

P1

Programación 1

UNIDAD
CLASE

CONCEPTOS INICIALES



| Introducción a la lógica de la programación



Al finalizar esta primera clase ya te habrás introducido con:

| La lógica de la programación
| Algoritmos
| Diagramas de flujo
| Variables
| Instalación de Python
| y finalmente, con la Codificación.

ISSD

-Des-
Desarrollo de
Software

MÓDULO
DIDÁCTICO
2023

¡Qué buen programa!

¡Bienvenidos a la primera clase de Programación 1!

El objetivo fundamental de la materia “Programación 1” es que puedas resolver problemas de distinta índole (matemáticos, administrativos, gráficos, contables, etc.) empleando como herramienta la computadora.

Tené en cuenta que para llegar a ser programador debés recorrer un largo camino en donde cada tema es fundamental para conceptos futuros. Por eso es importante que no dejes temas sin entender y relacionar.

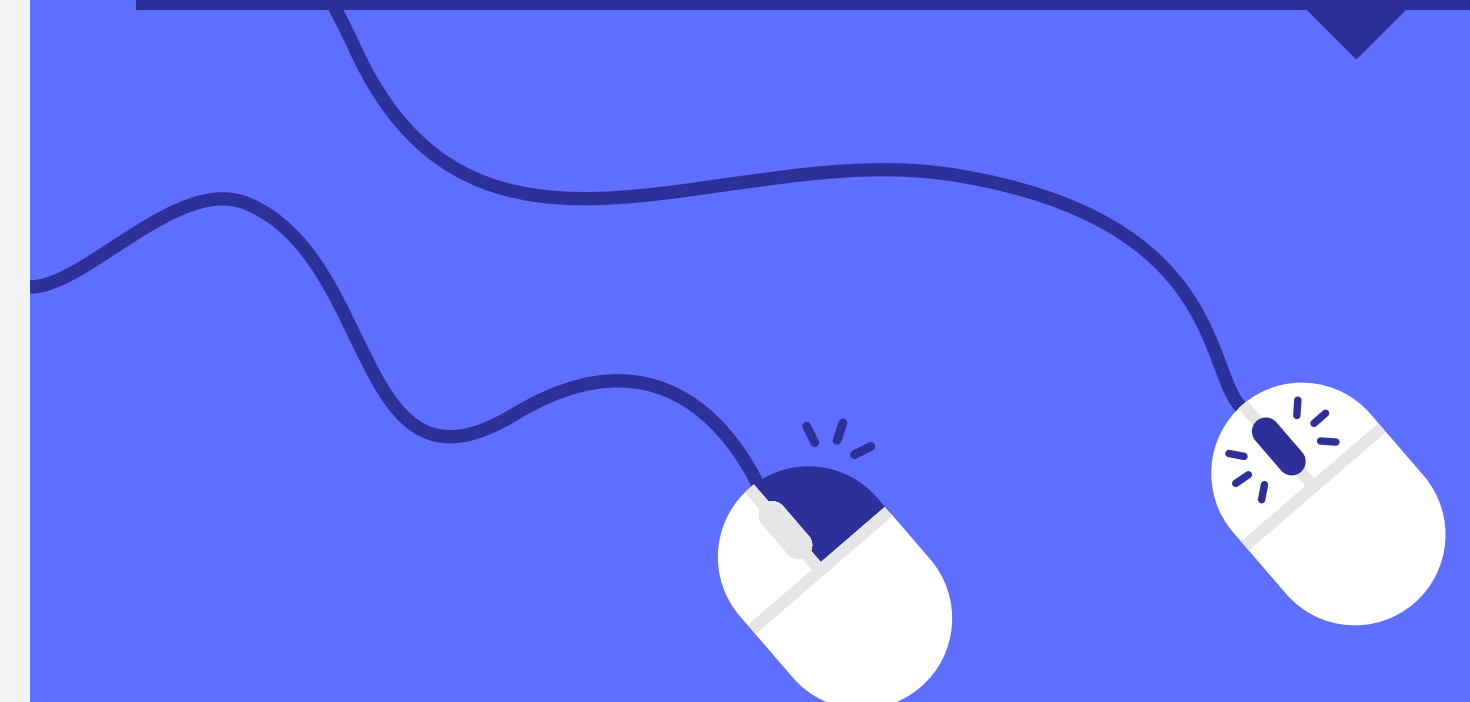
La programación, requiere un estudio metódico y ordenado y en AED1, se sientan las bases para todas las otras materias orientadas a la programación.

Como la programación es una actividad nueva para vos -ya que no hay en los estudios secundarios una materia parecida- te recomendamos tener paciencia cuando no puedas resolver los ejercicios por completo.

Sin embargo, es de fundamental importancia que dediques tiempo al análisis individual de los problemas y que ante la presencia de dudas consultes con tu tutor.

Si querés realizar las pruebas del código en tu pc, te recomendamos hacer click en el botón ⬇
Se encuentra ubicado junto a cada código ejemplificado. De esta manera, podrás copiar y pegar el texto puro para que evitar errores en la ejecución.

Si accedes al módulo desde un navegador, te recomendamos abrir los hipervínculos en una nueva pestaña presionando la rueda central del ratón, o bien usando el botón derecho y seleccionando la opción correspondiente.



Este material de estudio no es una fuente de información completa o definitiva, sino tan sólo una **guía general**. Un estudio profundo de esta asignatura, debería ser complementado con libros y manuales de consulta, además de una muy abundante ejercitación en la resolución de problemas y trabajo sobre la computadora.

En esta primera clase intentaremos, por un lado, introducirte a lo que significa programar una computadora, diferenciando el **diseño del algoritmo** de la **codificación** propiamente dicha.

Además trabajaremos para que puedas valorar la importancia de analizar a fondo el **problema a resolver** e **identificar sus elementos centrales** antes de proceder a programar su solución.

Por otro lado, trataremos que entiendas cómo un **diagrama de flujo** puede simplificar la construcción del **algoritmo** de resolución de un problema y también cómo utilizar **Python** como lenguaje de programación.

Para organizar estos contenidos los hemos dividido en cinco temas:

- | Primero vas a estudiar **¿qué significa programar?**
- | Luego analizaremos **¿cómo se programa una computadora?**
- | Posteriormente responderemos a la pregunta **¿por qué vamos a aprender Python?** y a través de esto estudiaremos **las lógicas de programación, ¿qué es un algoritmo?**, los **diagramas de flujo** y los **componentes más utilizados para confeccionarlo**.
- | En cuarto lugar, trabajaremos sobre unos **primeros ejercicios de ejemplo** a partir de los cuales estudiaremos las **variables** (sus **nombres válidos**, los **tipos de variables**) y los **operadores aritméticos**.
- | Finalmente, veremos la **instalación de “Python”** y realizaremos **nuestro primer programa**

En esta materia en particular, vas a sentar las bases que te permitirán ser un Analista de Sistemas, Programador o Desarrollador de Software de alto nivel.

Cuando abordes cada nuevo tema, tengas que resolver una situación problemática propuesta o profundices en los conceptos centrales de la lógica y el lenguaje de programación, no pierdas de vista que tu esfuerzo vale la pena, y que aquí estamos para ayudarte y acompañarte.

Antes de comenzar con la primera clase, leé estas dos noticias publicadas en medios de prensa.

¿Cuál es el empleo más buscado, en el que faltan profesionales y paga sueldos de hasta USD 10.000 al mes?

Un informe reveló hacia dónde va el mercado y qué necesidades tiene las empresas de servicios basados en el conocimiento.

La industria del software es una de las actividades que más empleo genera en Argentina y, de acuerdo con el informe de la Guía Salarial elaborada por **Adecco Argentina**, la demanda de perfiles ha cambiado desde el inicio de la pandemia. Los perfiles digitales hoy tienen un protagonismo importante. Según datos de la **Cámara de la Industria Argentina del Software de Argentina (Cessi)**, históricamente en el país hay un déficit de 5.000 puestos sin cubrir, cifra que puede convertirse en 15.000 si se contempla, además, la demanda insatisfecha de posiciones IT en otros sectores de la economía. (...)

Infobae – 05/01/2023

¿Por qué no se consiguen programadores?

Empresarios y compañías los buscan desesperadamente para trabajar dentro de sus proyectos y empresas. Sin embargo, la gente no estudia masivamente programación.

Hace unas semanas escuché en una entrevista radial a un *reconocido empresario de los medios* decir que no conseguía programadores para uno de sus exitosos portales. Así como él, son **muchos los empresarios y las compañías que día a día buscan desesperadamente programadores** para trabajar dentro de sus proyectos y empresas.

Sin embargo, encontrarlos no es una tarea fácil: *los programadores escasean*. Entonces, la cuestión es **entender por qué la gente no estudia masivamente programación**, si es sabido que se trata de una profesión que asegura un trabajo de por vida (no hay programadores desempleados), paga sueldos

altísimos y permite trabajar desde cualquier parte del mundo cobrando en dólares.

Después de años formando gente para entrar a trabajar en esa especialidad encuentro **cuatro grandes posibles causas** de por qué esto sucede. Yendo de general a particular.

La primera es que **la mayor parte de la gente no está al tanto de los beneficios exclusivos de trabajar en programación.** Puede que muchos lo hayan escuchado infinidad de veces, pero para una gran parte de la población aún no está totalmente claro cuán excepcional es la industria tecnológica a nivel laboral.

La segunda causa es que quienes saben los privilegios de esta profesión, **piensan que está reservada para “genios”** y que ellos no pueden programar. Hay preconceptos instalados que ahuyentan a muchísima gente, del estilo que hay que ser bueno en matemática, o ser muy inteligente para poder programar. Esto definitivamente no es así, **cualquiera puede aprender a programar.**

La tercera razón es que **no hay rumbos claros hacia el empleo.** A quienes estén al tanto de las oportunidades y se animen a aprenderlo les suele resultar difícil encontrar un camino que los forme a nivel profesional.

La primera opción habitual es **ir por una carrera universitaria. Pero éstas son muy largas,** (duran años y terminan por perder a un gran porcentaje de sus alumnos en el camino) y muchas veces están desactualizadas.

Por otro lado, hubo una **explosión de cursos de programación de todo tipo y color,** pero la enorme mayoría de ellos son básicos y quedan muy lejos del nivel mínimo que requiere la industria para empezar una carrera.

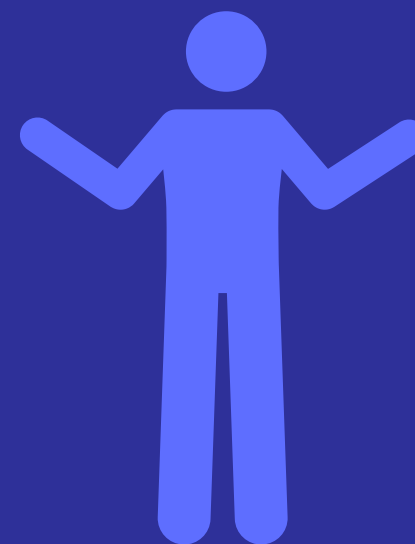
La cuarta causa es que para quienes dan con el curso correcto, **la inserción laboral es difícil, y requiere mucha práctica y perseverancia:** Si bien hay gran cantidad de puestos de trabajo sin cubrir, el nivel mínimo de programación que hay que tener para entrar a trabajar es muy alto. Llegar a ese nivel es difícil, por lo que si bien existen cada vez más recursos para aprender, cuesta encontrar un norte y dedicarle cientos de horas a un camino que demanda un enorme esfuerzo mental.

Perfil (15/2/2022)

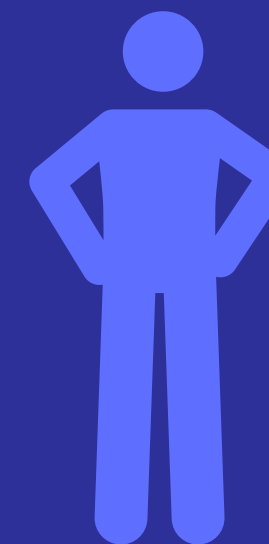
Como habrás notado, en Argentina-y en gran parte del mundo- se repite una paradoja: el área de la Programación y los Sistemas es la que requiere cada vez más profesionales capacitados y la que paga mejores salarios, pero a su vez los estudiantes eligen cada vez menos estudiar estas carreras.



Las proyecciones a futuro son claras y contundentes: has elegido una carrera clave para tu desempeño y crecimiento profesional, con demanda asegurada y altas posibilidades de inserción laboral de calidad.



Además, la Programación te permitirá trabajar en proyectos creativos, en relación constante con personas conformando equipos diversos –muchas veces transnacionales-, en ambientes llenos de desafíos y que demandan una actualización permanente.



Introducción a AED1

¡Bienvenidos al maravilloso mundo de la Programación!

Muchos aparatos de uso cotidiano, contienen algún tipo de programa: desde el lavarropas automático, hasta el microondas o la heladera. Estos dispositivos, siguen una secuencia de pasos que alguien determinó para que logren cumplir con su función.

Por ejemplo:

Un lavarropas automático al ser encendido, en primer lugar trabará la puerta activando una cerradura magnética. Luego, abrirá la válvula para cargar el agua de la red domiciliaría. Una vez detectado el nivel adecuado, la cerrará y encenderá la resistencia para calentarla. Alcanzada la temperatura especificada, apagará la resistencia y encenderá el motor, generando un ciclo de movimientos, alternando el sentido de giro. Pasado cierto tiempo, se abrirá la bomba de expulsión de agua, y se desalojará el tambor. Una vez vacío, el motor generará durante algunos minutos, un giro enérgico a muchas revoluciones por minuto, con el fin de centrifugar la ropa.

Por último, destrabará la cerradura magnética para que se pueda quitar la ropa ya lavada.

A diferencia de un lavarropas o un microondas (aparatos concebidos para cumplir una sola función), **una computadora es un dispositivo totalmente flexible, capaz de ejecutar programas muy complejos y de naturaleza extremadamente variada.**

Con el mismo dispositivo, podremos procesar millones de datos, buscar información en Internet, escuchar música, ver una película o jugar... e incluso ***¡todas estas actividades al mismo tiempo!***

Como programadores, ***seremos capaces de escribir las instrucciones que necesitamos***, de modo tal que la computadora lleve a cabo una secuencia de acciones, con el fin de resolver problemas de diferentes ámbitos.

Resumiendo: un programa es un conjunto de instrucciones detalladas que le dirán al dispositivo qué hacer exactamente, paso a paso.



Hay que tener en cuenta que para llegar a ser un programador se debe recorrer un largo camino donde cada tema es fundamental para conceptos futuros.

| Es importante no dejar temas sin entender y relacionar

La programación se diferencia de otras materias que has estudiado, como por ejemplo podría ser la historia, porque ésta **requiere un estudio metódico y ordenado** (en historia se puede estudiar la edad media sin tener grandes conocimientos de la edad antigua, esto no pasa con el aprendizaje de la programación)

A lo largo de estas clases, iremos construyendo paso a paso y con ejercicios un conocimiento que nos permitirá escribir instrucciones para resolver problemas.

Para ello, es de fundamental importancia que le *dediquemos tiempo a la resolución de los problemas* planteados a lo largo de las clases *y luego a su análisis* que nos permita desarrollar y potenciar nuestra destreza de programación



Desempeño 1

| Siguiendo el ejemplo anterior referido al lavarropas, escribí en una hoja, cómo sería el programa de otro dispositivo de uso cotidiano, como el microondas o la heladera.

¿Qué es un programa?

Un programa es un conjunto de instrucciones que entiende una computadora para realizar una determinada actividad.

Todo programa tiene un objetivo bien definido:

- | Un procesador de texto es un programa que permite cargar, modificar e imprimir textos
- | Un programa de ajedrez permite jugar al ajedrez contra el ordenador u otro contrincante humano.

La actividad fundamental del programador es resolver problemas empleando el ordenador como herramienta fundamental

Un programa es parecido a una receta: un grupo de instrucciones (*Algoritmo*) que le indican al cocinero cómo preparar un determinado plato.

Describe los ingredientes (*los datos*) y la secuencia de pasos (*el proceso*) necesarios para convertir los ingredientes (*entrada*) en una torta (*salida*).



Desempeño 2

| Elegí una comida sencilla que sepas cocinar (puede ser incluso preparar un sándwich o hacer un huevo frito), y escribí en una hoja paso a paso cómo realizarla.

Es muy importante que tengas en cuenta que la persona que va a seguir tu receta, no tiene ninguna experiencia, por lo tanto, no ahorres detalles al describir el proceso.

¿Qué es un algoritmo?

Un algoritmo son los **pasos secuenciales**, son los que deben ser ejecutados uno después de otro, y los **pasos ordenados** son los que deben llevar un orden obligatorio.

Como puede notarse, **lo que permite un algoritmo es lograr un objetivo.**

Volviendo el ejemplo del lavarropas automático, la secuencia de pasos ordenados que describimos constituiría un algoritmo, cuya finalidad será lavar la ropa.

Debido a la dificultad inherente a la construcción de un algoritmo informático, muchas personas optan por construir *diagramas de flujo*. Estos, pueden ayudar a resolver cualquier algoritmo, antes de la codificación real en un lenguaje de programación específico.

A continuación vamos a aprender cómo aplicar esta herramienta gráfica.

¿Qué es un Diagrama de Flujo?

Los diagramas de flujo son *la representación gráfica de un ALGORITMO*.

Se les llama diagramas de flujo, porque los símbolos utilizados se conectan por medio de flechas para indicar la secuencia de operación.

La respuesta obtenida con el desarrollo de un diagrama, no es única, sino que es una de las tantas que se pueden obtener.

Cada persona tiene una forma de razonar distinta a los demás, por lo tanto, distintas personas pueden llegar a la solución de un mismo problema de diversas maneras, es decir, que *puede haber varias soluciones para un determinado problema*.

Las ventajas de usar diagramas de flujo son:

- | Favorecen la comprensión del proceso a través de mostrarlo como un dibujo. Un buen diagrama de flujo reemplaza varias páginas de texto.
- | Permiten distinguir los problemas y las oportunidades de mejora del proceso.
- | Se identifican los pasos redundantes, los conflictos, las responsabilidades y los puntos de decisión.



Inicio/Fin del diagrama



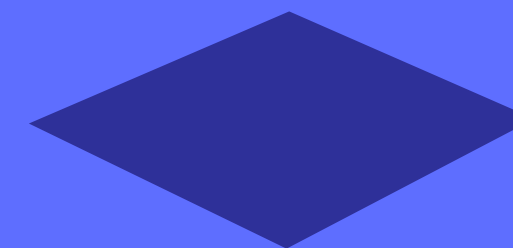
Entrada de Datos



Salida de Datos



Operación

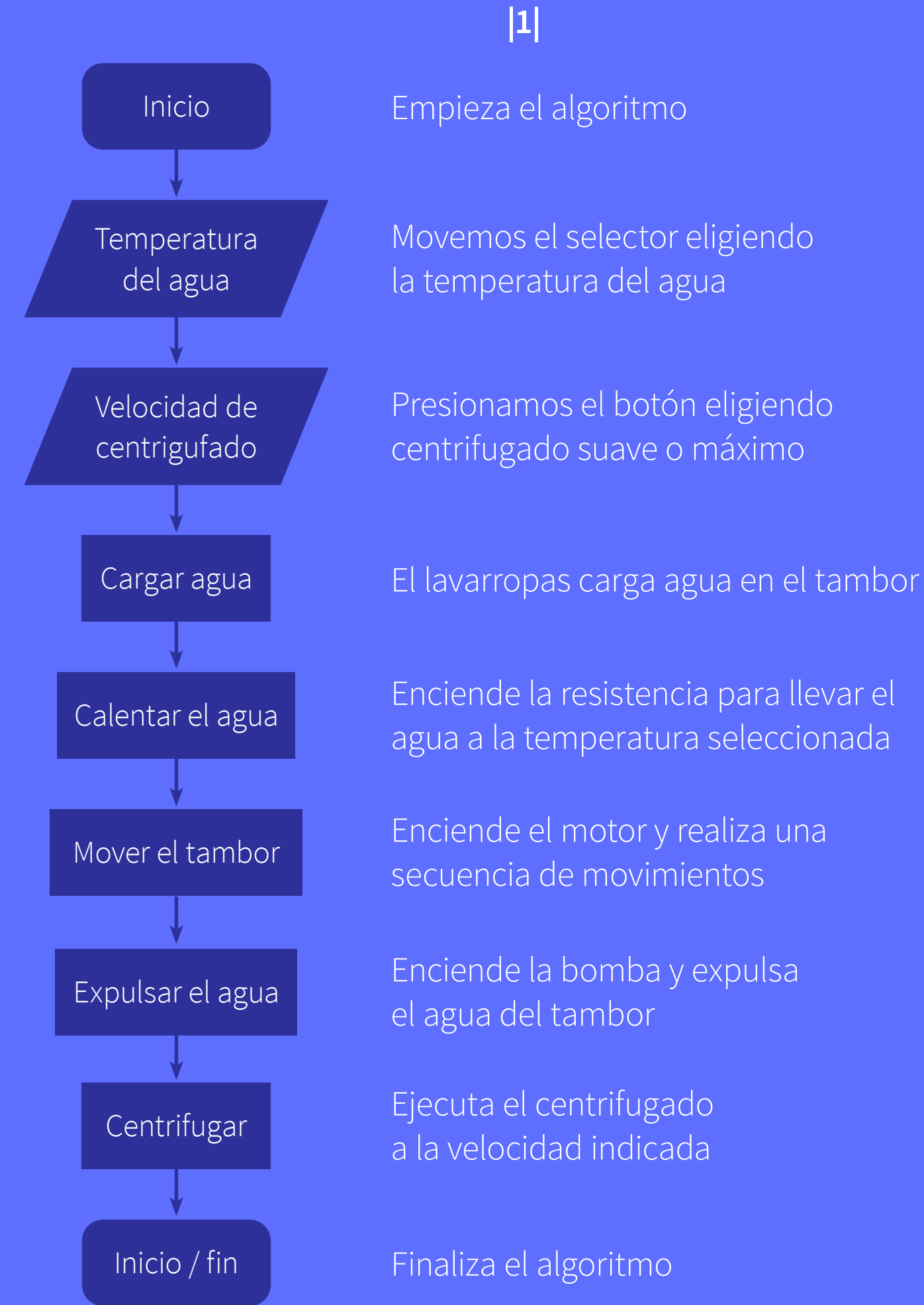


Condición

- Si hacemos un análisis, todo problema está constituido por:
- a) **Datos conocidos:** Datos con los que se cuenta al comenzar el problema.
 - b) **Proceso:** Operaciones a realizar con los datos conocidos.
 - c) **Información resultante:** Es la información que se obtiene del proceso y nos permite resolver el problema.

Esta forma de expresar un problema identificando los **datos conocidos, sus procesos e información resultante** puede llegar a ser engorrosa en situaciones complejas; es por eso que resulta enormemente más práctico y efectivo representar los pasos para la resolución del problema mediante **un diagrama de flujo que podemos comprender con un solo vistazo**.

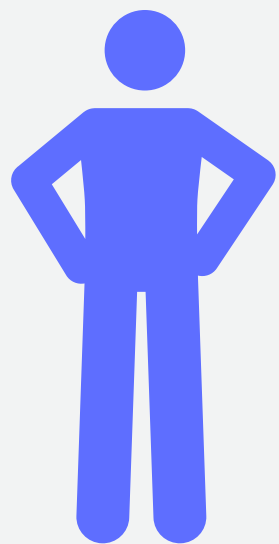
A continuación te compartimos un ejemplo de un diagrama acerca del programa de un lavarropas para que puedas comenzar a interpretar los diagramas de flujo.[1](#)



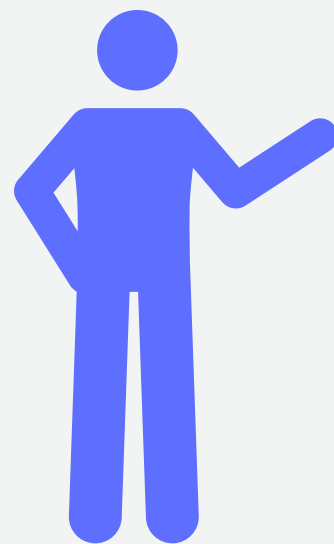
Primeros ejercicios de ejemplo

¿Te quedó claro de qué se trata un diagrama de flujo?

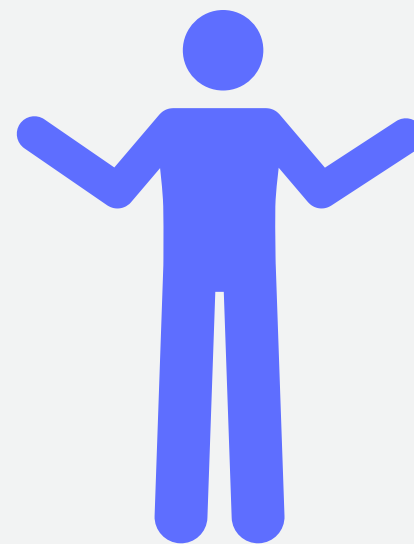
Vamos a aplicarlo en la resolución de un problema simple: nuestro primer ejercicio de ejemplo.



Es muy importante que de aquí en más, no te limites a leer los ejemplos, sino que los pruebes en tu computadora.



Practicar, probar y encontrar los errores de todos los ejemplos y de todos los ejercicios propuestos, es la única forma de aprender a programar. Y no te olvides de recurrir al tutor en cuanto veas que algo no funciona.



Ejemplo 1

Cálculo del salario de trabajo

Calcular el sueldo mensual de un operario conociendo la cantidad de horas trabajadas y el pago por hora.

Podemos identificar:

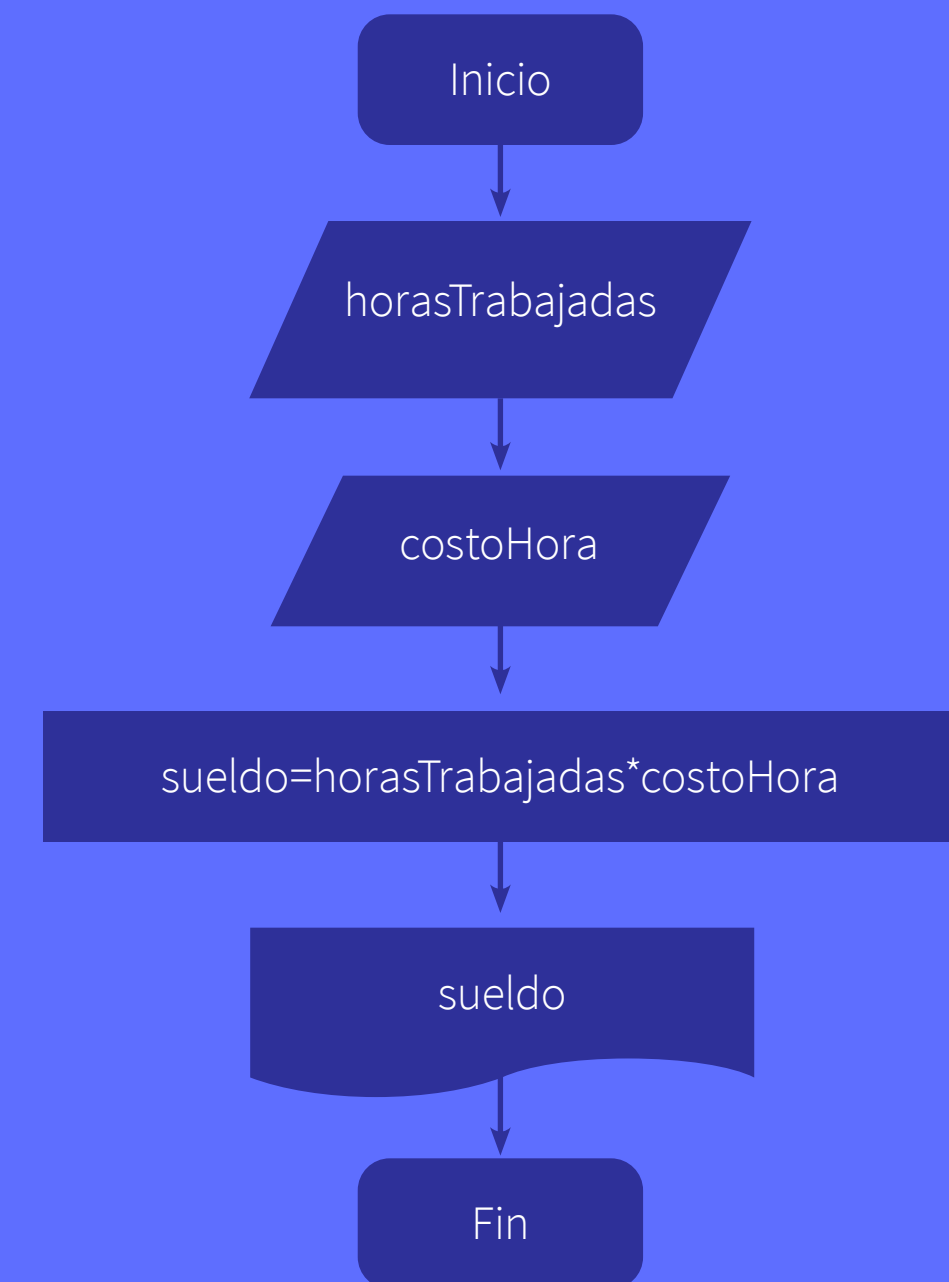
| **Datos conocidos:** Horas trabajadas en el mes y Pago por hora.

| **Proceso:** Cálculo del sueldo multiplicando la cantidad de horas por el pago por hora.

| **Información resultante:** Sueldo mensual.

En el ejemplo graficado a nuestro costado derecho podremos visualizar el diagrama de flujo [\[2\]](#)

[2]



Introducción a las variables

Una variable es un depósito donde hay un valor. Consta de un nombre y pertenece a un tipo de dato. Tomando como referencia el ejemplo 1, podemos visualizar que en ella tenemos 3 variables

- 1 | La variable **horasTrabajadas** almacena la cantidad de horas trabajadas por el operario
- 2 | La variable **costoHora** almacena el precio de una hora de trabajo
- 3 | La variable **sueldo** almacena el sueldo a abonar al operario

Tipos de variable.

Una variable puede almacenar:

- | Valores Enteros (100, 260, etc.)
- | Valores Reales (1.24, 2.90, 5.00, etc.)
- | Cadenas de caracteres ("Juan", "Compras", "Listado", etc.)

Elección del nombre de una variable

Al momento de definir el nombre de una variable, te sugerimos elegir el nombre de variables representativas.

En el ejemplo el nombre “**horasTrabajadas**” es lo suficientemente claro para darnos una idea acabada sobre su contenido. Siempre podemos darle otros buenos nombres. Algunas veces no son tan representativos, por ejemplo “**hTr**”.

Posiblemente cuando estemos resolviendo un problema dicho nombre nos recuerde que almacenamos las horas trabajadas por el operario, pero cuando pase el tiempo y leamos el diagrama probablemente no recordemos ni entendamos qué significa hTr.

| También debes tener en cuenta que los códigos muy a menudo requieren **mantención** o que una **persona externa lo revise**, por lo que **es prudente colocar nombres fáciles de identificar**.

Instalación de *Python*

Ya estamos casi listos para empezar a programar en Python. Para poder hacerlo, necesitamos descargar e instalar el entorno de programación que usaremos en esta materia.

Para la descarga del lenguaje Python lo hacemos del sitio: python.org. (Descargar la versión más actual. La 3.11 es la última al momento de la redacción de este material, así que es la que verás en las capturas de pantalla. Sin embargo, puede ser que te toque usar una versión más nueva)|3|

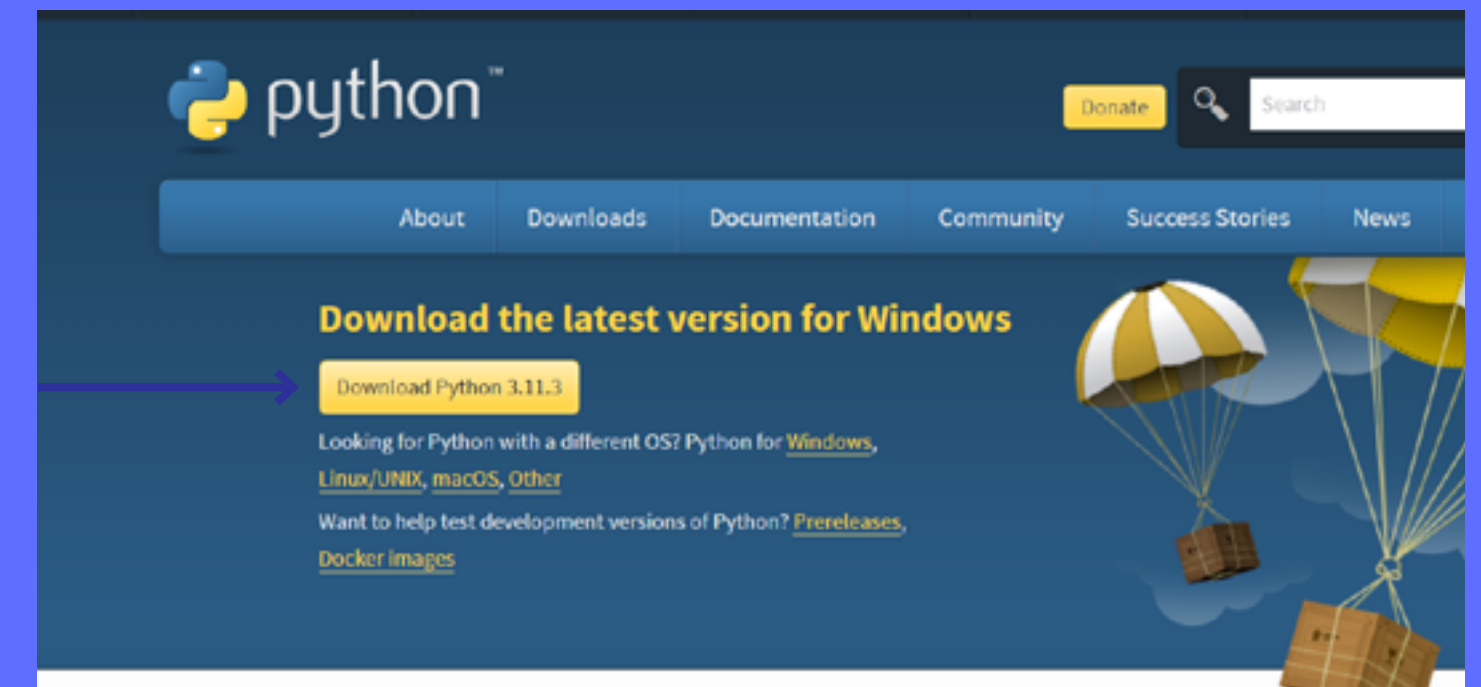
Elegí **Descargar con el navegador** y esperá hasta que se complete la descarga al 100%. Luego, hacé click en el archivo ya descargado para comenzar el proceso de instalación. |4|

No te olvides de tildar la opción (**Add Python3.7 to PATH**) antes de comenzar el proceso de instalación. Luego, para que el proceso de instalación comience, deberás hacer click en **"Install Now"**. Entonces, aparecerá el progreso de instalación en tu pantalla. |5|

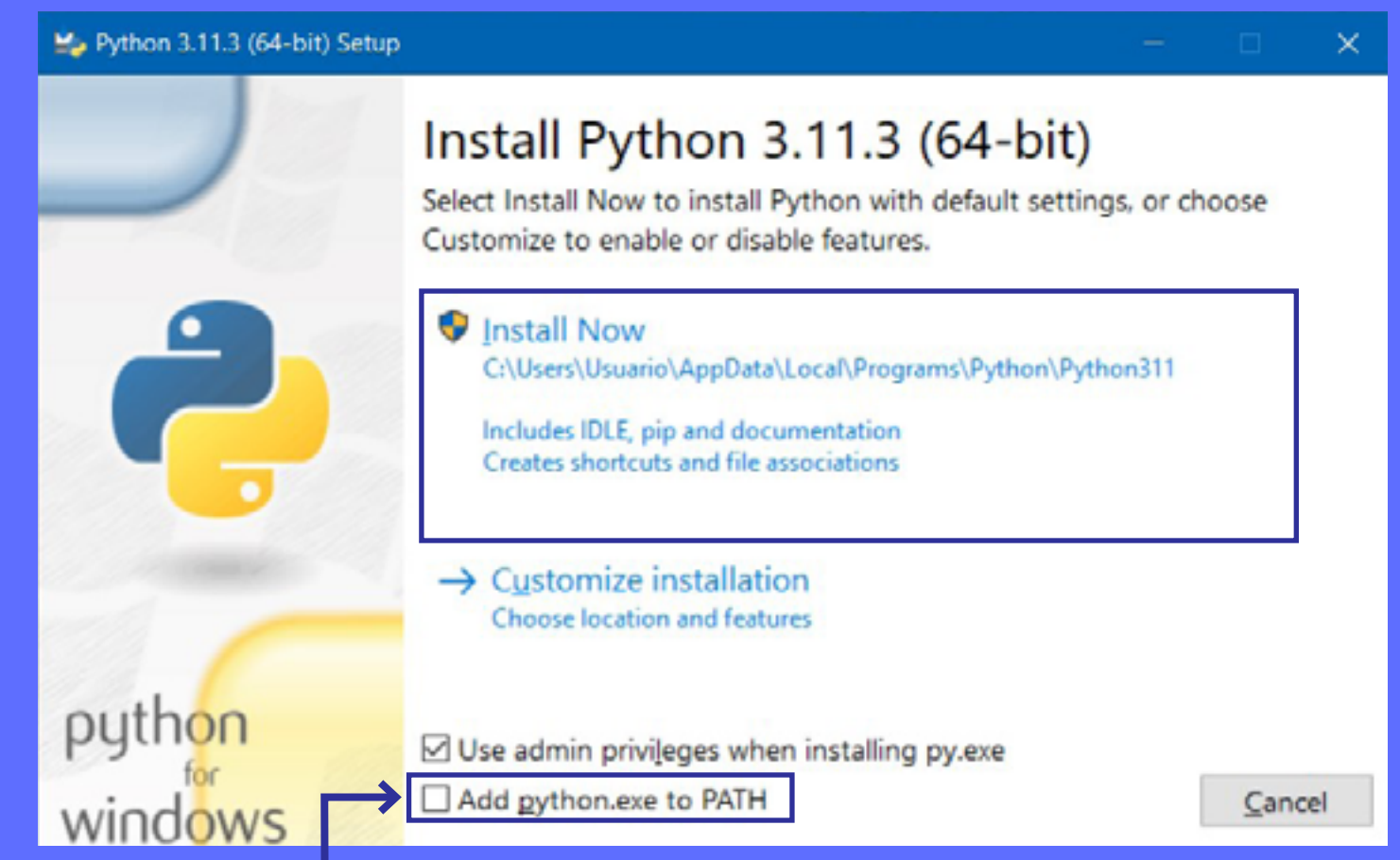
¿Ya se terminó de instalar? Entonces ahora estamos en condiciones de comenzar a escribir nuestro primer programa en Python.|6|

De ahora en adelante podrás acceder al programa dirigiendote al archivo **"IDLE (Python 3.10)"** desde el menú de inicio en la carpeta "Python 3.10".

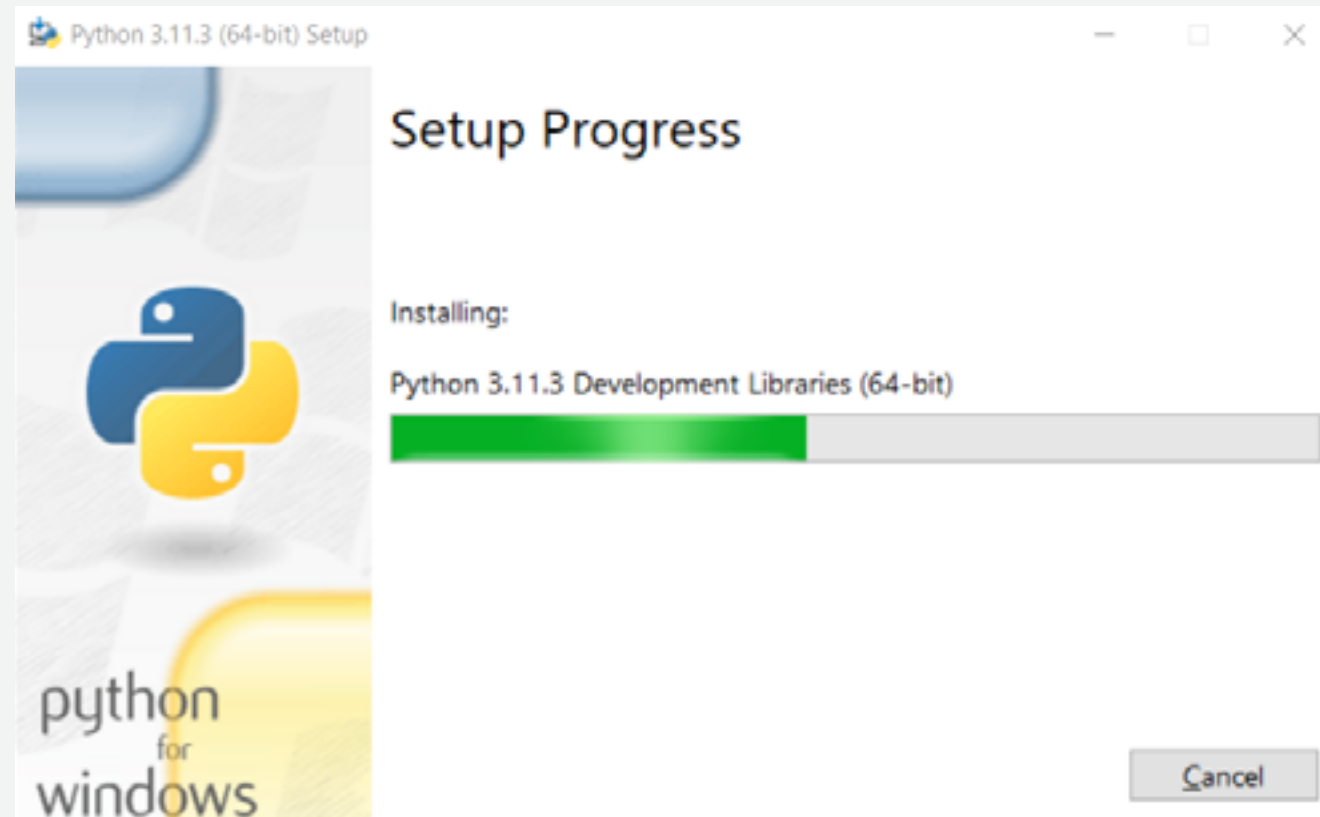
|3|



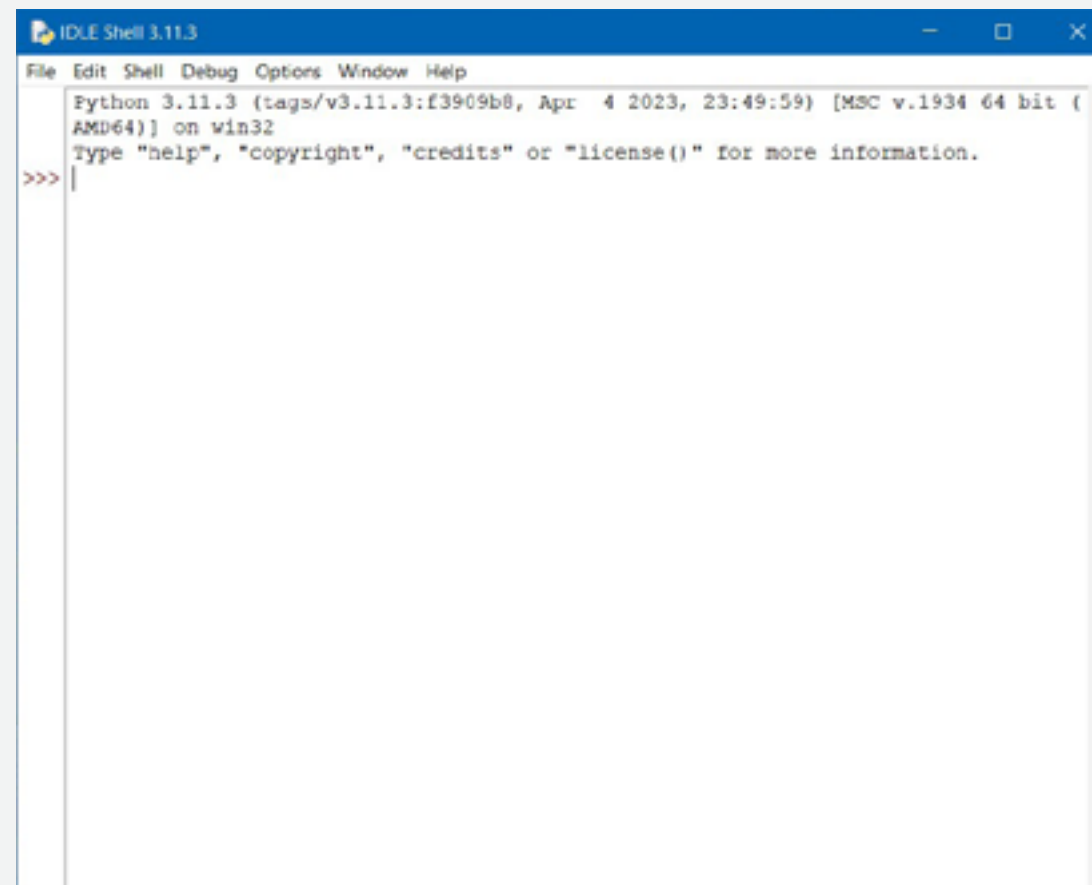
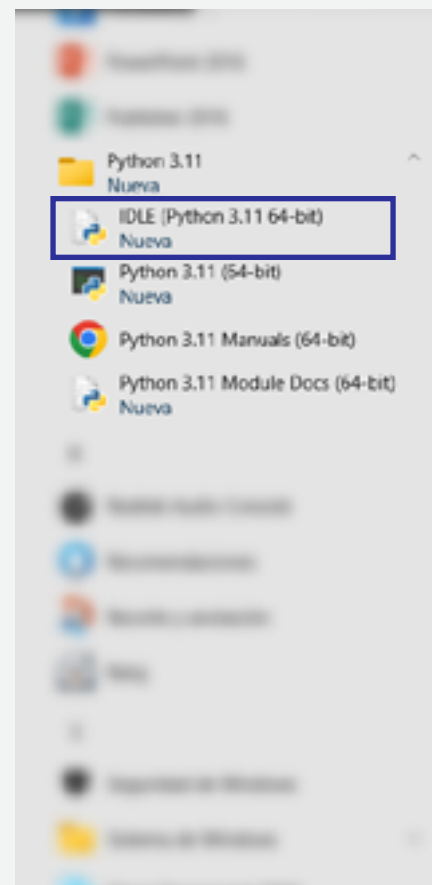
|4|



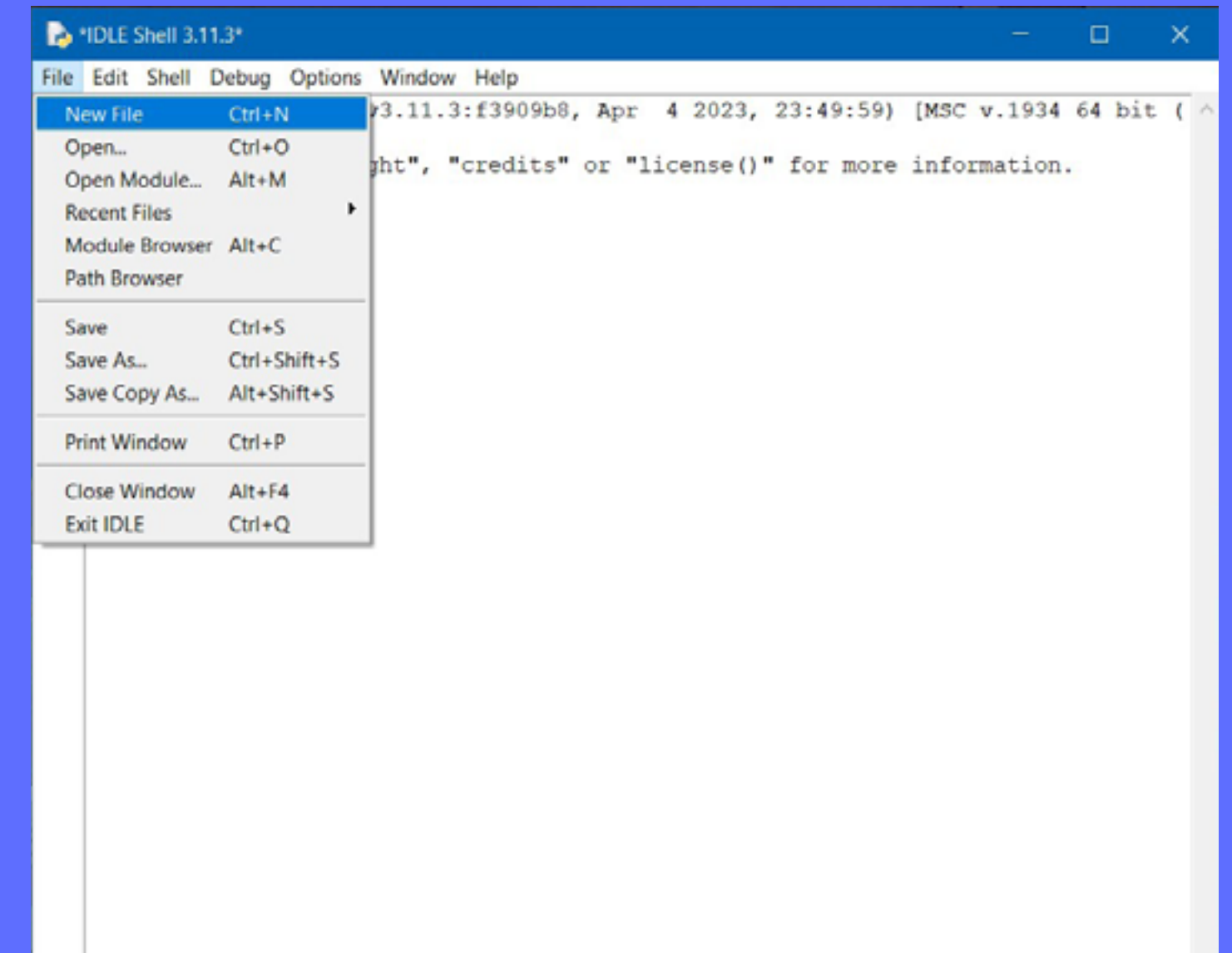
|5|



|6|



|7|



Nuestro primer programa en Python

Para crear nuestro primer programa seleccionaremos desde el menú de opciones "File" -> "New File" ó el atajo de teclado Ctrl + "N". [7]

Comenzaremos creando un texto que muestre el siguiente mensaje en pantalla:
"Hola mundo"

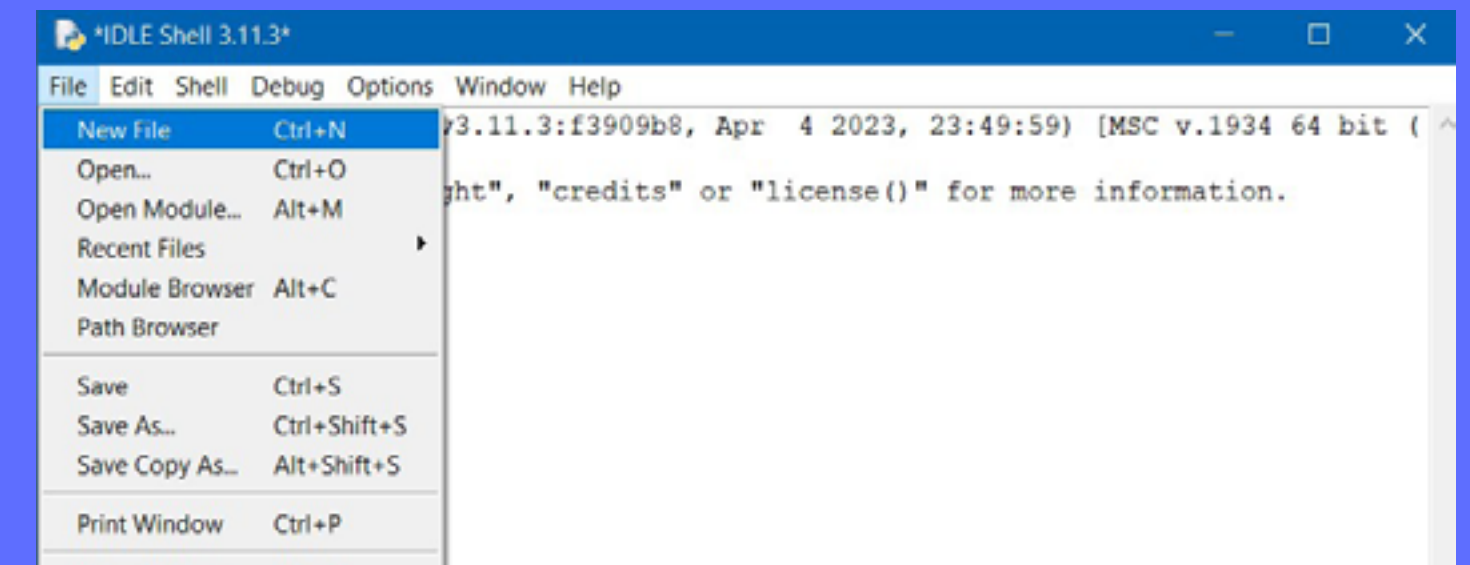
Para ello, iniciarás el programa y crearas un *nuevo archivo* como lo mostramos anteriormente. [8]

| En esta nueva ventana debemos codificar nuestro programa introduciendo la siguiente linea de código.

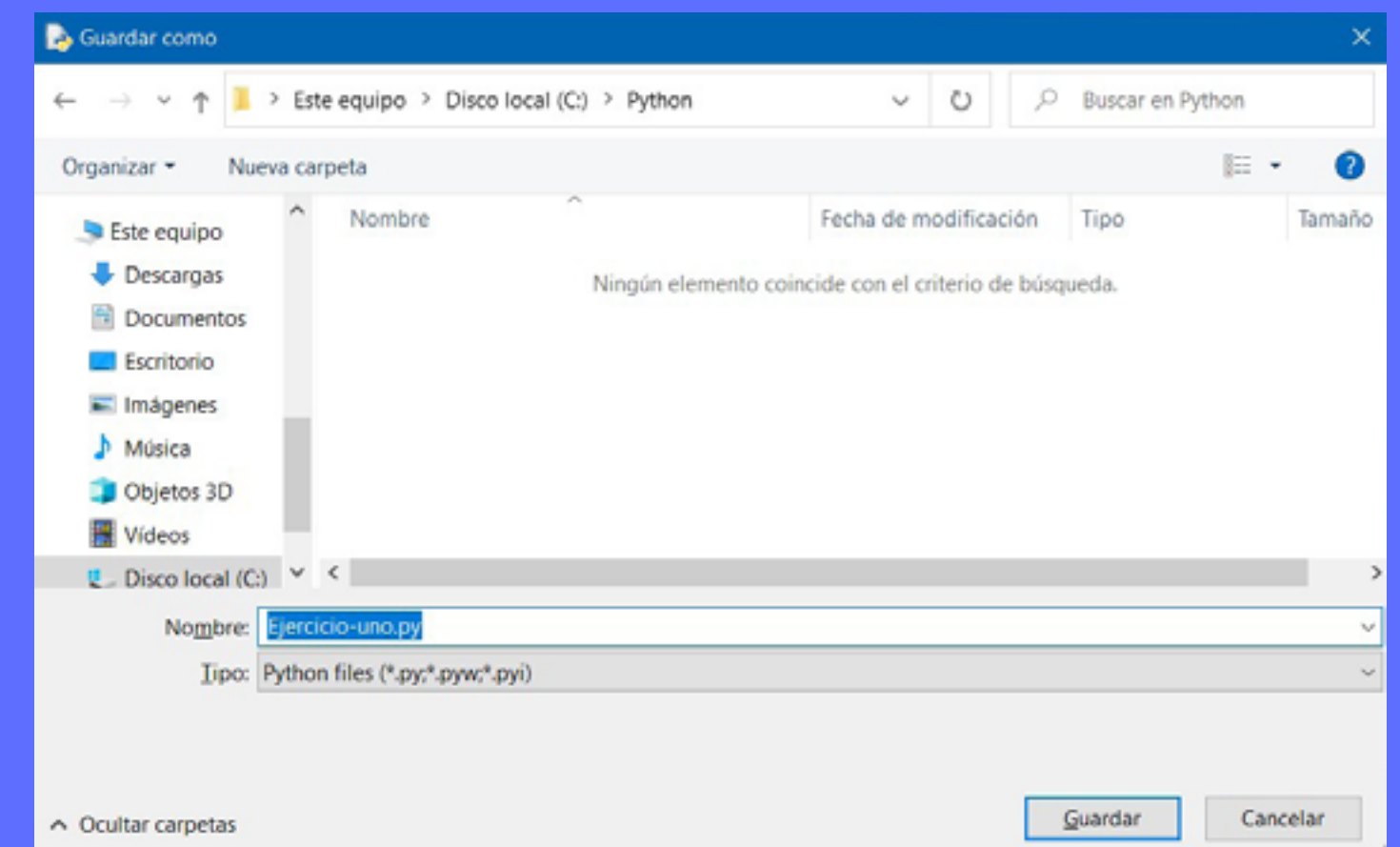
```
print("Hola mundo")
```

Procedemos a **guardar** en el "*Disco duro*" nuestro primer programa en Python seleccionando la opción "File" -> "Save", creamos un directorio donde almacenaremos cada uno de nuestros ejercicios que desarrollaremos durante este módulo. [9]

[8]



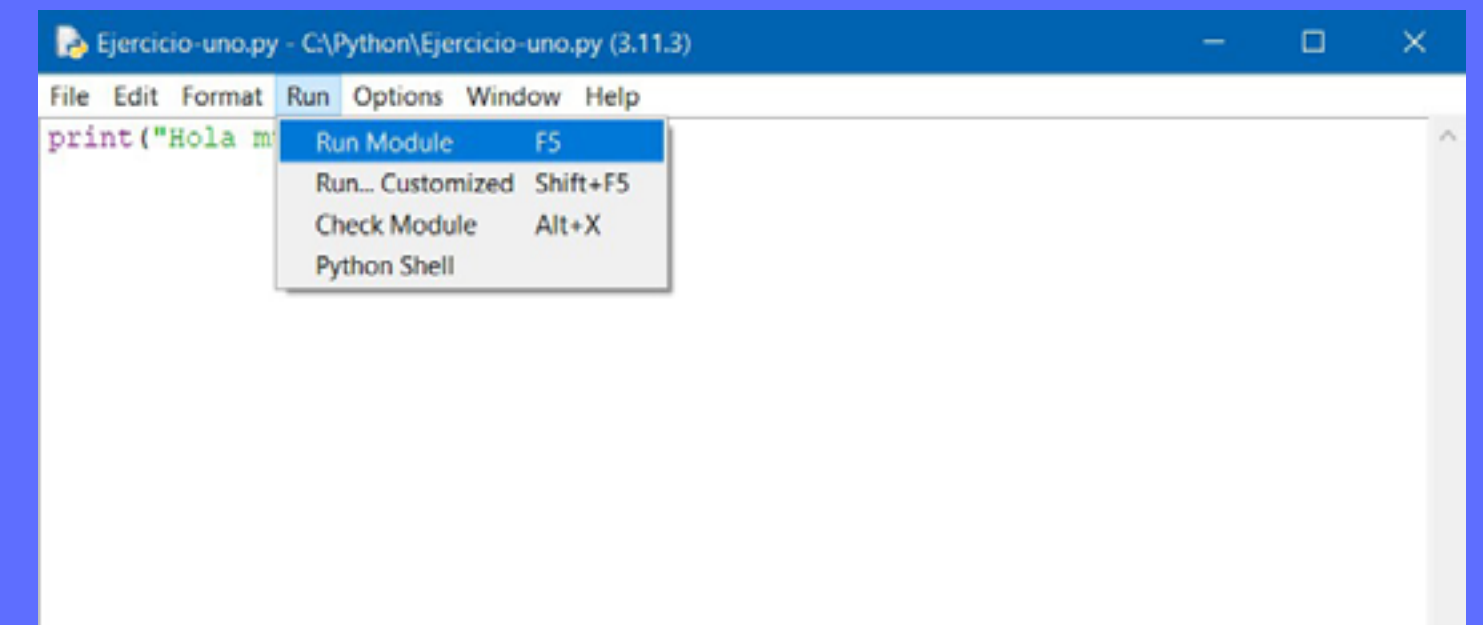
[9]



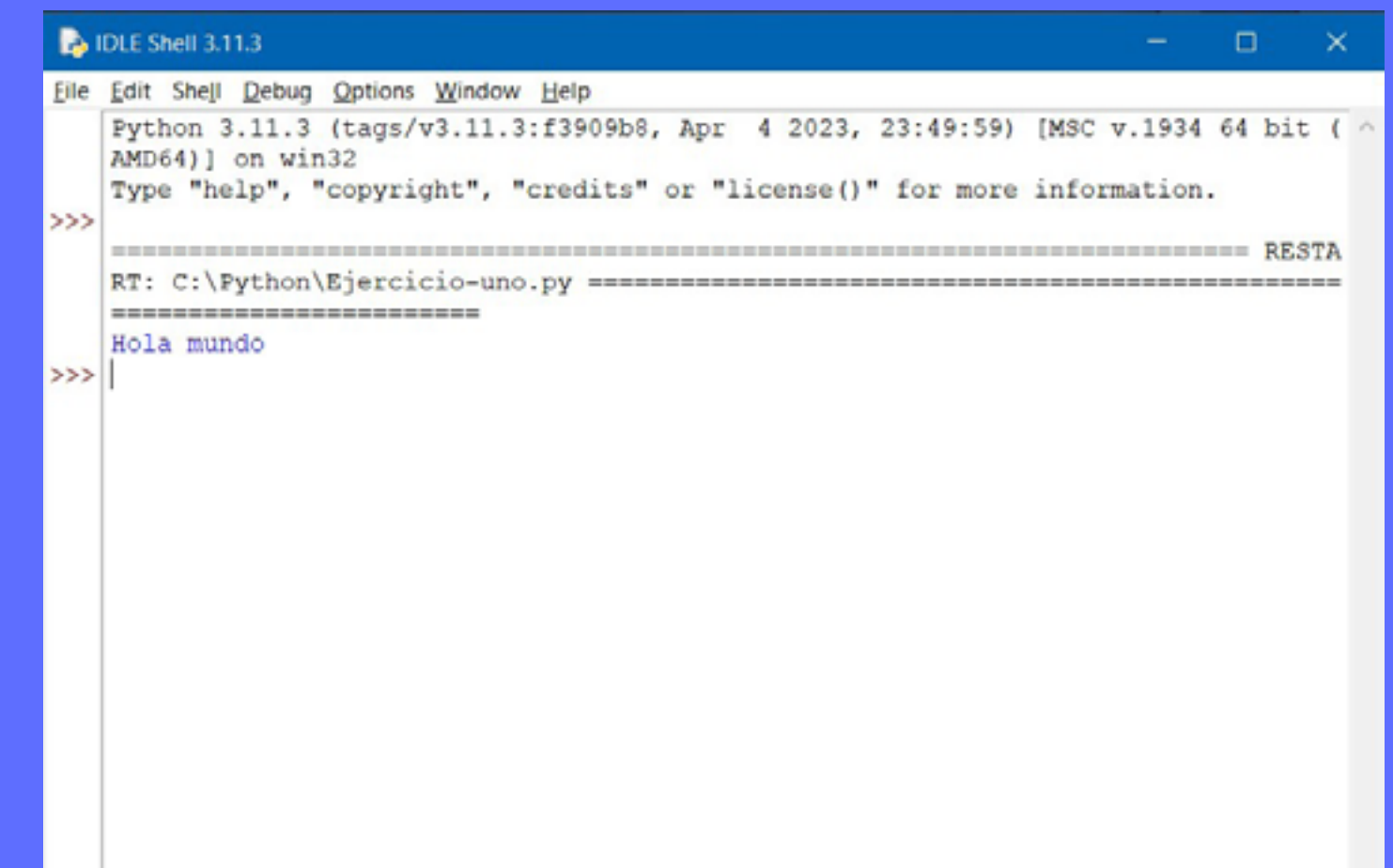
|10|

Para **ejecutar el programa** que codificamos debemos seleccionar desde la ventana de nuestro editor la opción "Run" -> "Run Module" [\[10\]](#).

Podremos ver en una nueva ventana emergente el resultado de la ejecución de nuestro primer programa en Python [\[11\]](#).



|11|



Ejemplo 2

Calculador de superficie de cuadrados

Hallar la superficie de un cuadrado conociendo el valor de un lado.

Empleando un editor de texto para Python (tales como "[VSCode](#)", "[Sublime Text](#)", "[Notepad++](#)", etc.) procedemos a crear nuestro segundo programa "ejemplo2.py", lo grabaremos en el disco duro y codificaremos en Python la siguiente solución al diagrama de flujo:

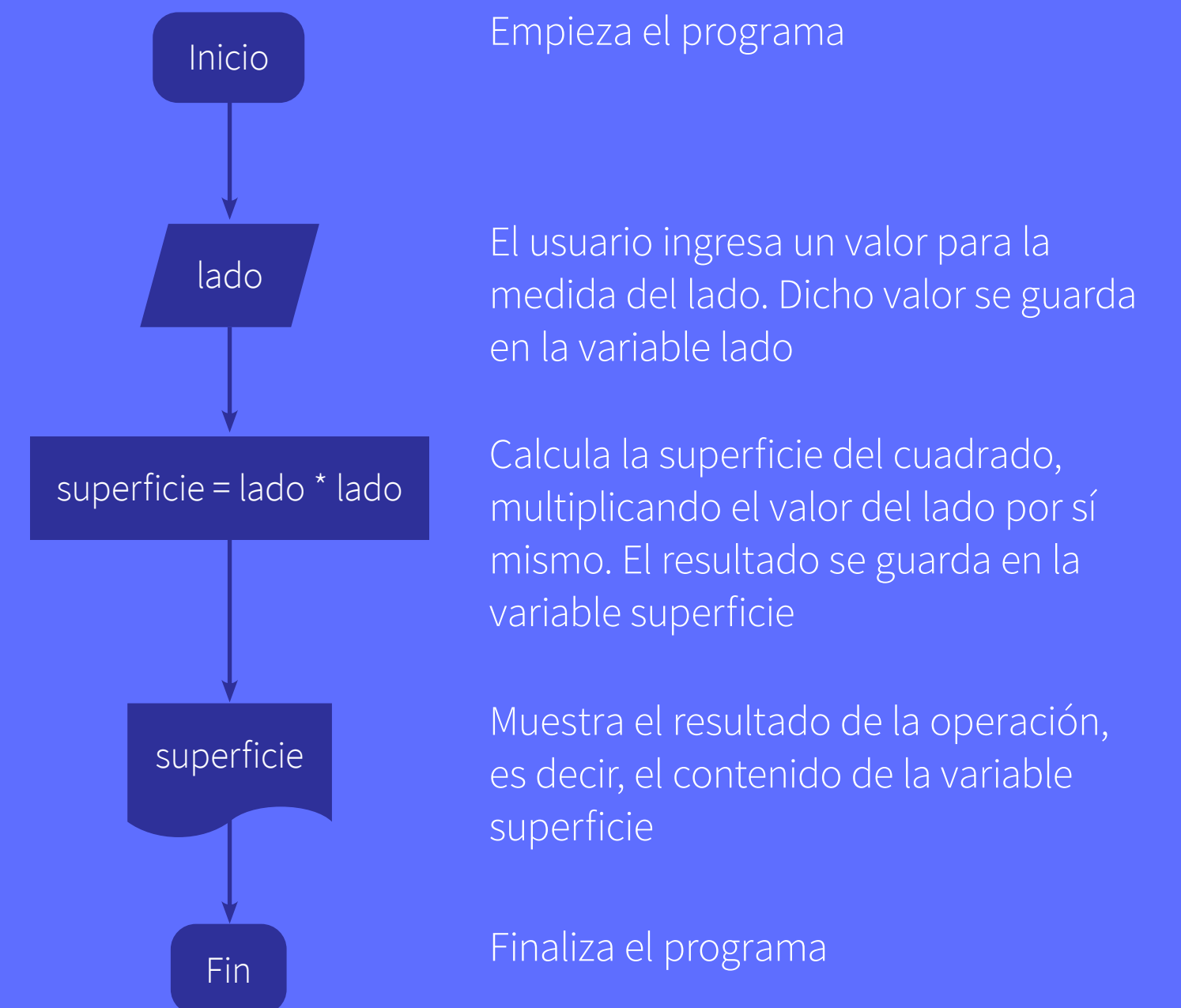
| Diagramación de ejemplo del cálculo de la superficie conociendo un lado. [|12|](#)

| Programa codificado: 

```
lado=input("Ingrese la medida del lado del cuadrado:")
lado=int(lado)
superficie=lado*lado
print("La superficie del cuadrado es")
print(superficie)
```

Si ejecutamos el programa "[Run](#)" -> "Run Module" podemos comprobar que se solicita el ingreso por teclado de la medida del lado del cuadrado y seguidamente nos muestra la superficie dependiendo del valor ingresado [|13|](#).

[|12|](#)



| Para el *ingreso de un dato* por teclado y mostrar un mensaje se utiliza la función `input`, esta función retorna todos los caracteres escritos por el operador del programa:

```
lado=input("Ingrese la medida del lado del cuadrado:")
```

| La variable *lado* guarda todos los caracteres ingresados pero no en formato numérico, para esto debemos llamar a la función `int`:

```
lado=int(lado)
```

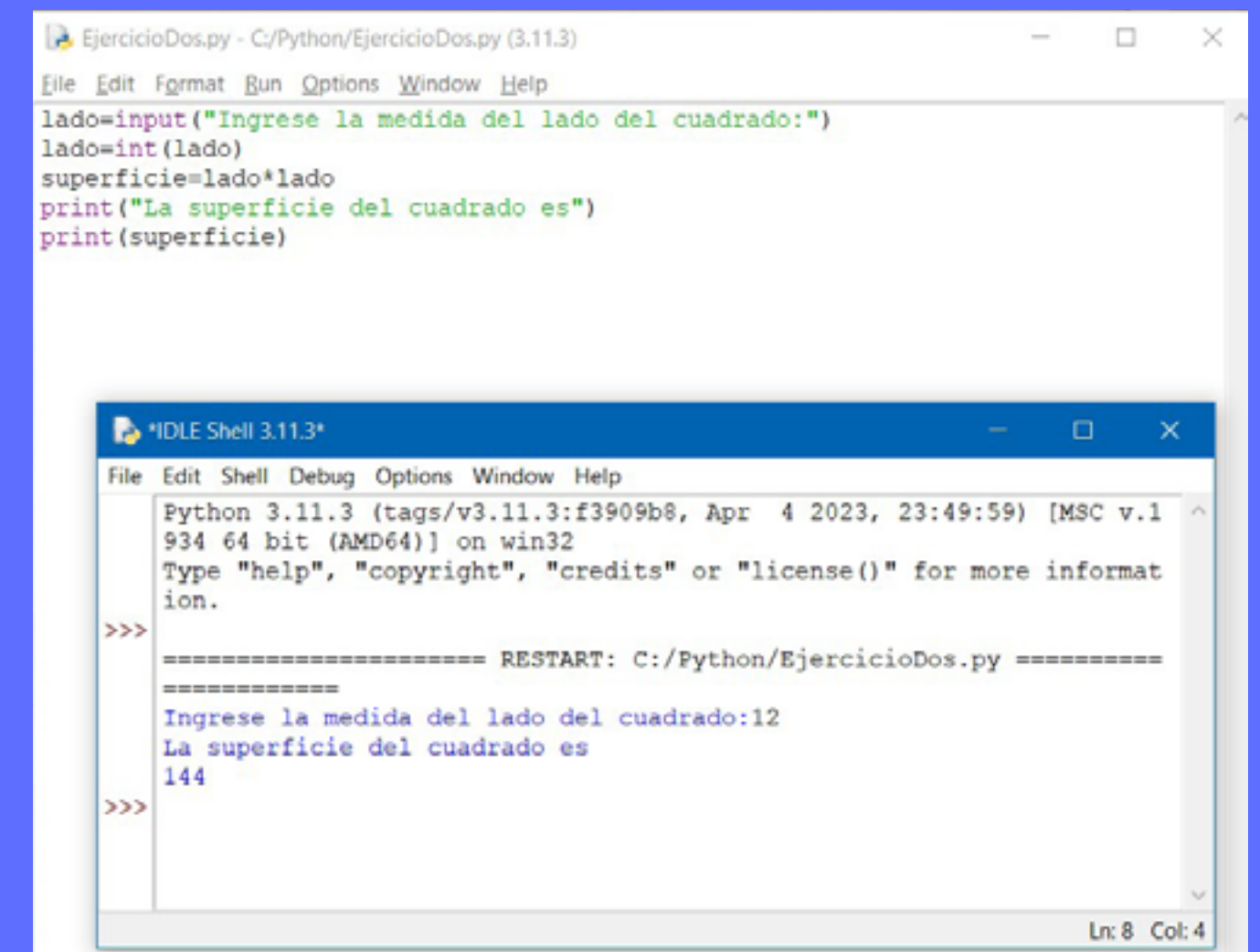
| Ahora se vuelve a guardar en la variable *lado* el valor que ingresó el operador pero en *formato entero* que posibilita hacer operaciones matemáticas con el mismo.

Un formato simplificado para ingresar un valor entero por teclado y evitarnos escribir las dos líneas anteriores es:

```
lado=int(input("Ingrese la medida del lado del cuadrado:"))
```

| Procedemos a efectuar el cálculo de la superficie luego de ingresar el dato por teclado y convertirlo a entero:

```
superficie=lado*lado
```



| Para mostrar un mensaje por pantalla tenemos la **función print** que le pasamos como parámetro una cadena de caracteres a mostrar que debe estar entre **simple o doble comillas**:

```
print("La superficie del cuadrado es")
```

| Para mostrar el contenido de la variable superficie no debemos encerrarla entre comillas cuando llamamos a la función print:

```
print(superficie)
```

¡Importante!

Todo el código debe escribirse en la misma columna, estará incorrecto si escribimos:

```
lado=input("Ingrese la medida del lado del cuadrado:")
lado=int(lado)
superficie=lado*lado
print("La superficie del cuadrado es")
print(superficie)
```

Hay algunas restricciones más, pero no te preocupes, las iremos aprendiendo a medida que avance el curso.

Algunas consideraciones

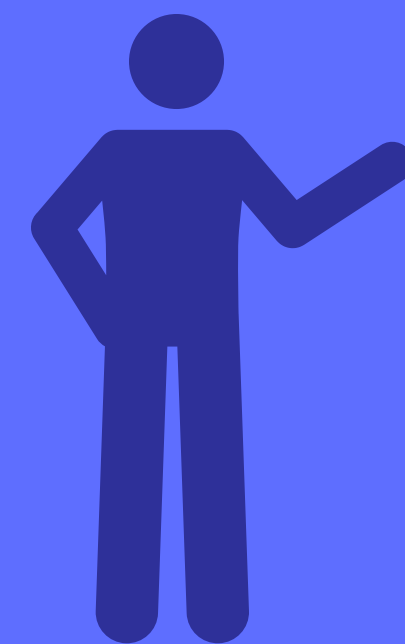
| **Python es sensible a mayúsculas y minúsculas**, por lo que no es lo mismo llamar a la función `input` con la sintaxis: `Input`.

| **Los nombres de variables también son sensibles a mayúsculas y minúsculas.**

Estaremos trabajando con dos variables distintas si inicializamos la variable "superficie" y luego hacemos referencia a "Superficie"

| Los nombres de variable **no pueden tener espacios en blanco, caracteres especiales o empezar con un número.**

¿Te confunden los errores?
No te preocupes,
aprenderemos a identificar
estos desaciertos y a poder
anticiparnos a todos ellos.



Errores sintácticos y lógicos

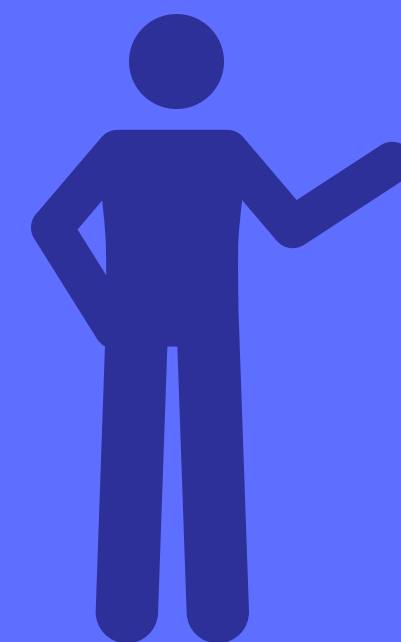
Modificaremos el problema anterior y le agregaremos adrede una serie de errores tipográficos. Este tipo de errores siempre *son detectados por el intérprete de Python*, antes de ejecutar el programa.

A los errores tipográficos, como por ejemplo indicar el nombre incorrecto de la función, nombres de variables incorrectas, falta de paréntesis, palabras claves mal escritas, etc, los llamamos errores **SINTÁCTICOS**.

Un programa no se puede ejecutar sin corregir absolutamente todos los errores sintácticos.

Existe otro tipo de errores llamados **ERRORES LÓGICOS**. Este tipo de errores en programas grandes (miles de líneas) son más difíciles de localizar. Por ejemplo, un programa que permite hacer la facturación, pero la salida de datos por impresora es incorrecta.

¿Te confunden los errores?
No te preocupes,
aprenderemos a identificar
estos desaciertos y a poder
anticiparnos a todos ellos.



Ejemplo 3

Errores comunes al codificar

Al momento de redactar el código para hallar la superficie de un cuadrado conociendo el valor de un lado es normal que cometamos errores.

A continuación te mostraremos ejemplos para que comiences a identificarlos.

| Programa correctamente codificado: ⬇

```
lado=int(input("Ingrese la medida del lado del cuadrado:"))
superficie=lado*lado
print("La superficie del cuadrado es")
print(superficie)
```

| Programa con un error sintáctico:

```
lado=int(input("Ingrese la medida del lado del cuadrado:"))
superficie=lado*lado
print("La superficie del cuadrado es")
print(Superficie)
```

¿Podés descubrir dónde está el error?

Es incorrecto la impresión de una variable nunca inicializada: "**Superficie**" (debemos respetar como la iniciamos en las líneas anteriores)

A continuación, te mostraremos como el programa visualiza e informa al usuario de este error al intentar correr el programa [\[14\]](#).

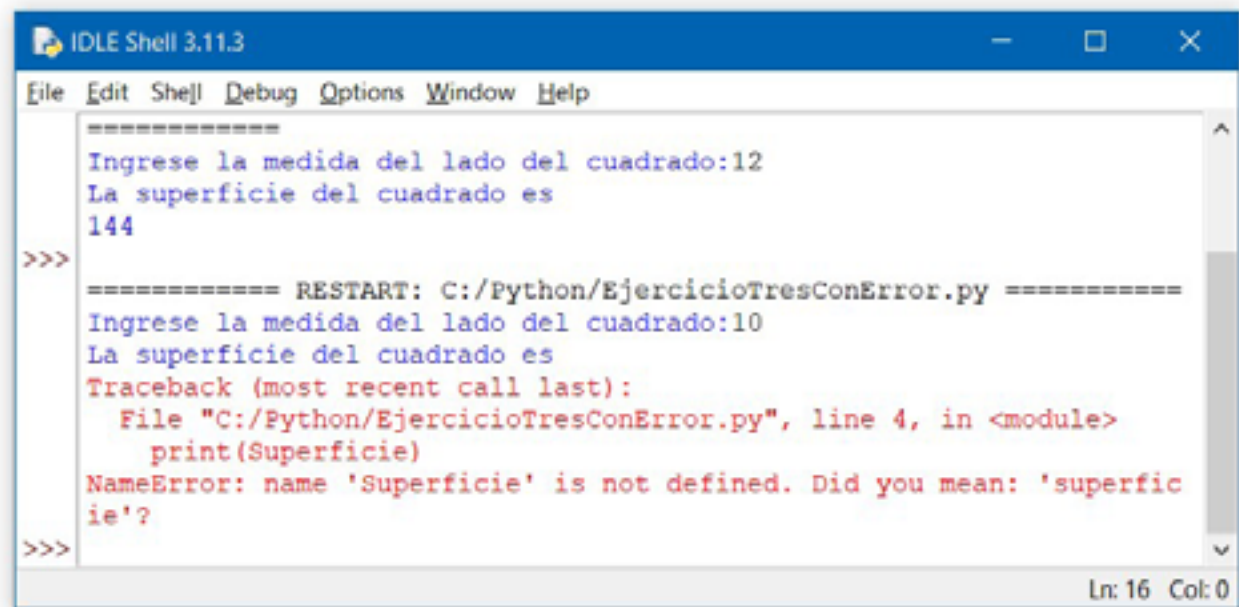
| Programa con un error lógico:

```
lado=int(input("Ingrese la medida del lado del cuadrado:"))
superficie=lado*lado*lado
print("La superficie del cuadrado es")
print(superficie)
```

| Como podemos observar si ejecutamos el programa no presenta ningún error sintáctico [\[15\]](#), pero luego de ingresar el valor del lado del cuadrado (por ejemplo el valor 12) obtenemos como resultado un **valor incorrecto** (imprime el 1728), esto debido que definimos incorrectamente la fórmula para calcular la superficie del cuadrado:

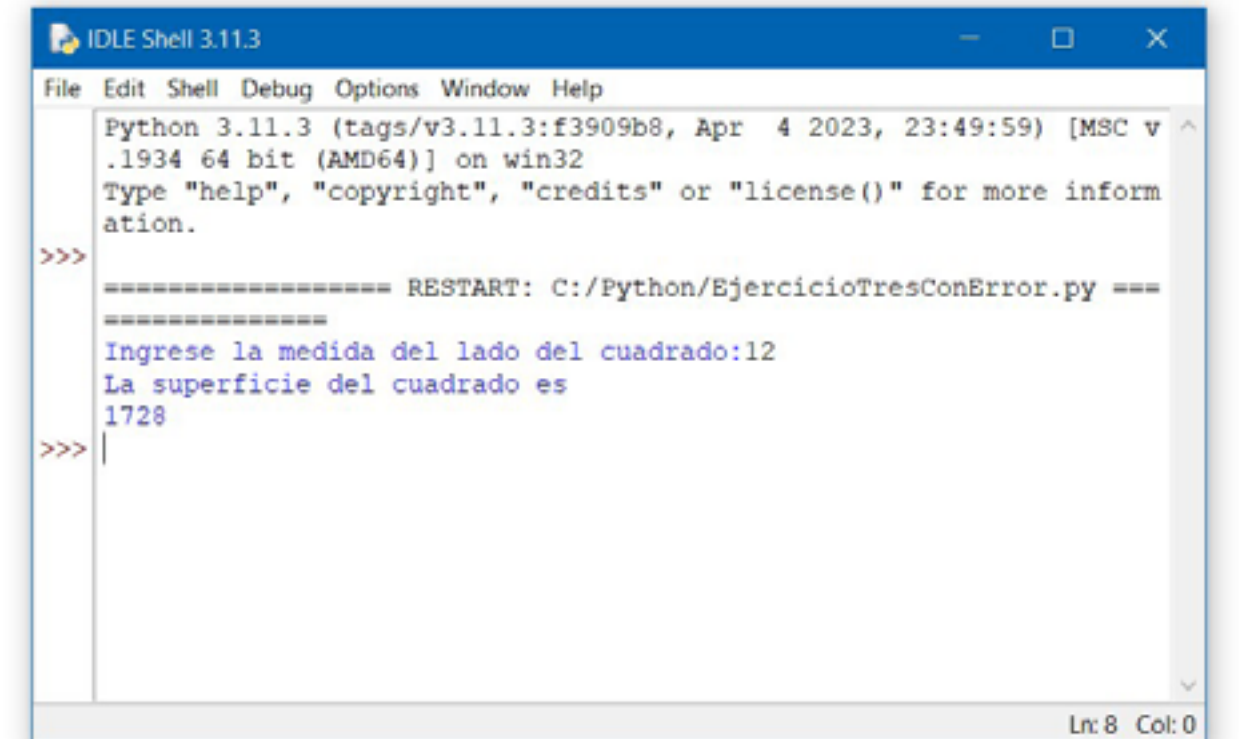
```
superficie=lado*lado*lado
```

[14]



```
IDLE Shell 3.11.3
File Edit Shell Debug Options Window Help
=====
Ingrese la medida del lado del cuadrado:12
La superficie del cuadrado es
144
>>>
===== RESTART: C:/Python/EjercicioTresConError.py =====
Ingrese la medida del lado del cuadrado:10
La superficie del cuadrado es
Traceback (most recent call last):
  File "C:/Python/EjercicioTresConError.py", line 4, in <module>
    print(Superficie)
NameError: name 'Superficie' is not defined. Did you mean: 'superficie'?
>>>
```

[15]



```
IDLE Shell 3.11.3
File Edit Shell Debug Options Window Help
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v
.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more inform
ation.
>>>
===== RESTART: C:/Python/EjercicioTresConError.py =====
Ingrese la medida del lado del cuadrado:12
La superficie del cuadrado es
1728
>>> |
```

Estructuras de programación secuencial

Cuando en un problema *solo participan operaciones, entradas y salidas* se la denomina una estructura secuencial. Los problemas diagramados y codificados previamente emplean solo estructuras secuenciales.

La programación requiere una práctica ininterrumpida de diagramación y codificación de problemas.

Ejemplo 4

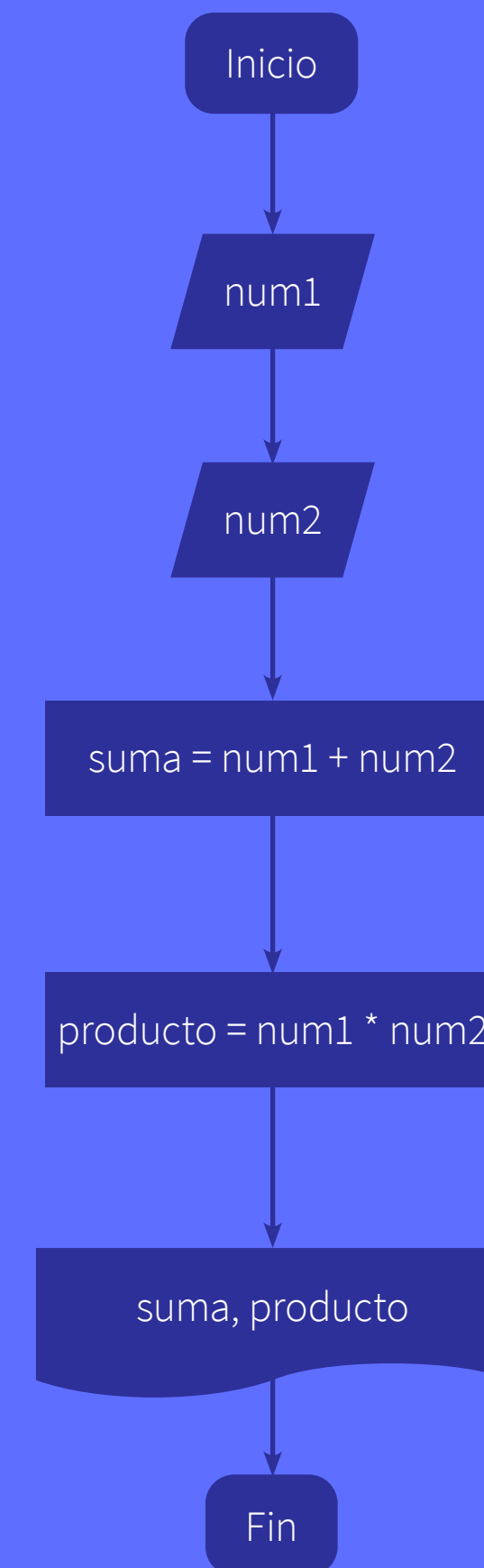
Doble entrada de numeros y operaciones

Realizar la carga de dos números enteros por teclado e imprimir su suma y su producto [16].

- | Tenemos dos entradas num1 y num2.
- | Dos operaciones: la suma y el producto de los valores ingresados.
- | Dos salidas, que son los resultados de la suma y el producto de los valores ingresados.

En el símbolo de impresión podemos indicar una o más salidas, eso queda a criterio del programador, lo mismo para indicar las entradas por teclado.

[16]



ejercicioCuatro.py

| Programa codificado: 

```
num1=int(input("ingrese primer valor:"))
num2=int(input("ingrese segundo valor:"))
suma=num1+num2
producto=num1*num2
print("La suma de los dos valores es")
print(suma)
print("El producto de los dos valores es")
print(producto)
```

Prueba a ejecutar el programa y verificar que los resultados de salida sean correctos.

Ejemplo 5

Calculador de importe

Realizar la carga del precio de un producto y la cantidad a llevar.

Mostrar cuánto se debe pagar (se ingresa un valor entero en el precio del producto) [|17|](#).

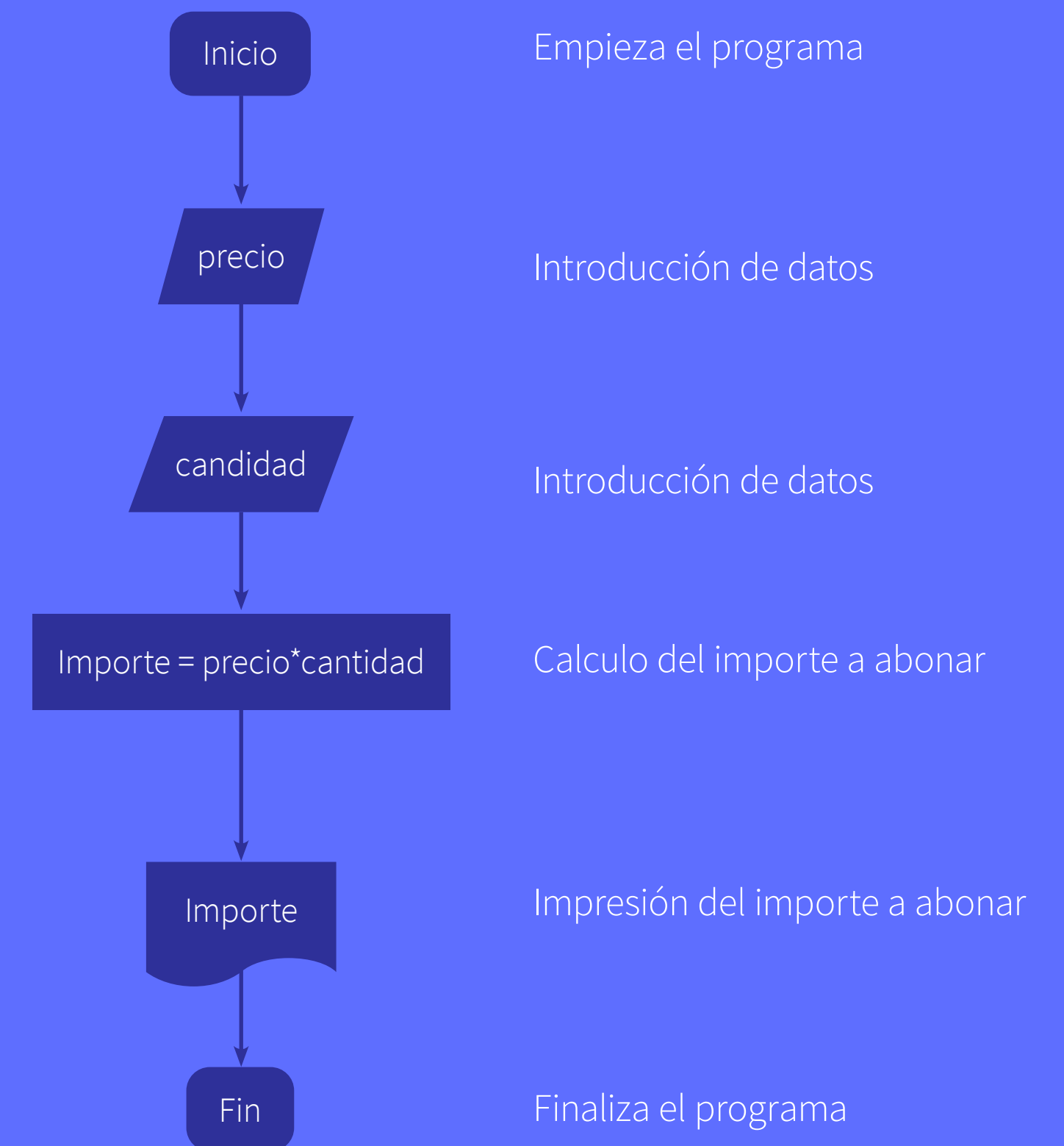
- | Tenemos dos entradas: Precio y cantidad
- | Una operación: Para calcular el importe
- | Una salida: En pantalla

Prueba a ejecutar el programa y verificar que los resultados de salida sean correctos.

| Programa codificado: 

```
precio=int(input("Ingrese el precio del producto:"))
cantidad=int(input("Ingrese la cantidad de productos a llevar:"))
importe=precio*cantidad
print("El importe a pagar es")
print(importe)
```

[|17|](#)





Desempeño 3

| Realizar la carga del lado de un cuadrado por teclado, mostrar por pantalla el perímetro del mismo (El perímetro de un cuadrado se calcula multiplicando el valor del lado por cuatro)



Desempeño 4

| Escribir un programa en el cual se ingresen cuatro números, calcular e informar la suma de los dos primeros y el producto del tercero y el cuarto.



Desempeño 5

| Realizar un programa que lea cuatro valores numéricos e informar su suma y promedio.



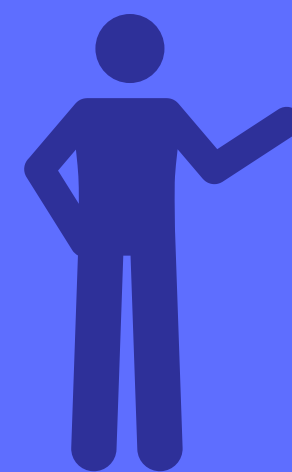
Desempeño 6

| Calcular el sueldo mensual de un operario conociendo la cantidad de horas trabajadas y el valor por hora.

Soluciones



Cuando te sientas listo, hacé click en el boton "Soluciones" para verificar tus respuestas



© 2023

Esta semana hemos aprendido muchísimo: desde cómo analizar un problema y traducir su solución a un diagrama de flujo, hasta codificar nuestros primeros programas en Python.

La clave para seguir correctamente la materia es que lleves a cabo todos los ejercicios en la computadora, tanto los que son ejemplos como los desempeños, y que no dudes en consultar ante cualquier inconveniente.

¡Nos vemos la próxima clase!

Créditos

Imágenes

Encabezado: Image by Steve Buissinne from Pixabay

<https://pixabay.com/photos/pawn-chess-pieces-strategy-chess-2430046/>

Tipografía

Para este diseño se utilizó la tipografía *Source Sans Pro* diseñada por Paul D. Hunt.

Extraída de Google Fonts.

Si detectás un error del tipo que fuere (falta un punto, un acento, una palabra mal escrita, un error en código, etc.), por favor comunicate con nosotros a correcciones@issd.edu.ar e indicanos por cada error que detectes la página y el párrafo. Muchas gracias por tu aporte.

Soluciones a los problemas


Te recomendamos utilizar esta sección luego de haber intentado por un largo tiempo la resolución y también para verificar tus soluciones.

Solución al desempeño 3

```
lado=int(input("Ingrese el lado del cuadrado:"))
perimetro=lado*4
print("El perimetro del cuadrado es")
print(perimetro)
```

Solución al desempeño 4

```
num1=int(input("Ingrese primer valor:"))
num2=int(input("Ingrese segundo valor:"))
num3=int(input("Ingrese tercer valor:"))
num4=int(input("Ingrese cuarto valor:"))
suma=num1+num2
producto=num3*num4
print("La suma de los dos primero valores es")
print(suma)
```




```
print("El producto del tercer y cuarto valor es")
print(producto)
```

Solución al desempeño 5

```
num1=int(input("Ingrese primer valor:"))
num2=int(input("Ingrese segundo valor:"))
num3=int(input("Ingrese tercer valor:"))
num4=int(input("Ingrese cuarto valor:"))
suma=num1+num2+num3+num4
promedio=suma/4
print("La suma de los cuatro valores es")
print(suma)
print("El promedio es")
print(promedio)
```

Solución al desempeño 6

```
horastrabajadas=int(input("Ingrese la cantidad de horas trabajadas:"))
valorhora=int(input("Importe a pagar por hora:"))
sueldo=horastrabajadas*valorhora
print("Sueldo a pagar")
print(sueldo)
```

Volver a la clase



Quando te sientas listo,
hacé click en el boton
"Volver a la clase" para
continuar con el módulo.

