

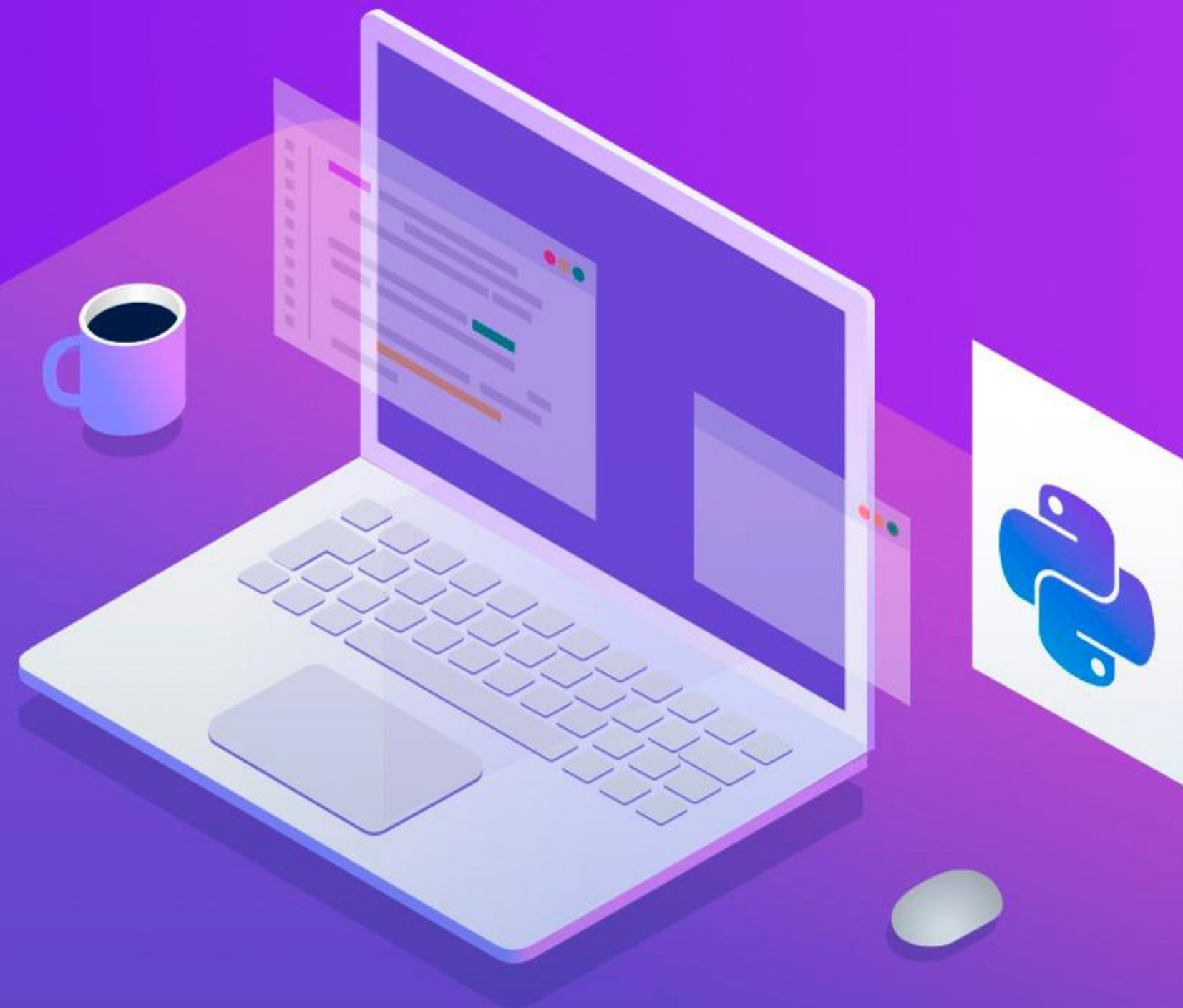


PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

INTRODUCCIÓN

A LA PROGRAMACIÓN

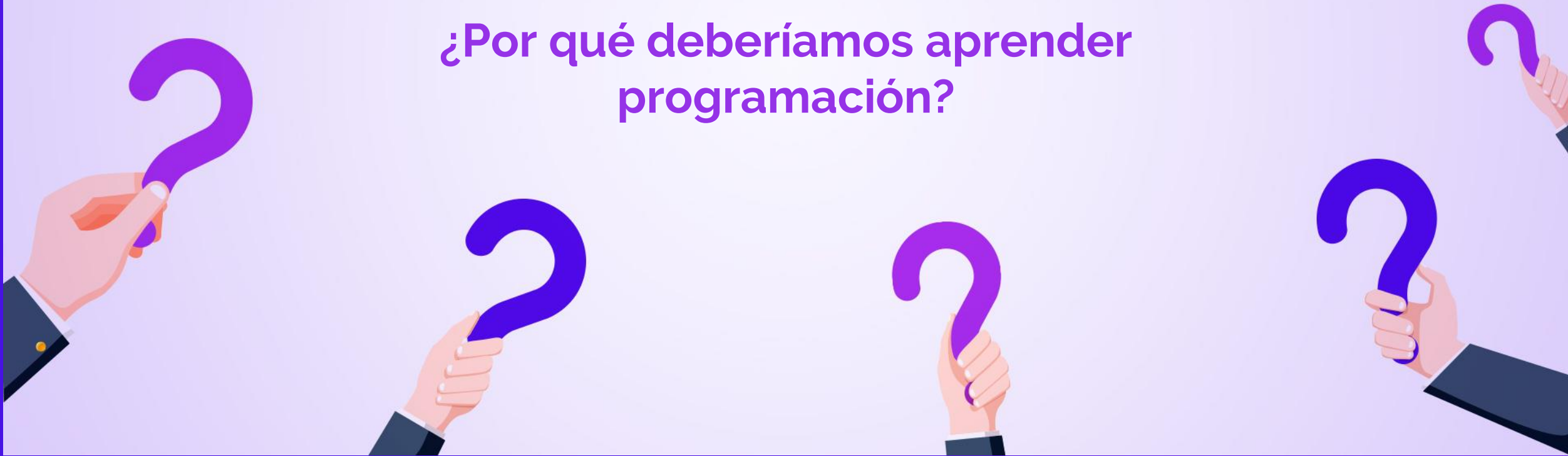
>>> Parte 1: Secuencias de comandos en Python



Motivación

Reflexión

¿Por qué deberíamos aprender programación?



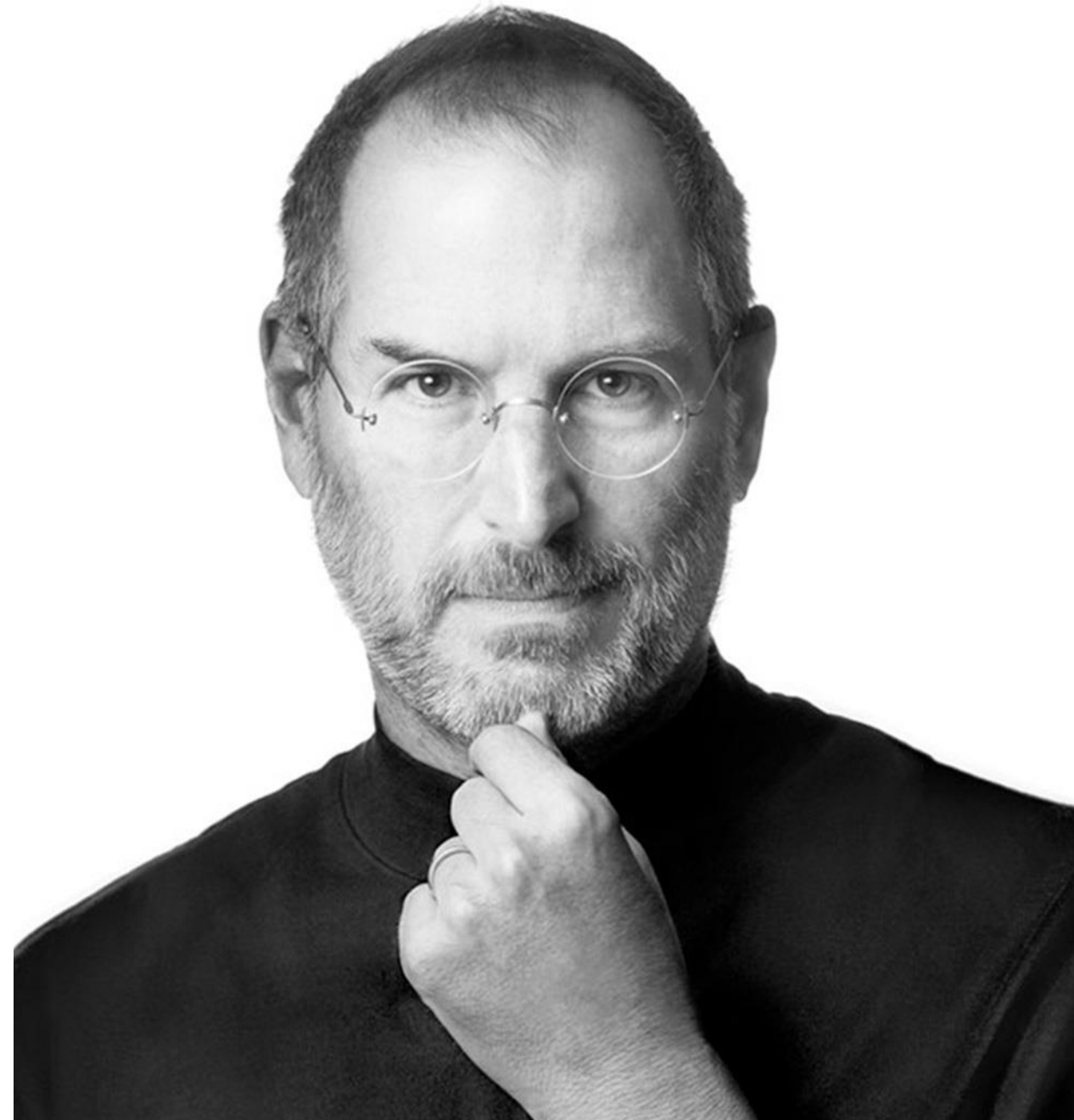
Motivación

Reflexión

Una posible respuesta que nos hace sentido, es que la programación es útil en diversas áreas del conocimiento, desarrolla el pensamiento lógico y potencia una de las habilidades fundamentales que es la resolución de problemas.

“Todo el mundo debería aprender a programar un ordenador, porque te enseña a pensar”

Steve Jobs



Motivación

Reflexión

¿Por qué deberíamos aprender a programar en Python?

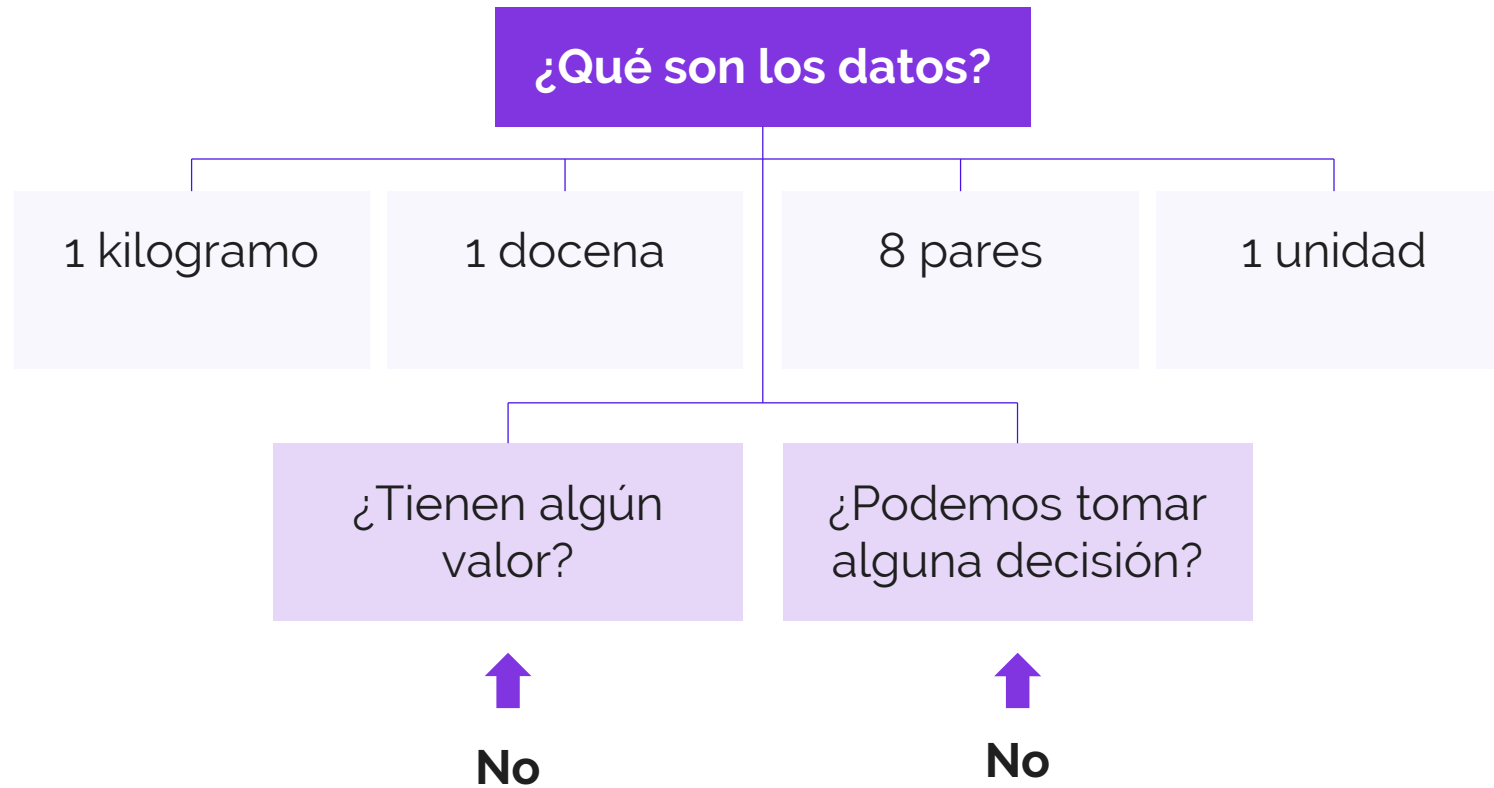
En particular se ocupará el lenguaje de programación Python, porque está presente en muchas aplicaciones y servicios que utilizamos de manera habitual, por ejemplo, Smartphone, computadores, video juegos, YouTube y Google entre muchos otros usos.

En este curso abordaremos el uso de conceptos básicos y herramientas de programación que permitan el procesamiento de datos.

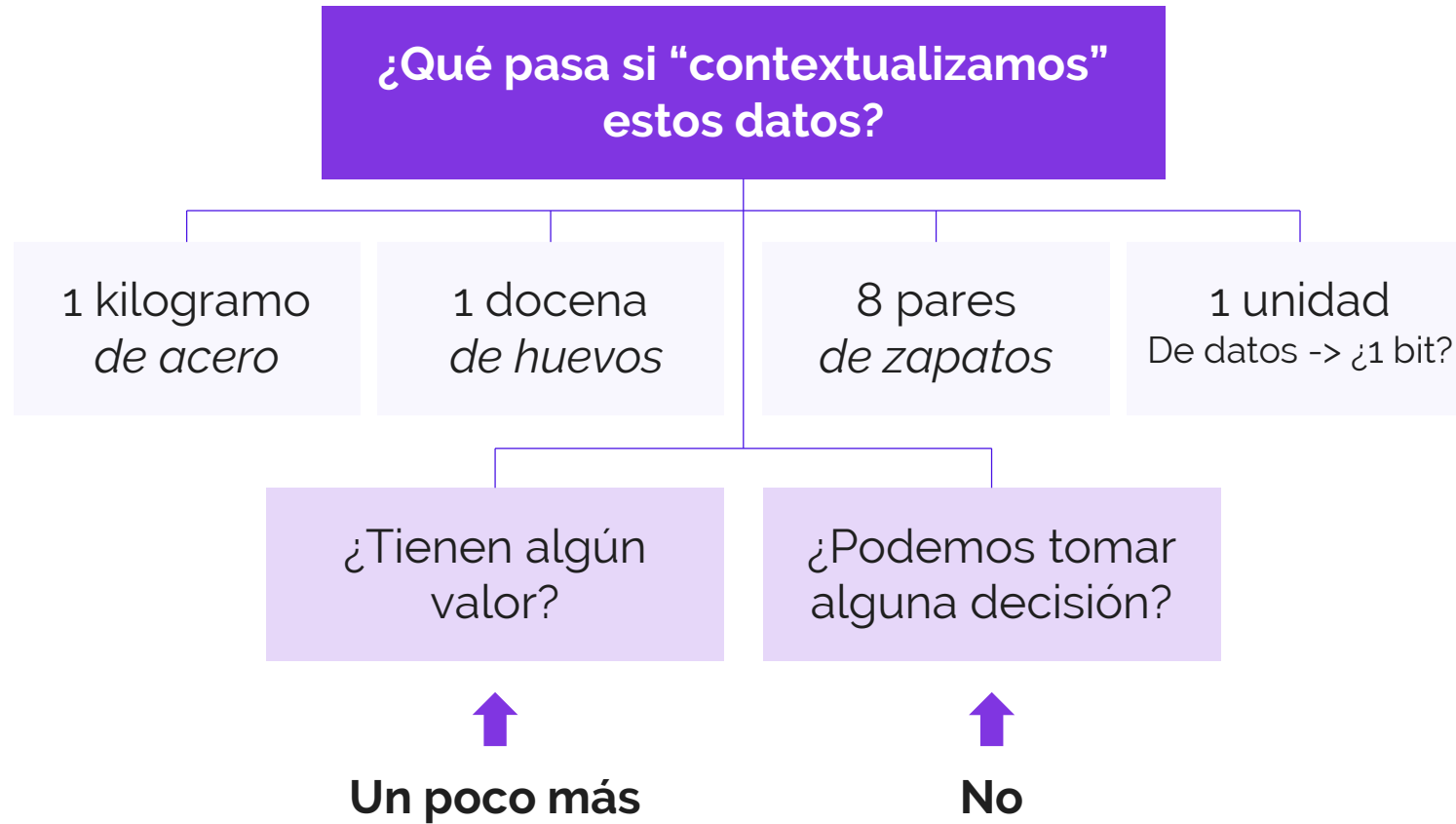


Como ya reflexionamos sobre la relevancia e impacto que tiene el aprendizaje sobre programación en Python, ahora te invitamos a explorar los conceptos claves y su relación con el mundo laboral, para iniciar el estudio del curso.

De los datos a la información



De los datos a la información

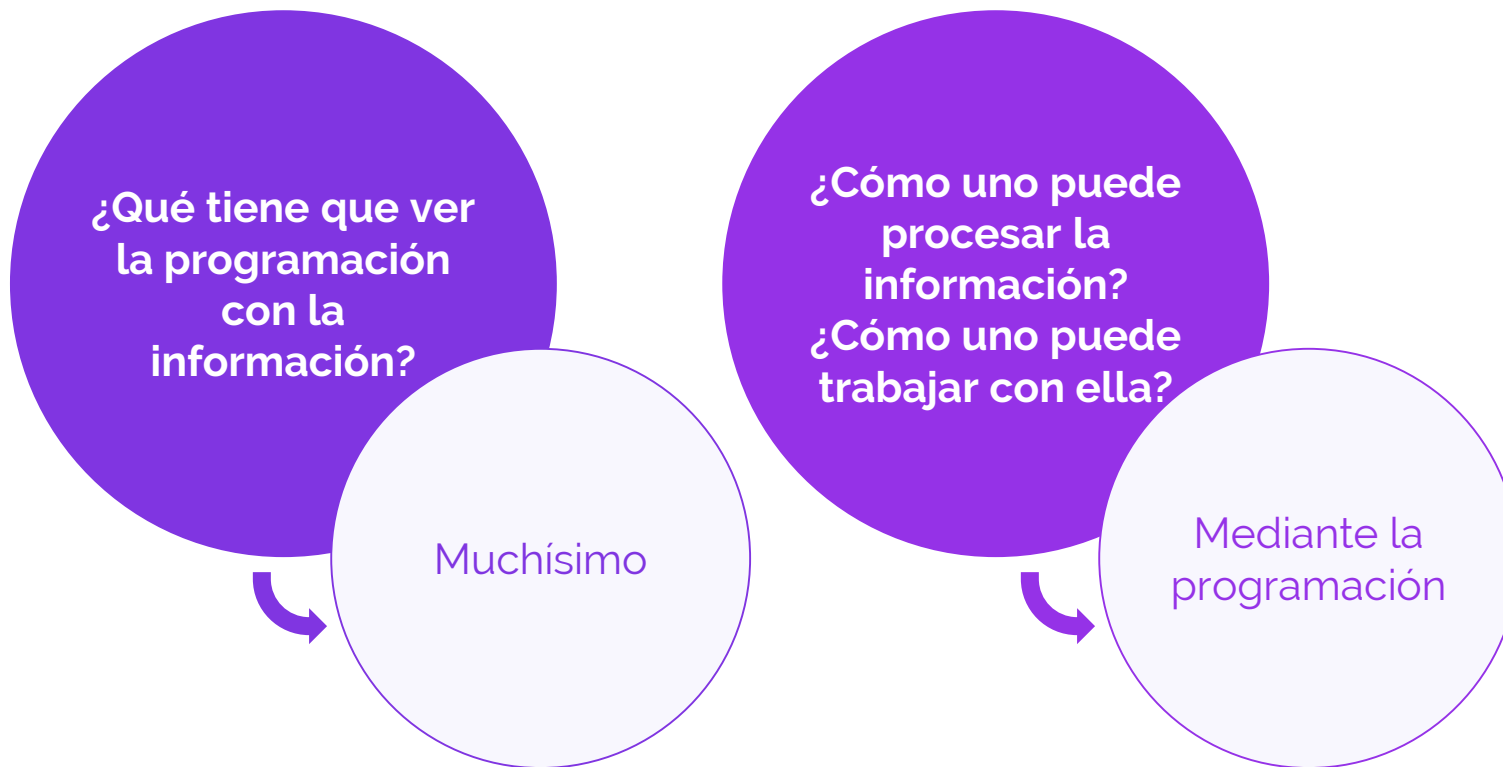


De los datos a la información

Estos datos los transformamos en información.



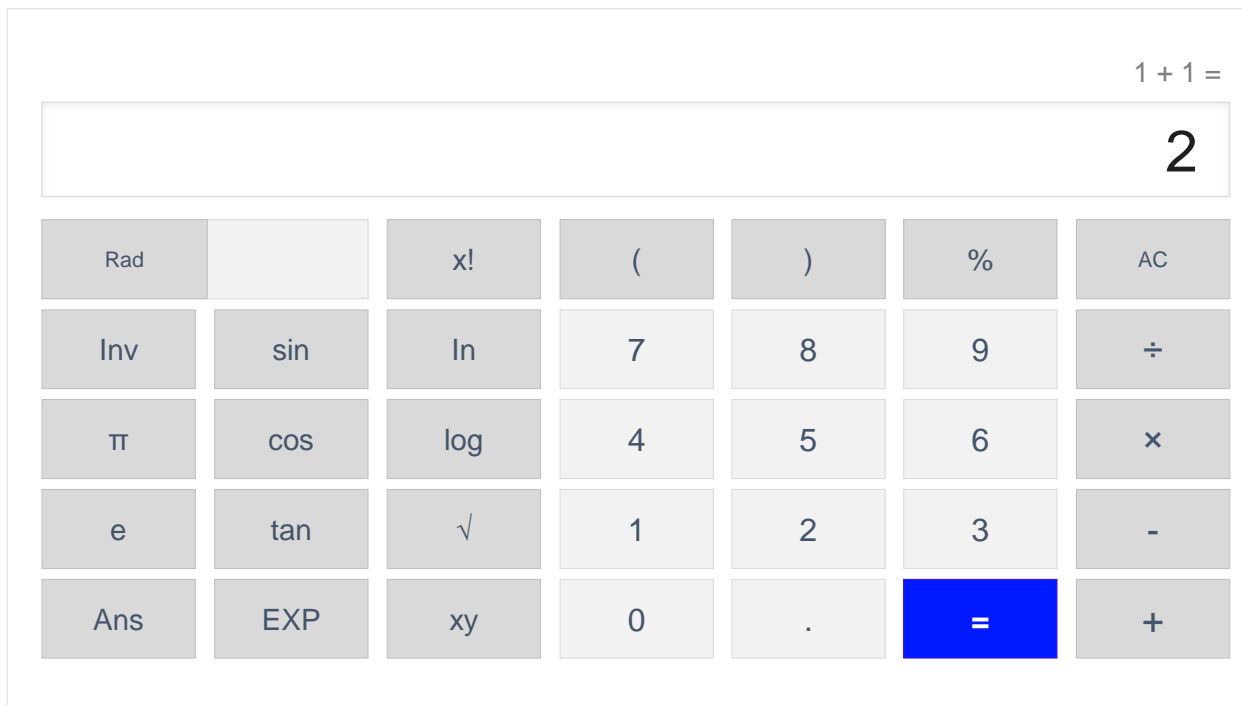
Datos, Información y Programación ¿Cómo conviven hoy estos elementos en el mundo laboral?



¿Qué es la programación?

Es darle instrucciones a un computador para poder ejecutar una tarea.

Ejemplo:



¿Cómo funciona la programación?

Las instrucciones se dan a través de una secuencia de comandos.

El computador puede interpretar la secuencia de comando, por medio de distintos lenguajes de programación.

Hay muchos lenguajes de programación. Todos tienen distintos orígenes y propósitos.

Algunos se ocupan en los más diversos ámbitos.

Otros son sumamente específicos.

Aplicaciones prácticas

¿Cómo se aplica la programación?

La programación se ocupa en los más diversos ámbitos y contextos.



Hoy en día, los autos tienen computadores que permiten manejar todos sus implementos electrónicos. Estos mismos computadores permitirían en un futuro no tan lejano que los autos puedan llegar a manejarse de forma autónoma.



La agricultura y el mundo de la salud usan computadores a diario y de forma constante para mejorar todos sus procesos y poder hacerlos más eficientes y efectivos.

Programación en el mundo laboral y cómo debe convertirse en un hábito

¿Cuál es el desafío?

El desafío radica en cómo podemos aprovechar esa información, y además hacerlo de forma eficiente.

Por ejemplo a un volumen muy grande -> BIG DATA

¿Qué pasa a una pequeña escala?

Por ejemplo:

Supongamos que trabajo en un call-center y recibo las conversaciones por chat de uno de los ejecutivos durante un año.

¡El problema es que esta información es mucha!

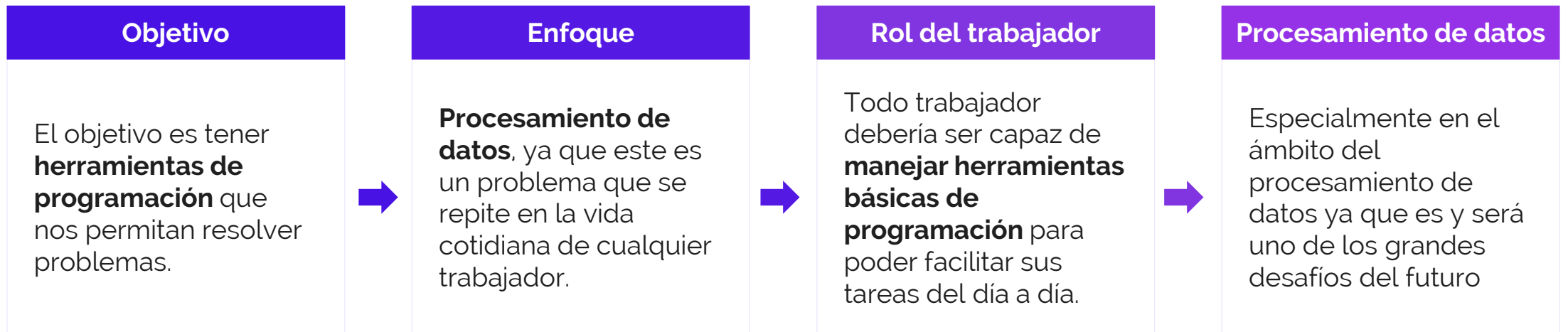
¿Qué otros problemas podría tener?

Surgen otras dificultades:

¿Cómo busco la conversación de un solo día?
¿Qué pasa si la conversación tiene faltas de ortografía?

Y muchos otros más...

Programación en el mundo laboral y cómo debe convertirse en un hábito



>>> Secuencias de comandos en Python

¿Existe solo una manera de ocupar la programación?

No, hay muchas maneras de ocupar la programación como **una herramienta para el procesamiento de datos**

La base detrás de esta idea es poder darle instrucciones a un computador para poder ayudarnos en esta tarea.

¿Cuál es la forma de entregarle instrucciones al computador?

Mediante lenguajes de programación.

El ejemplo más clásico para darle instrucciones a un computador ocupando cierto lenguaje de programación como operaciones matemáticas.

Ejemplo:

1 + 1 =

2

Rad

x!

(

)

%

AC



Secuencias de comandos en Python

- Es uno de los lenguajes de programación más populares en el mundo
- Simple y fácil de aprender
- Usado en muchos contextos (especialmente en ciencia de datos, que luego explicaremos por qué es importante)



Hagamos una prueba

Recuerda revisar la
Ruta de ejercicios.
Ejercicio EM1-01 →



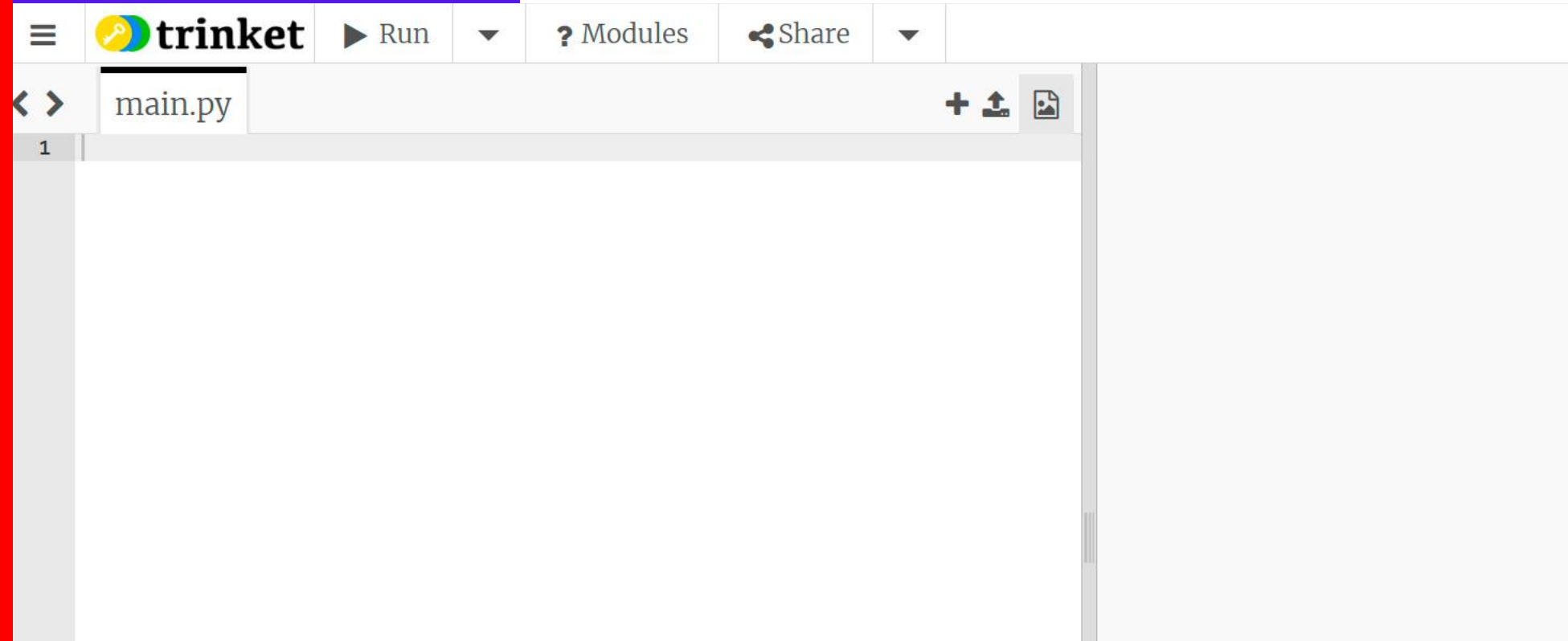
```
>>> print("Hola mundo")
Hola mundo
```

Ingresa el siguiente comando en la consola y presiona la tecla enter

Comando `print`

Recuerda revisar la
Ruta de ejercicios.

Ejercicio EM1-01 →



El primer comando que veremos es el comando `print(...)`.
Sirve para poder ver el resultado de los comandos que se ingresan en la consola.

Es una acción básica, ya que si no imprimimos lo que hacemos en la consola entonces no sabremos si nuestra operación está bien o mal.

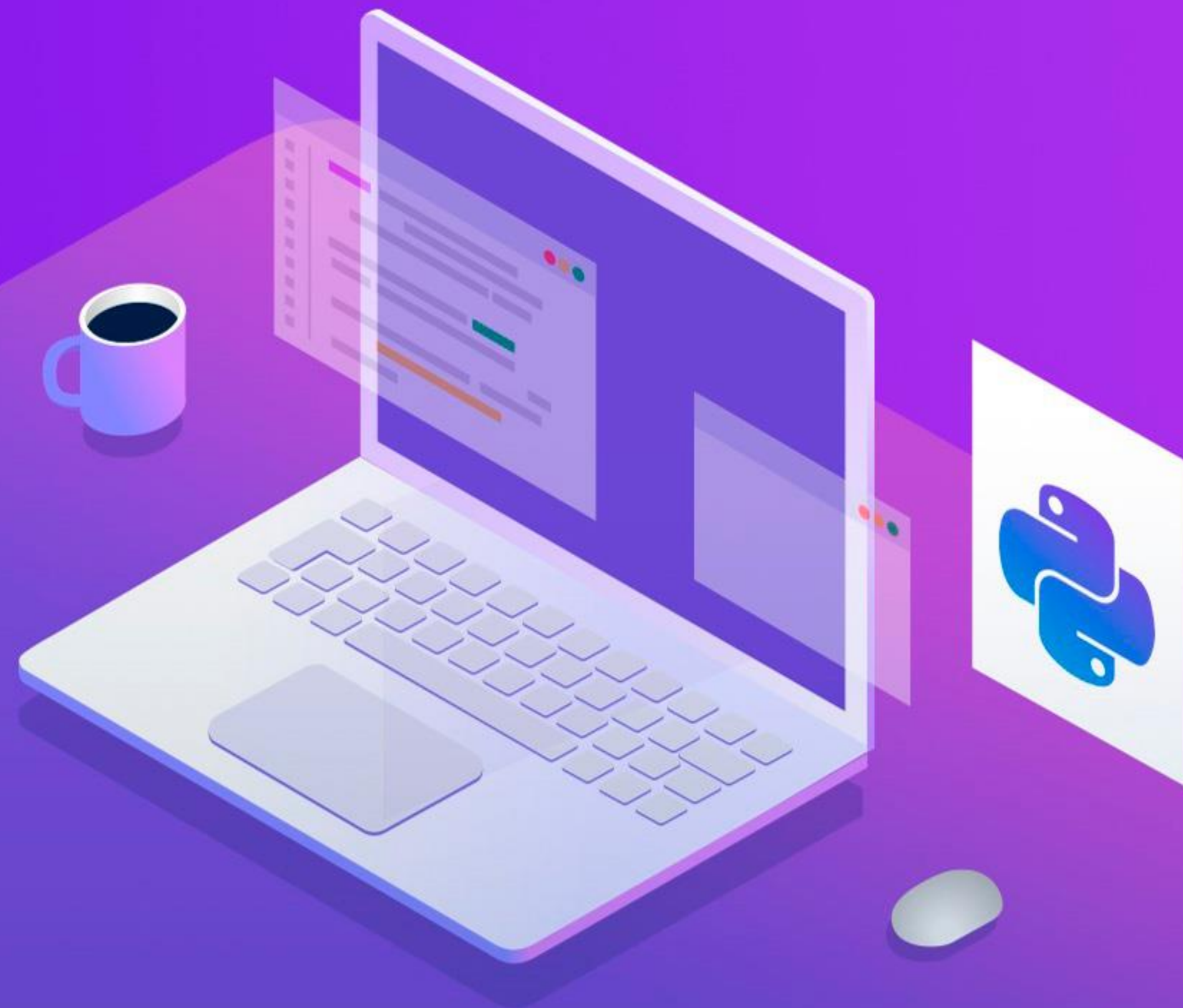


PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

INTRODUCCIÓN

A LA PROGRAMACIÓN

>>> Parte 2: Operaciones básicas



¿Qué otros comandos puedes ingresar en Python?

Partamos por los más básicos, que son los **operadores matemáticos**

En general son la base de los lenguajes de programación y fueron la razón de su origen

A medida que la tecnología avanzaba se requerían cada vez cálculos más complejos.

Operadores matemáticos

CÓDIGO	RESULTADO
<code>print(4+7)</code>	11
<code>print(1001+817)</code>	1818

+ Suma los valores de la izquierda y la derecha

Operadores matemáticos

CÓDIGO	RESULTADO
<code>print (4-7)</code>	-3
<code>print (7-4)</code>	3

— Resta el valor de la derecha al valor de la i

Operadores matemáticos

CÓDIGO	RESULTADO
<code>print (5*8)</code>	40
<code>print (25*0)</code>	0

*

Multiplica el valor de la izquierda por el de la derecha

Operadores matemáticos

CÓDIGO	RESULTADO
<code>print(12/4)</code>	3.0
<code>print(4/12)</code>	0.3333333333333333

El resultado de la operación es un número decimal. También puedes trabajar con ellos.

Más adelante explicaremos los distintos **tipos de datos**.



Divide el valor de la izquierda por el de la derecha

Operadores matemáticos avanzados

CÓDIGO	RESULTADO
<pre>print(2**5)</pre>	32

Calcula la potencia. El valor de la izquierda es la base y el de la derecha es el exponente.

Operadores matemáticos avanzados

CÓDIGO	RESULTADO
<pre>print(5//4)</pre>	1

En esta operación, el resultado de la división es 1.25

El resultado y lo que se imprime en la consola es 1 porque el operador retorna la parte entera del número.

// Divide el valor de la izquierda por el de la derecha, y entrega el valor entero del resultado.

Operadores matemáticos avanzados

CÓDIGO	RESULTADO
<code>print(7%5)</code>	2

Se puede observar que el 5 cabe una vez en el 7 y sobran 2 unidades, que es el resto de la operación. Por lo tanto, este es el resultado de la operación y lo que se imprime en la consola.

% Divide el valor de la izquierda por el de la derecha. No entrega el resultado sino que el resto de esta división.

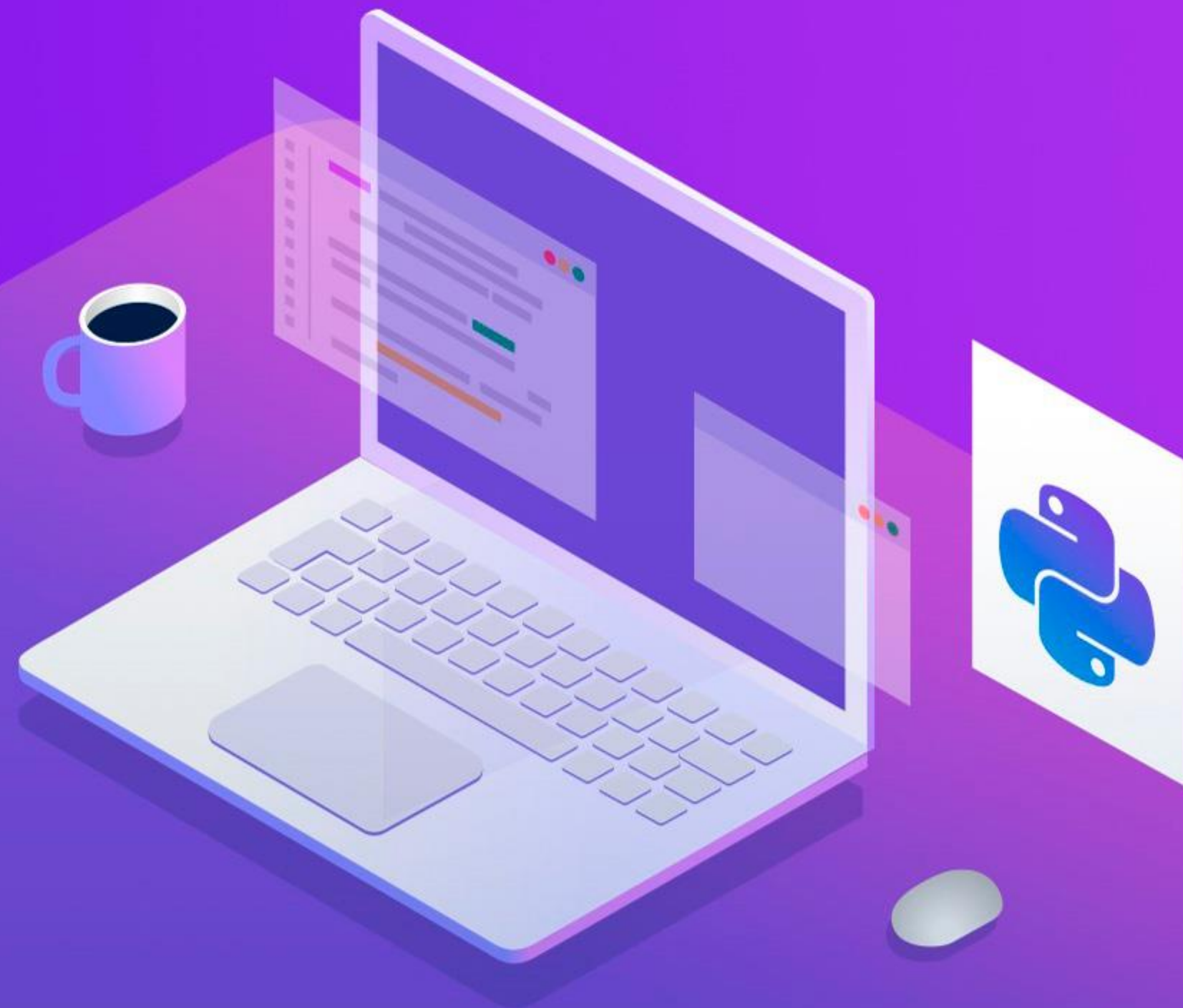


PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

INTRODUCCIÓN

A LA PROGRAMACIÓN

>>> Parte 3: Creación y asignación de variables



¿Qué tendríamos que hacer si quisiéramos guardar el resultado de una operación para utilizarlo nuevamente?

Variables



Para responder a esto introducimos el concepto de **variables**

Variables

Las variables son objetos donde podemos almacenar información.

PYTHON	RESULTADO
<pre>variable1=2 print(variable1)</pre>	2

Uno puede crear variables y asignarles inmediatamente un valor.
Es decir, estamos declarando una variable de forma explícita.

Variables

También podemos crear una variable asignándole el resultado de alguna operación.

PYTHON	RESULTADO
<pre>variable2=30*50 print(variable2)</pre>	1500

Variables

Es importante notar que las variables pueden tener cualquier nombre.

PYTHON	RESULTADO
<code>casa=1</code>	1
<code>print(casa)</code>	4.0
	28
<code>perro=8/2</code>	
<code>print(perro)</code>	
<code>perro_grande1=28</code>	
<code>print(perro_grande1)</code>	

Variables

¡Y lo más importante, puedes operar con variables!

Digamos que tienes 30 manzanas y 6 niños. Uno quiere saber cuántas manzanas puede comer cada niño.

PYTHON	RESULTADO
<pre>manzanas=30 niños=6 cantidad_de_manzanas_por_niño=manzanas/niños print(cantidad_de_manzanas_por_niño)</pre>	5.0

Haciendo la operación con las variables, podemos también llegar a un resultado.

Además, este resultado lo podemos asignar a otra variable.

Variables

Analicemos el ejemplo anterior:

1

El resultado de la división lo asignamos a la variable `cantidad_de_manzanas_por_niño`.

2

Sin embargo y a diferencia de las variables anteriores, el resultado de la división (aunque fuera un número entero), apareció como 5.0

3

Esto es porque este tipo de dato es **decimal**, a diferencia de los ejemplos anteriores donde el tipo de dato es **entero**.

Variables en Python

Python tiene varios tipos de datos, por ahora, solo debes manejar estos:

int

Variables que contienen números enteros.

float

Variables que contienen números decimales.

string

Variables que contienen información de tipo texto.

Variables en Python

int

Variables que contienen números enteros.

CÓDIGO	RESULTADO
<pre>variable_entera=-9 print(variable_entera)</pre>	<pre>-9</pre>

Variables en Python

float

Variables que contienen números decimales

CÓDIGO	RESULTADO
<pre>variable_decimal=8.75 print(variable_decimal)</pre>	8.75

Variables en Python

string

Variables que contienen información de tipo texto.

CÓDIGO	RESULTADO
<pre>variable_texto="este es un texto de ejemplo" print(variable_texto)</pre>	<pre>este es un texto de ejemplo</pre>

Las variables tipo `string` las podemos crear de forma explícita.

No obstante, muchas veces nos gustaría poder crear (o usar) variables de texto que contengan información que el usuario ingresa.

Input del usuario

Para crear (o usar) variables de texto que contengan información que el usuario ingresa ocuparemos el **comando** `input`

Los comandos son funcionalidades y herramientas específicas de Python, que nos ayudan a realizar tareas específicas.

1

El comando `input` imprime texto en la consola.

La consola “espera” que el usuario escriba algo y presione la tecla Enter. Esta información luego se almacenará en la variable a la que apunta este comando.

2

La información que el usuario ingresa siempre será de tipo texto, no obstante, después puede “transformarse” a otros tipos de datos.

Revisemos un caso sobre `input`

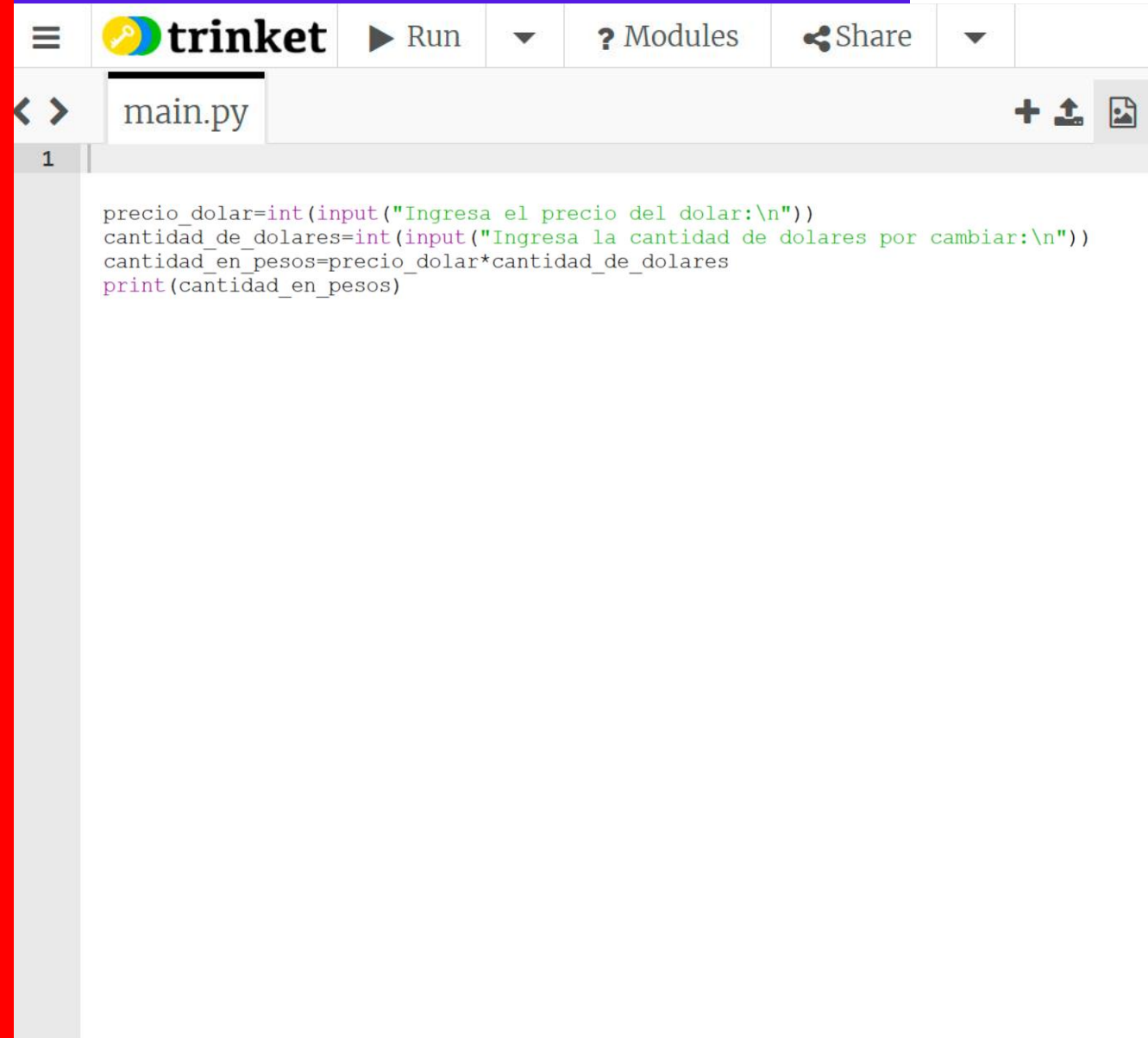
Eres el dueño de la casa de cambio "Los Robles". Quieres optimizar tu negocio, para ello, tendrás un sistema que automáticamente cambiará de pesos chilenos a dólares. Como sabes que el precio del dólar se actualiza diariamente, necesitas que el usuario ingrese el precio del dólar del día de hoy y luego ingrese la cantidad de dólares que desea transformar a pesos chilenos.



Solución Ejercicio Resuelto 1

Recuerda revisar la
Ruta de ejercicios.

Ejercicio EM1-16 →



The screenshot shows the Trinket IDE interface. At the top, there is a navigation bar with a menu icon, the Trinket logo, a 'Run' button, a dropdown arrow, a 'Modules' link, and a 'Share' button. Below this, the file name 'main.py' is displayed. The main area contains a Python script that takes two inputs: the price of a dollar and the quantity of dollars to convert, and then prints the total amount in pesos.

```
1
precio_dolar=int(input("Ingresa el precio del dolar:\n"))
cantidad_de_dolares=int(input("Ingresa la cantidad de dolares por cambiar:\n"))
cantidad_en_pesos=precio_dolar*cantidad_de_dolares
print(cantidad_en_pesos)
```

Ingresa el precio del dolar:

644

Ingresa la cantidad de dolares por cambiar:


1500

966000

Solución Ejercicio Resuelto 1

Recuerda revisar la
Ruta de ejercicios.

Ejercicio EM1-16 →



```
1
precio_dolar=int(input("Ingresa el precio del dolar:\n"))
cantidad_de_dolares=int(input("Ingresa la cantidad de dolares por cambiar:\n"))
cantidad_en_pesos=precio_dolar*cantidad_de_dolares
print(cantidad_en_pesos)
```

Ingresa el precio del dolar:

644

Ingresa la cantidad de dolares por cambiar:

1500


966000

Es interesante notar que en las líneas 1 y 2, el comando input está dentro de un paréntesis, y antecedido por la palabra `int`.

Solución Ejercicio Resuelto 1

Recuerda revisar la
Ruta de ejercicios.

Ejercicio EM1-16 →



```
1 precio_dolar=int(input("Ingresa el precio del dolar:\n"))
2 cantidad_de_dolares=int(input("Ingresa la cantidad de dolares por cambiar:\n"))
3 cantidad_en_pesos=precio_dolar*cantidad_de_dolares
4 print(cantidad_en_pesos)
```

```
Ingresa el precio del dolar:
644
Ingresa la cantidad de dolares por cambiar:
1500
966000
```

La lógica detrás de esto es que Python asume que cualquier dato que el usuario ingresa es de tipo `string`. Como posteriormente nosotros queremos multiplicarlo por otro valor (línea 3), necesitamos que Python los reconozca como enteros, y no como texto. Por eso, le estamos diciendo que la información que recibamos por el comando `input` se considere como entero.

Cierre

Has finalizado la revisión de los contenidos que corresponden a esta clase.

A continuación, te invitamos a estudiar la siguiente clase del módulo.