, Power Bi, Visualización de datos y Big Data para convertirte un perfil clave en la toma de decisiones de cualquier empresa tecnológica.

¡Más información!

V. HOLA MUNDO VERSIÓN 4

Ahora es donde va a tener sentido la condición **if** que usamos en la entrega anterior del taller. Colocamos la instrucción **print** dentro de un bloque **if** en cada uno de los archivos y al ejecutarlos de manera independiente, solo se ejecuta un bloque **if**:

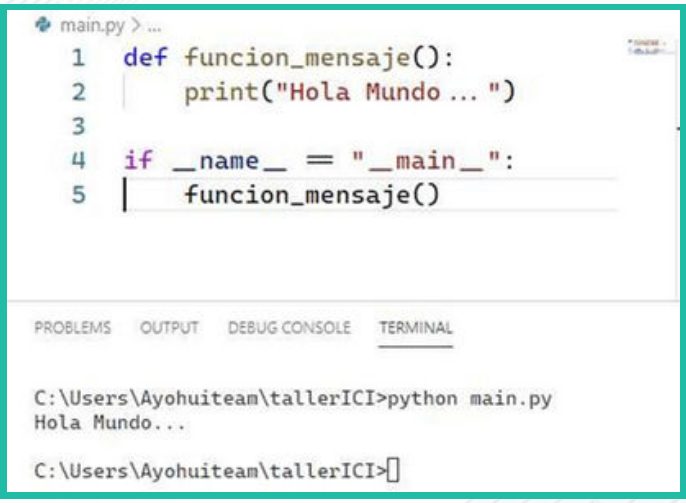


Como podrás observar, se ejecutó el código de **mi_libreria**. Pero si en **main.py**, necesito escribir código, mira lo que sucede en la siguiente versión al dejar la instrucción **print** del **main**.

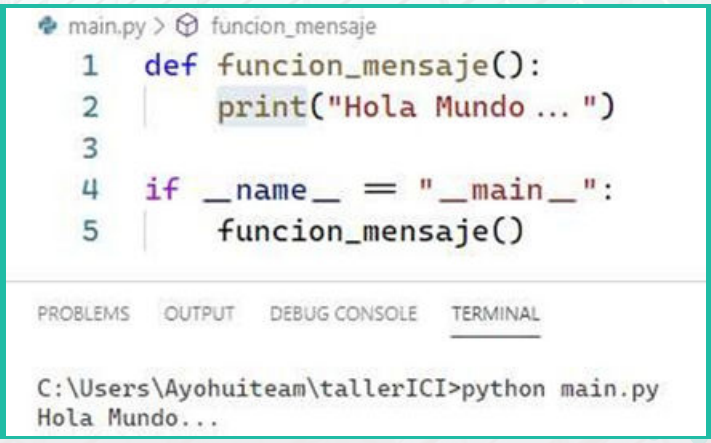
VI. HOLA MUNDO VERSIÓN 5

Uno de los elementos más importantes de cualquier lenguaje de programación son las funciones o procedimientos. Una función en programación es un pequeño subprograma que resuelve un problema pequeño. La forma más sencilla es usarla como si fuera un procedimiento;

Ahora el mensaje de **"Hola Mundo"** , lo pondremos en una función de este tipo.



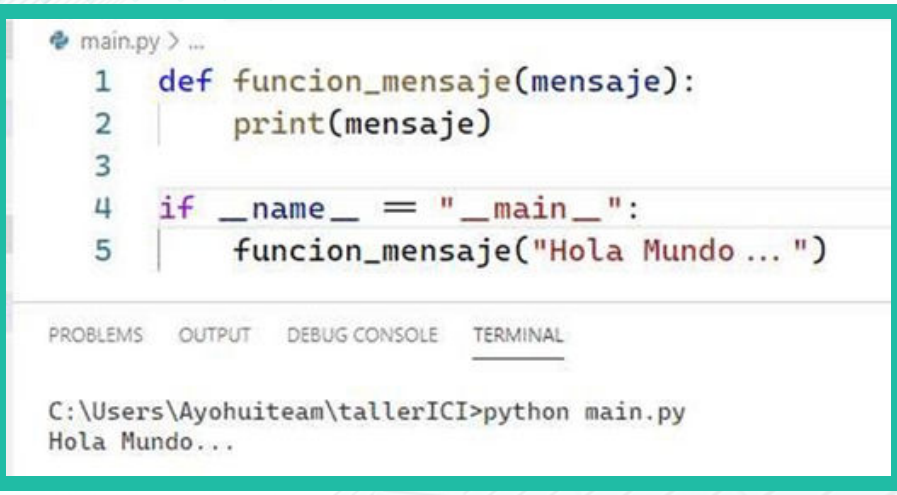
La palabra **def** permite crear una función, posteriormente lleva el nombre de lafunción, que se recomienda por cuestiones de estilo, lleve un guión bajo (_) como separación en cada palabra del nombre de la función. Después lleva los paréntesis para indicarle los argumentos que va a llevar la función (en la siguiente versión le pondremos argumento) y por último los dos puntos (:) que indican dónde inicia el bloque de la función al igual que con el **if**. En la línea 5, cambiamos la instrucción **print** por la llamada o invocación a la función.



VII. HOLA MUNDO VERSIÓN 6

Como dijimos en la versión anterior, ahora vamos a enviarle a la función un mensaje personalizado desde su invocación. La primera versión de esta función es la más sencilla.

Como mencionamos previamente, Python es un lenguaje no tipado, lo que significa que no es necesario especificar el tipo de dato de un argumento o de una variable. El programa quedaría de la siguiente forma:

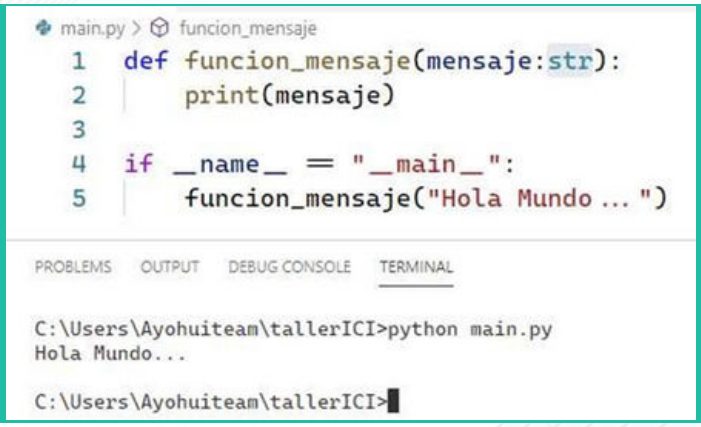


En la línea 1, entre los paréntesis colocamos la variable **mensaje**, la cual funciona como argumento de entrada a la función para su impresión. En la línea 5, al momento de invocar a **funcion_mensaje** , le colocamos el mensaje **"Hola Mundo..."** entre los paréntesis para que sea recibido su valor por la variable de la función.

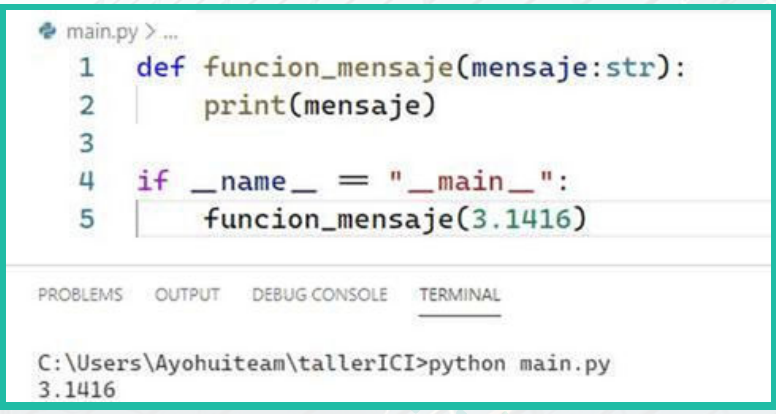
VIII. HOLA MUNDO VERSIÓN 7

Cuando se tienen pocas líneas de código, puede no ser necesario o útil saber qué tipo de datos tienen las variables, lo ideal es saber de alguna forma cuál es el tipo de dato a utilizar al llamar una función.

El primer tipo de dato que vamos a usar es el **string** (cadena de caracteres), que en Python se identifica con **str**. Nuestra función tipada quedaría de la siguiente forma:



Basta con escribir dos puntos y el tipo de dato a un lado de la variable(:**str**). Estos son conocidos como **hints** (pistas) y son solo eso, no significa que la variable sea **string**, puede ser de cualquier tipo. Lo que si ayuda es para hacer análisis estático del código, esto lo veremos en **Hola Mundo Versión 16**.
Veamos el siguiente ejemplo:

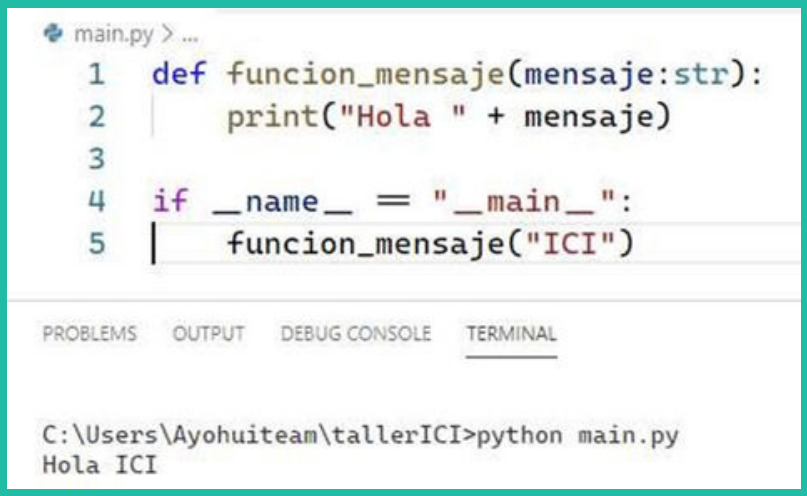


Como te pudiste dar cuenta, el indicar que **mensaje** es **str** no evitó que aceptara un valor con decimales, sirve como ayuda para saber qué datos enviarle a la función.

IX. HOLA MUNDO VERSIÓN 8

Ahora vamos a hacer un Hola Mundo más personalizado, hagamos que la función le diga Hola a todo aquello que reciba como argumento.

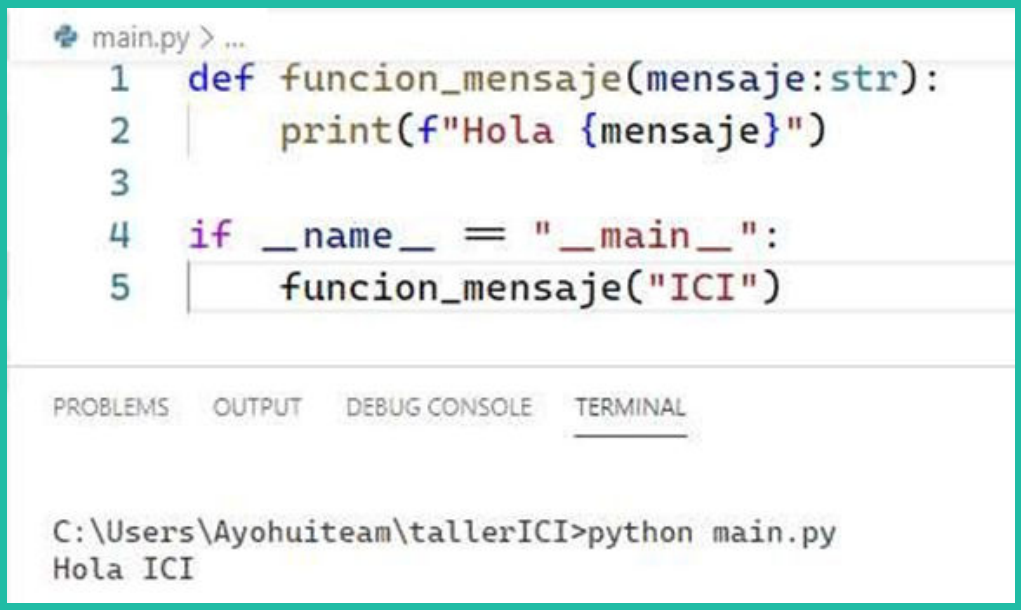
Si le mando **"Mundo"**, va a contestar la función Hola Mundo, si le mando **"ICI"** va a contestar **Hola ICI**.



La única línea que modificamos fue la 2, dentro de los paréntesis del **print**, dejamos la cadena **"Hola "**, y escribimos el operador de concatenación + (unión de cadenas), con el argumento de entrada en **funcion_mensaje**.

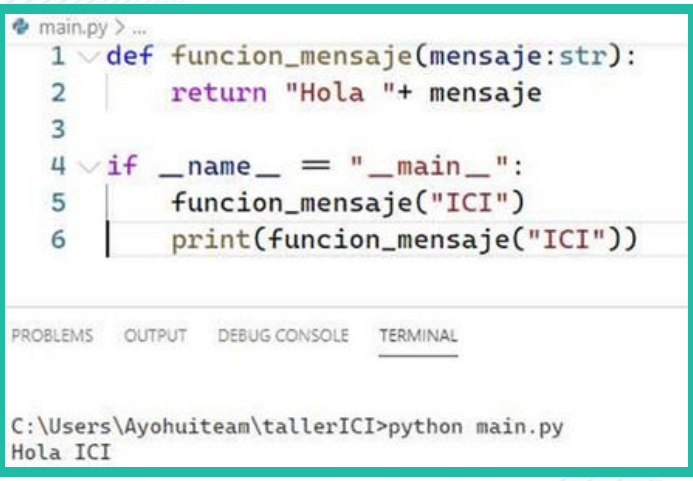
X. HOLA MUNDO VERSIÓN 9

Una versión mejorada y más elegante de este mismo programa es usando el formato de cadenas mediante f-strings. Para esto solo se antepone la **f** antes de la comilla inicial y cada una de las variables se coloca entre corchetes ({}), sin necesidad de utilizar el operador de concatenación



XI. HOLA MUNDO VERSIÓN 10

Lo normal es utilizar una función que realice un proceso y regrese o retorne algo a quien la invocó o donde se asignó. Vamos a hacer ahora que la función nos retorne la cadena en una sola variable, para esto hay que utilizar la instrucción **return**, la cual es la que regresa el valor a quien hizo uso de la función.



La línea 2, regresa la concatenación de las dos cadenas de caracteres, en la línea 5 se invoca la función, pero como no se hace nada con el valor retornado, se pierde y no se imprime nada. Por su parte en la línea 6, la instrucción **print** recibe la cadena de la función y la imprime en pantalla. Cómo corregimos el error de la línea 5, simplemente asignándola a una variable y posteriormente imprimiéndola, esto es:

Ahora en la línea 5 asignamos **funcion_mensaje**("ICI") a la variable **mensaje_recibido** y en la línea 6 imprimimos su valor

www.pontia.tech

¡Esperamos que este recurso haya sido útil para ti en tu camino al aprendizaje con Python! No es sencillo, pero con práctica y dedicación sabemos que podrás lograrlo.

Visita nuestra web: www.pontia.tech

¡Ir a la web!