

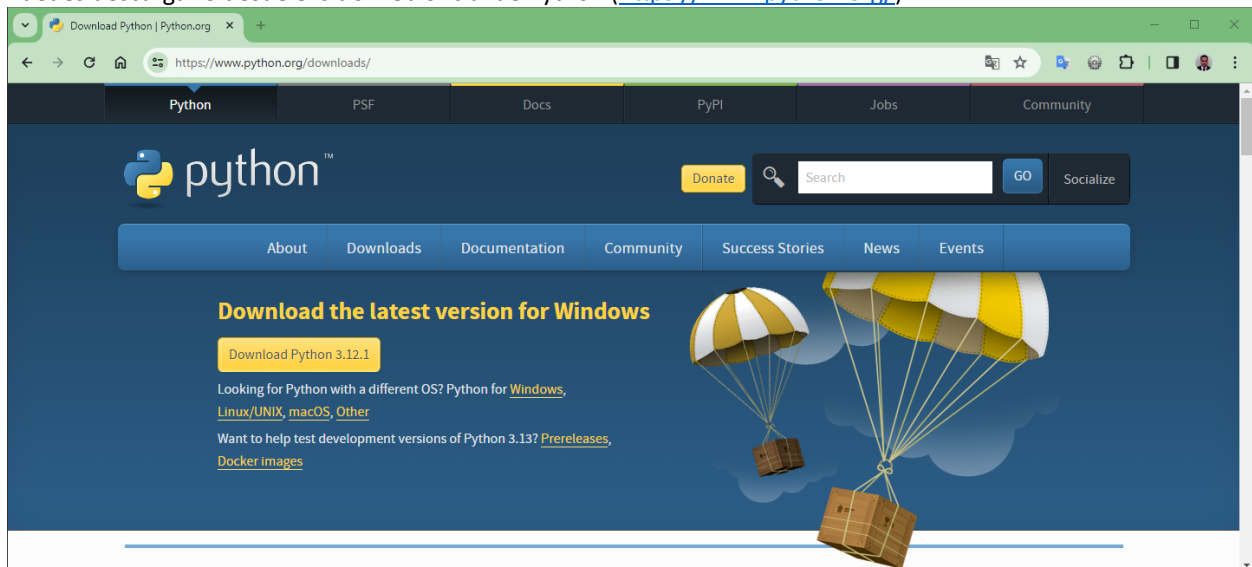
## Guía Práctica N.º 4: Preparación de Entorno y Primeros Programas en Python

### Resultado De Aprendizaje:

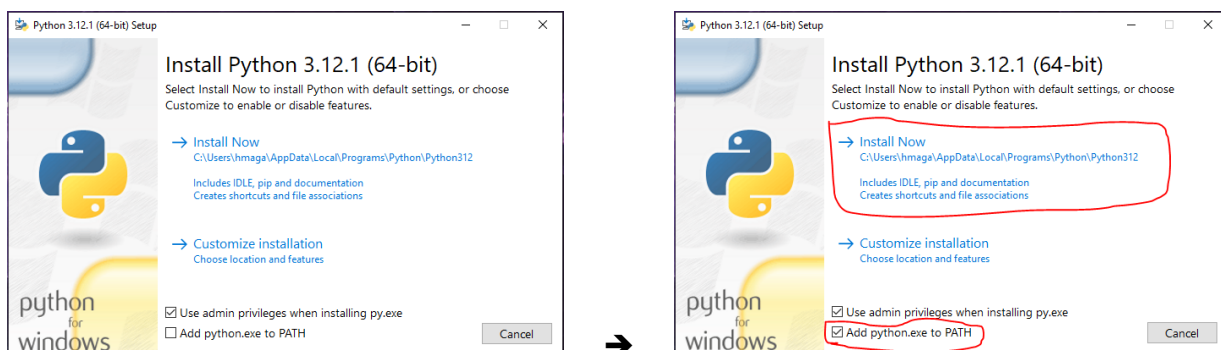
- Conocer los entornos de desarrollo para Python en consola.
- Conocer los lineamientos básicos para crear programas de consola en Python.
- Identificar y definir las variables en una aplicación de consola

### Instalación de Python

Puedes descargarlo desde el sitio web oficial de Python (<https://www.python.org/>)

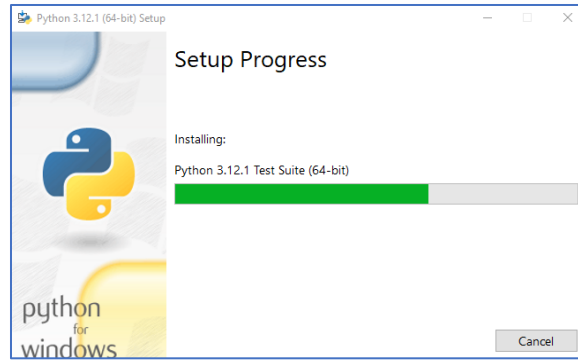


Una vez descargado proceder a ejecutar el instalador y saldrá la siguiente ventana:

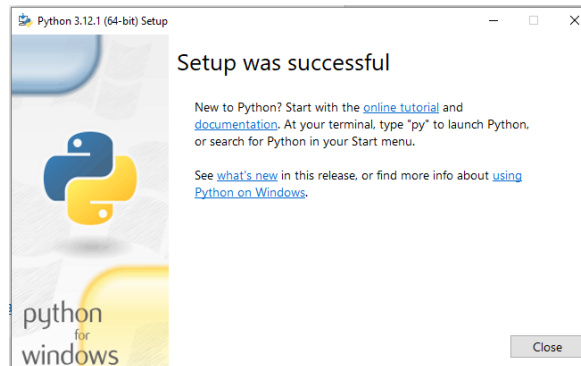


Debe seleccionar el segundo check de abajo (**Add python.exe to PATH**) y luego clic en la opción de **Install Now**

Iniciara el proceso de instalacion:

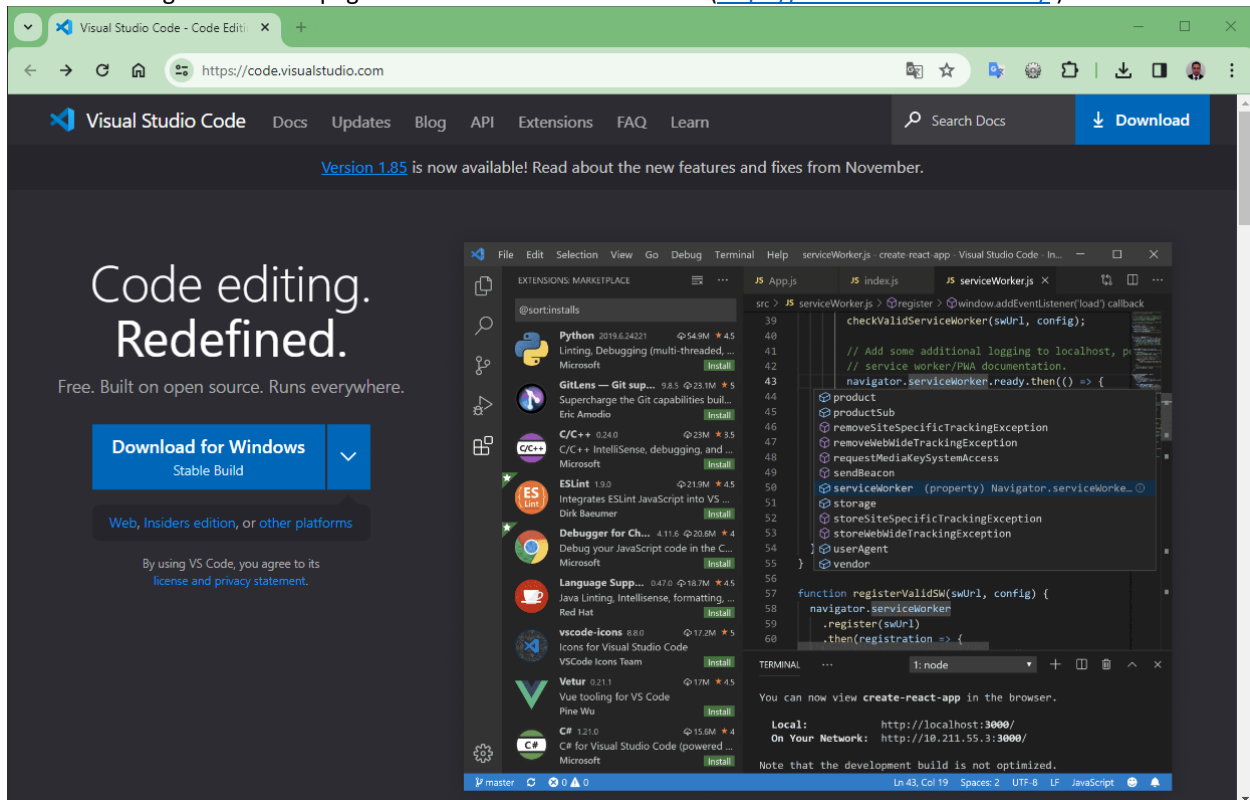


Al finalizar mostrará:



## Instalación de Visual Studio Code

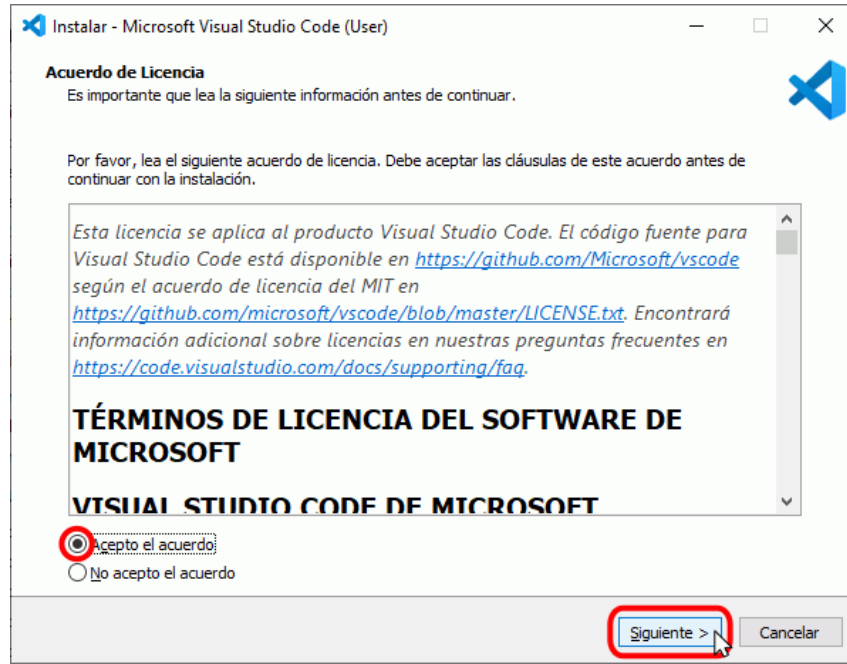
Puedes descargarlo desde la página oficial de Visual Studio Code (<https://code.visualstudio.com/>)



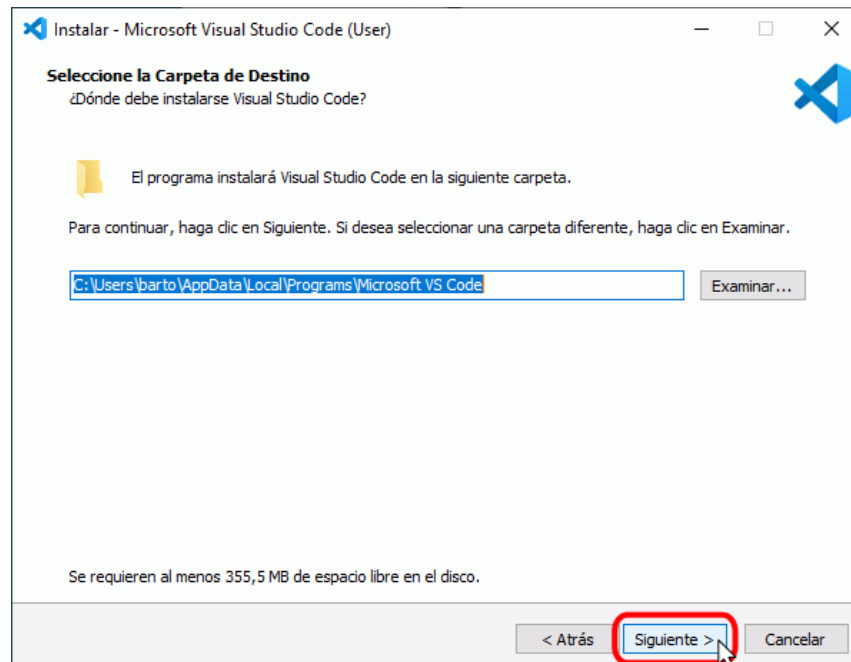
## Instalar Visual Studio Code en Windows

**Nota:** Las capturas siguientes corresponden a Visual Studio Code 1.82 (*User installer*) en Windows 10 de 64 bits. Versiones posteriores pueden ser ligeramente diferentes.

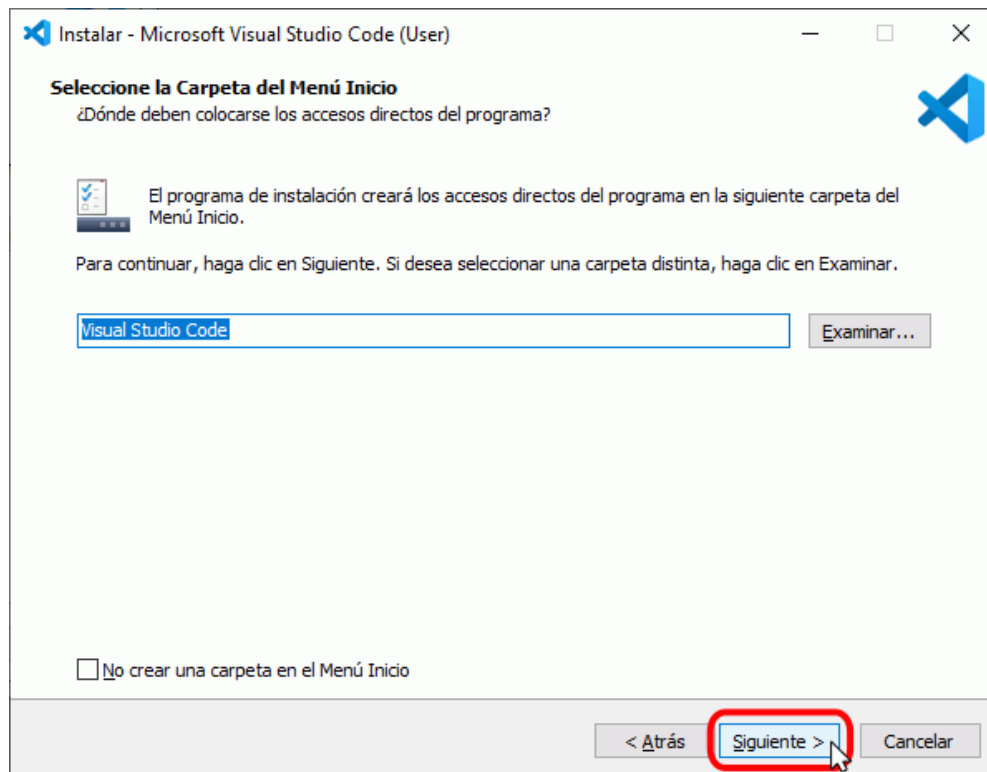
1. Haga doble clic sobre el instalador de Visual Studio Code para poner en marcha el asistente de instalación.
2. La primera pantalla exige aceptar la licencia de Visual Studio Code para continuar la instalación:



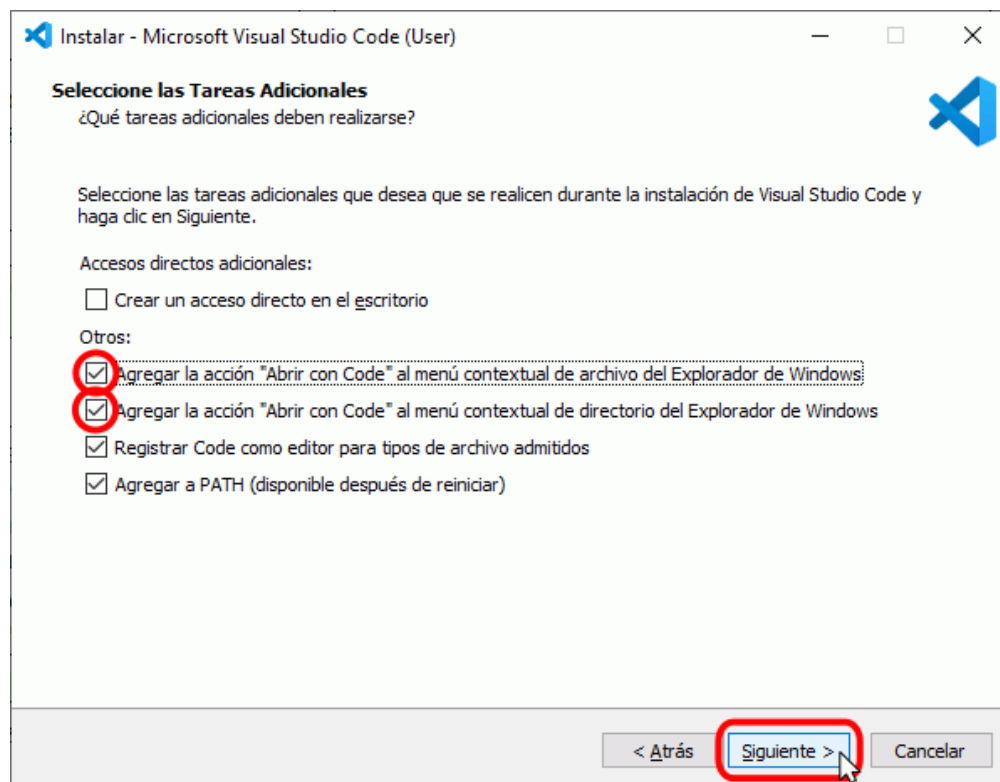
3. La segunda pantalla permite elegir el directorio de instalación (por tratarse de la versión *User installer*, el directorio de instalación está en la carpeta de usuario, no en Archivos de programa):



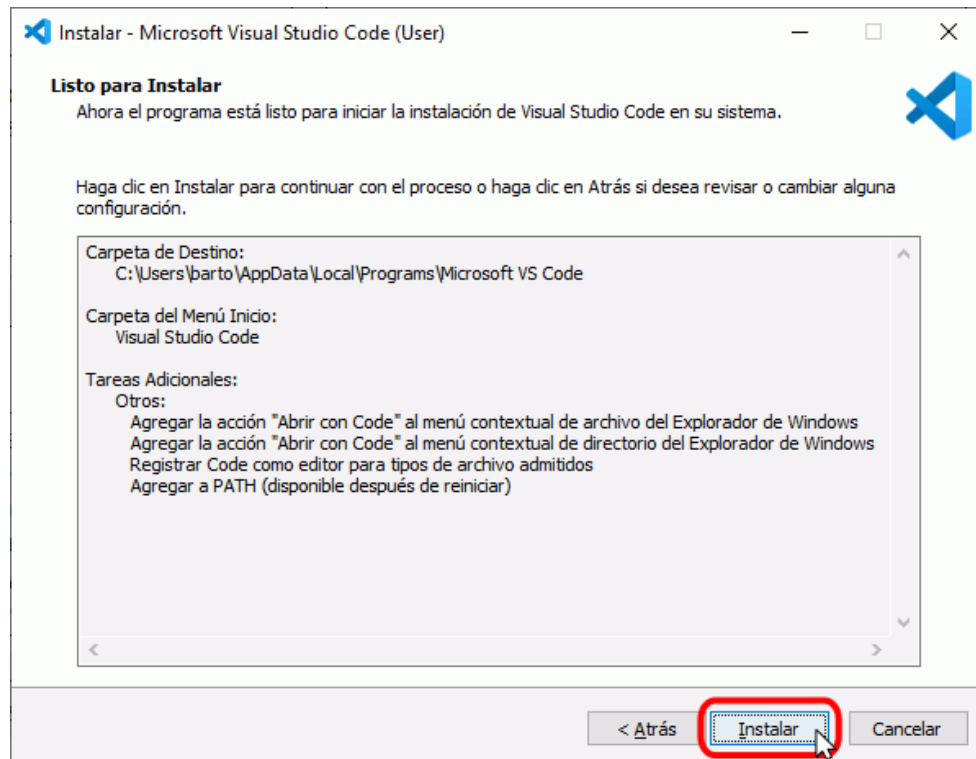
4. La tercera pantalla permite elegir el nombre de la carpeta del menú de inicio:



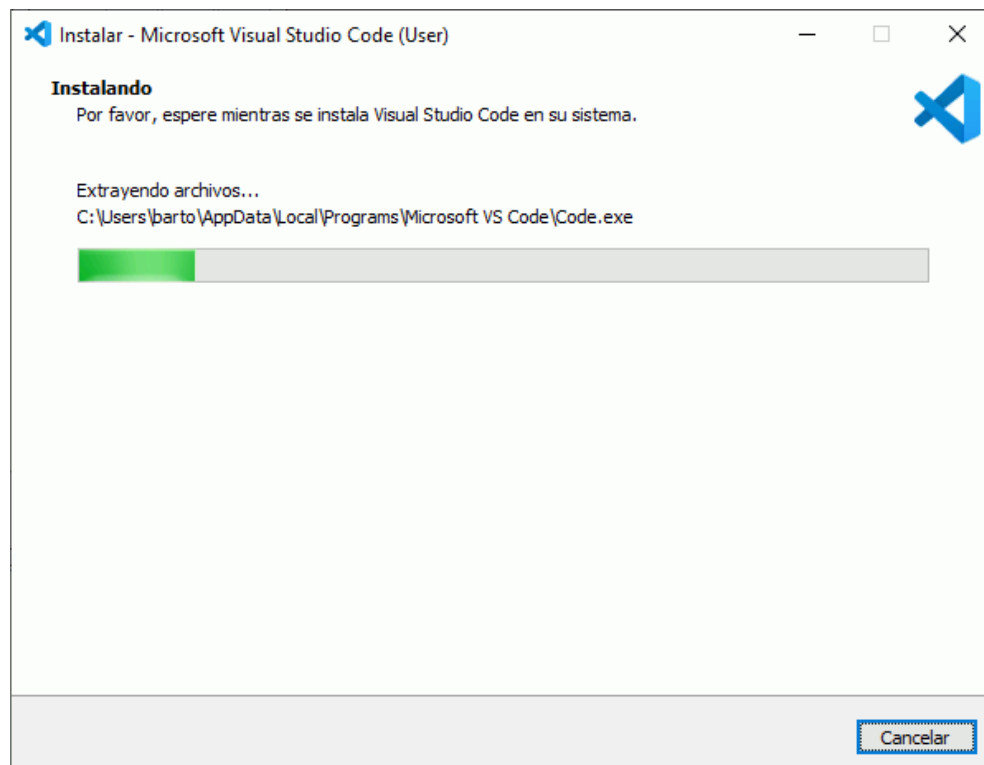
5. La cuarta pantalla permite elegir algunas tareas adicionales tras la instalación. Personalmente, aconsejo marcar las casillas "Agregar la acción ...":



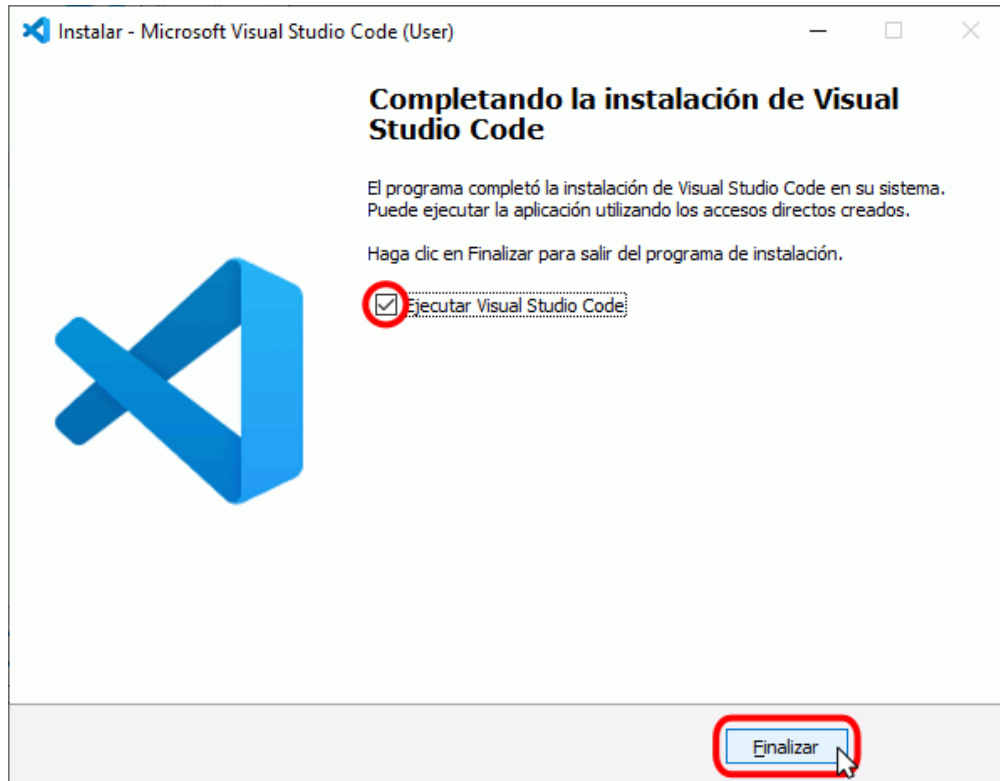
6. Finalmente se muestran las opciones elegidas en las pantallas anteriores. Para iniciar la instalación, haga clic en Instalar.



7. A continuación, se instalará Visual Studio Code.

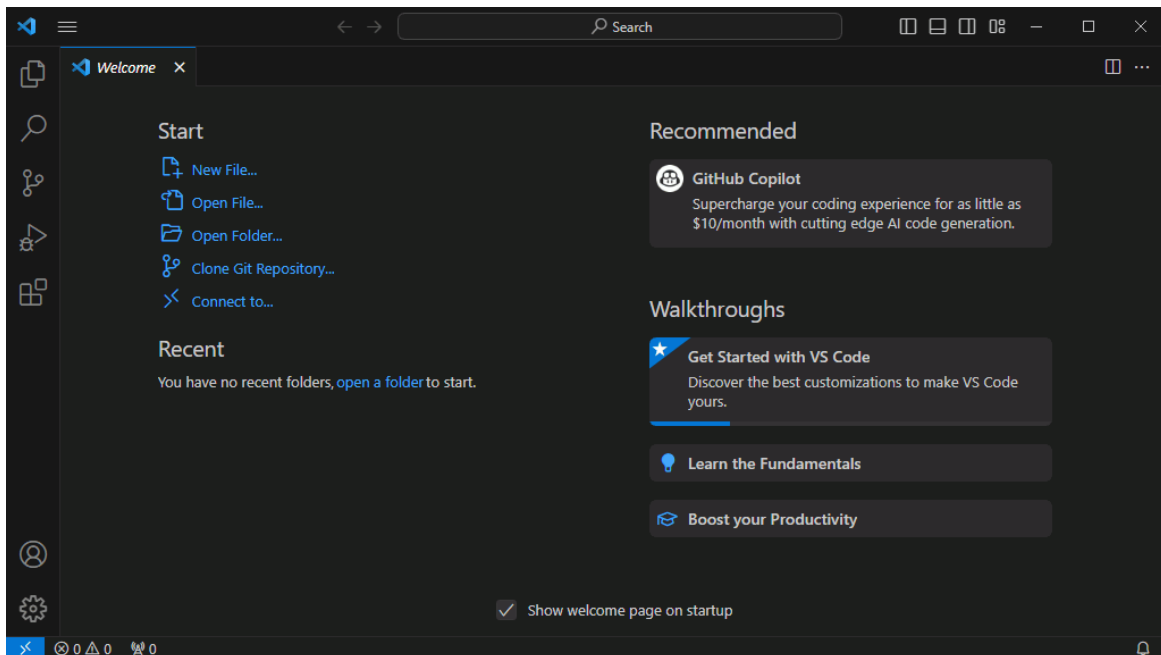


8. Una vez completada la instalación, se muestra la pantalla final. Si va a utilizar [Git](#) con Visual Studio Code, desmarque la casilla "Ejecutar Visual Studio Code", haga clic en Finalizar e [instale Git](#).



## Primera ejecución

La primera vez que se abre Visual Studio Code tras la instalación, se muestra una página de bienvenida al programa:

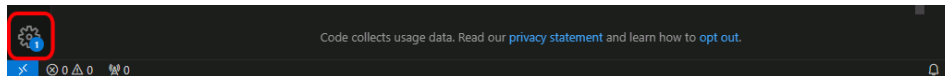


## Actualizar Visual Studio Code

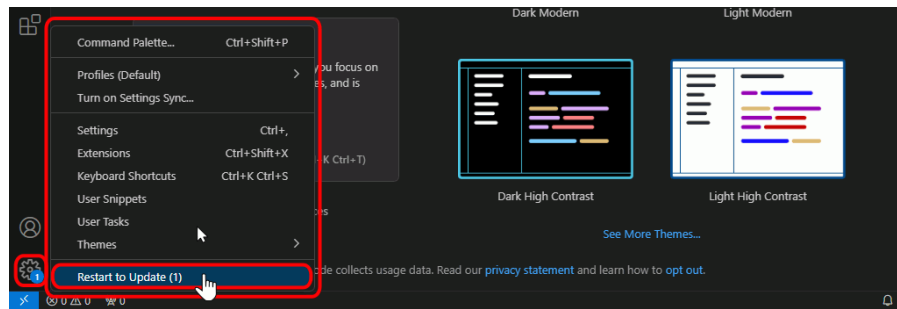
Visual Studio Code se actualiza automáticamente cada vez que se publica una nueva versión. Visual Studio Code muestra un aviso cuando la actualización se ha descargado y podemos aplicar la actualización inmediatamente, como se muestra en las capturas de pantalla siguientes. En cualquier caso, hayamos pedido o no aplicar la actualización, si cerramos Visual Studio Code, la actualización se aplicará al volver a abrirlo.

**Nota:** Las capturas siguientes corresponden a la actualización de Visual Studio Code 1.82 a Visual Studio Code 1.85 "Restart to Update" / en Windows 10 de 64 bits. Versiones posteriores pueden ser ligeramente diferentes.

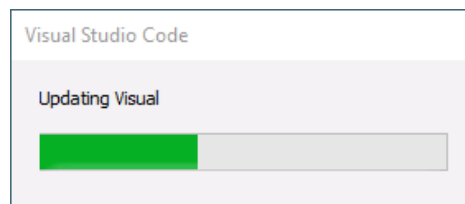
1. Cuando hay una actualización disponible, Visual Studio Code muestra un número en el icono de la rueda dentada situado en la parte inferior izquierda:



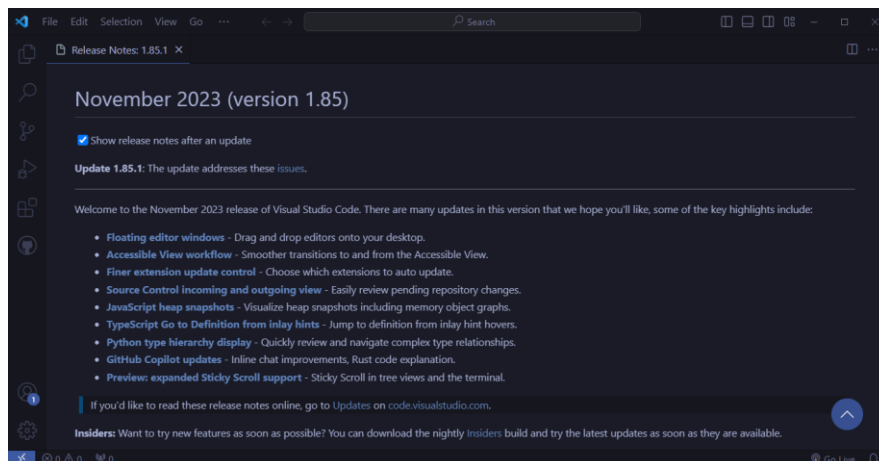
2. Haga clic en el icono de la rueda dentada y elija la opción "Restart to Update" / "Reiniciar para actualizar":



3. Visual Studio Code instalará la actualización:



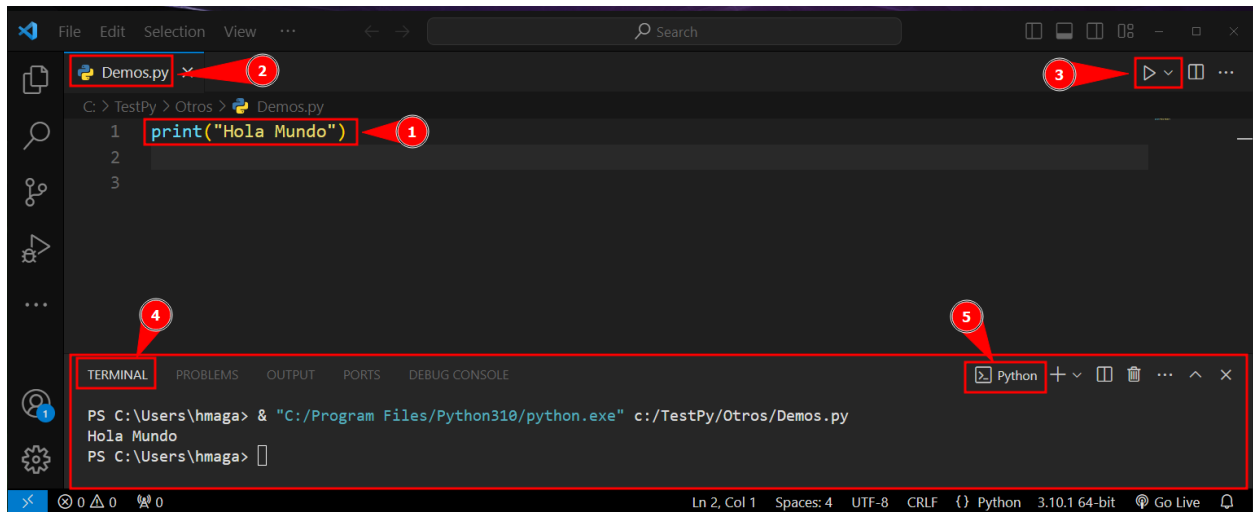
4. Al reiniciarse, se mostrarán en una nueva pestaña las notas de la versión instalada y el icono de la rueda dentada ya no mostrará el número:



## Primeros Programas en Python

### Ejemplo1: Hola Mundo

- 1- Escribir la instrucción: `print("Hola Mundo")` en el editor de Visual Studio Code,
- 2- Guardar el archivo con extensión `.py`
- 3- Ejecutar el programa
- 4- Mostrará el resultado en la consola (Terminal).
- 5- Observen que se está usando una consola propia de Python.



Para mostrar un mensaje en pantalla también se pueden usar comillas simples.

```
1 print("Hola Mundo")
2 #Se puede usar comillas dobles o simples
3 print('Hola Mundo')
4 #Forma correcta de combinar y que muestre las comillas dobles en la salida
5 print('Hola "Mundo"')
```

La salida será:

```
Hola Mundo
Hola Mundo
Hola "Mundo"
```

Pero se debe tener cuidado al combinar

```
#Forma incorrecta
print("Hola "Mundo")
```

Al ejecutar mostrará el error:

```
File "c:\TestPy\Otros\Demos.py", line 3
    print("Hola "Mundo")
            ^^^^^^^^^^^^^
SyntaxError: invalid syntax. Perhaps you forgot a comma?
```



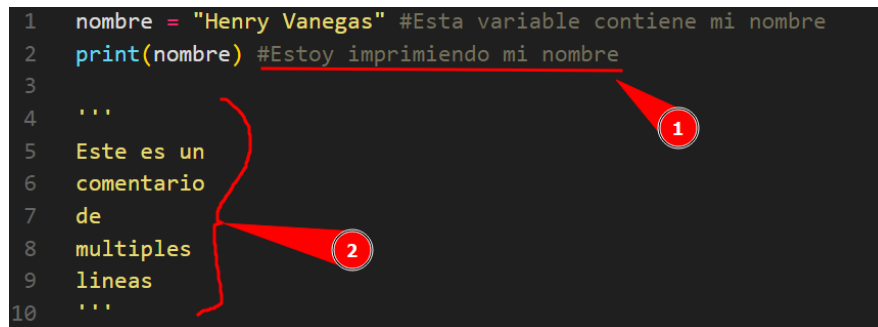
## Comentarios en Python

En Python, los comentarios comienzan con el símbolo #. (1 en la imagen).

En los códigos anteriores, se ha usado un comentario para explicar lo que hace el código. Cuando se ejecuta el código, el intérprete ignorará el comentario y ejecutará la función print().

También puedes usar cadenas de texto con triple comilla para hacer **comentarios de varias líneas** (2 en la imagen). Aunque técnicamente no son comentarios, ya que Python los interpreta como cadenas de texto, se pueden usar como tales si no se asignan a ninguna variable. Ejemplo:

```
1 nombre = "Henry Vanegas" #Esta variable contiene mi nombre
2 print(nombre) #Estoy imprimiendo mi nombre
3
4 '''
5 Este es un
6 comentario
7 de
8 multiples
9 lineas
10 '''
```



## Declaración de variables

En Python como todo lenguaje de programación, una variable es un contenedor que almacena un valor. En otras palabras, una variable es el nombre que se le da a un valor, para que sea fácil referirse a él más adelante. A diferencia de otros lenguajes de programación como C# o Java, no es necesario definir explícitamente una variable en Python antes de usarla.

Las variables en Python se declaran asignándoles un valor.

Para asignar un valor a una variable utilizando el operador de asignación =.

**nombre = valor**

Por ejemplo:

```
1 nombre = "Henry Vanegas"
2 numero = 89
3 flotante = 3.14
```

En esas tres instrucciones se está declarando tres variables respectivamente cada una asignándole un valor diferente.

Python permite la asignación en cadena, lo que hace posible asignar el mismo valor a varias variables simultáneamente.

```
a = b = c = 300
```

Por ejemplo:

Python permite las Múltiples Asignaciones.

```
1 nombre , apellido = 'Henry' , 'Vanegas' #Múltiples Asignaciones
2 print(nombre)
3 print(apellido)
```

Por ejemplo:

## Reglas para nombrar(colocarle el identificador) a las variables

En Python existen una serie de reglas para los nombres de variables:

1. Sólo pueden **contener los siguientes caracteres**:
  - Letras minúsculas.
  - Letras mayúsculas.
  - Dígitos.
  - Guiones bajos (\_).
2. **No** debe haber **espacios en blanco** entre el nombre del identificador (se puede usar las **nomenclaturas** propuestas por los programadores como buenas prácticas de programación).
3. Deben empezar con una letra o un guion bajo, **nunca con un dígito**.

```
3erValor = 7 #Este nombre de variable no es valido
```

4. No pueden llevar **caracteres especiales**: { @, #, ñ, Ñ, á, é, ... } (<https://www.uv.es/jac/guia/coditab.htm>)
5. No pueden ser una **palabra reservada** del lenguaje («keywords»).

Podemos obtener un listado de las palabras reservadas del lenguaje de la siguiente forma:

```
help('keywords')
```

y proporciona el siguiente resultado:

```
TERMINAL  PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE

Here is a list of the Python keywords.  Enter any keyword to get more help.

False      class      from       or
None       continue  global     pass
True       def       if         raise
and        del       import     return
as         elif      in         try
assert     else      is         while
async      except    lambda     with
await      finally  nonlocal   yield
break      for       not
```

**Nota:** Por lo general se prefiere dar nombres **en inglés** a las variables que utilicemos, ya que así hacemos nuestro código más «**internacional**» y con la posibilidad de que otras personas puedan leerlo, entenderlo y – llegado el caso – modificarlo. **Es sólo una recomendación, nada impide que se haga en castellano.**

**Importante:** Python es «**case-sensitive**» (Diferencia mayúsculas de minúsculas)

```
numero = 7
```

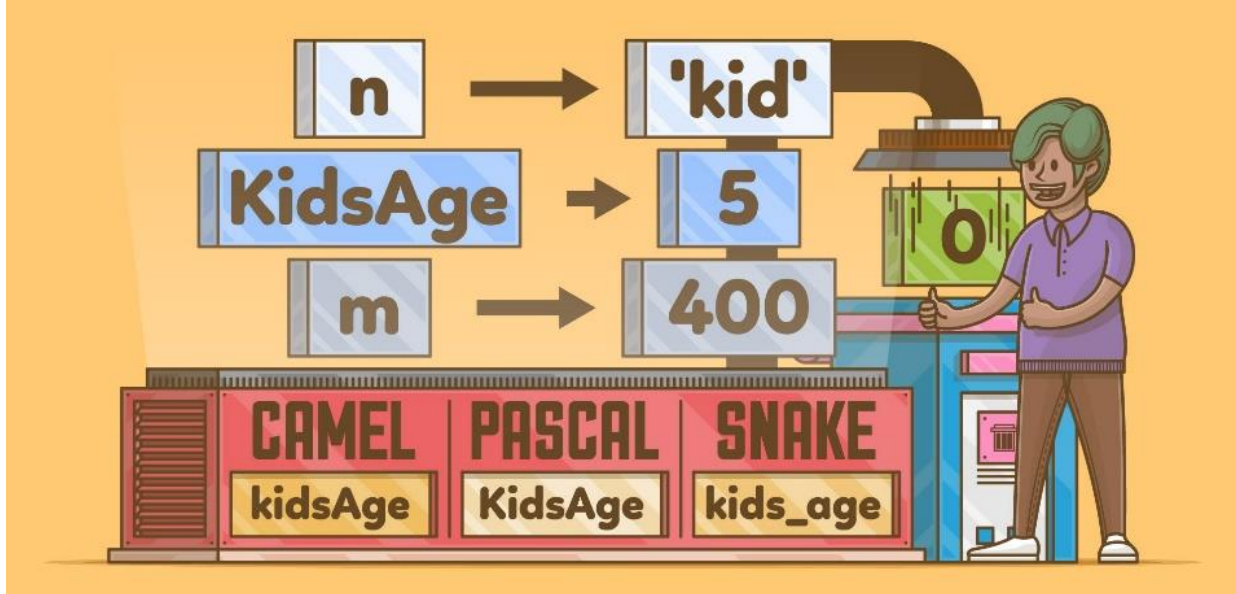
```
Numero = 9
```

```
#Las anteriores son variables diferentes
```

Por ejemplo:

## Nomenclaturas Convenciones para nombres

Mientras se sigan las reglas para nombrar variables no hay problema en la forma en la que se escriban, pero sí existe una convención para la nomenclatura de las variables. Se utiliza el llamado **snake\_case** en el que utilizamos caracteres en minúsculas (incluyendo dígitos si procede) junto con guiones bajos – cuando sean necesarios para su legibilidad. Y así también existe la nomenclatura **PascalCase** y el **camelCase**.



## Constantes

Un caso especial y que vale la pena destacar son las constantes. Podríamos decir que es un tipo de variable pero que su valor no cambia a lo largo de nuestro programa. Por ejemplo, la velocidad de la luz. Sabemos que su valor es constante de 300.000 km/s.

En el caso de las constantes utilizamos mayúsculas (incluyendo guiones bajos si es necesario) para nombrarlas. Para la velocidad de la luz nuestra constante se podría llamar: **LIGHT\_SPEED**.

En Python no se reconocen las constantes como tal, pero una buena práctica de programación de la nomenclatura nos indica que se debe escribir todo el nombre de la variable en mayúscula para hacer referencia que se trata de una constante.

```
1  PI = 3.14
2  SOUND_SPEED = 343.2
3  WATER_DENSITY = 997
```

Algunos ejemplos de asignaciones a constantes:

## Entrada de datos del usuario

Para pedir datos al usuario se utiliza la función **input()** pero este dato debe ser asignado a una variable. Si se quiere que ese dato sea de un tipo en específico se debe convertir antes de asignar a la variable.

```
1  nombre = input("¿Cómo te llamas?")
2  edad = int(input("Ingresa tu edad:"))
```

Ejemplo:

Observar que la edad fue convertida a entera antes de ser asignada a la variable edad.

## Salida de datos para el usuario

Para mostrar datos al usuario se utiliza la función `print()` pero se puede realizar de diferentes maneras:

### 1. *Mostrar directamente el dato de la variable*

Ejemplo:

```
3 # Mostrando datos directamente desde las variables
4 print(nombre)
5 print(edad)
```

### 2. *Concatenando cadenas con variables*

Ejemplo:

```
6 # Mostrando datos usando concatenacion de cadenas con variables
7 print("¡Hola, " + nombre + "!")
8 print("Tienes " + str(edad) + " años")
```

Observe que la variable `edad` primero se tiene que convertir a cadena `String` antes de concatenar con las otras cadenas.

### 3. *Mostrar usando `.format`*

Ejemplo:

```
9 # Mostrando datos usando .format
10 print("Hola {} tienes {} años".format(nombre, edad))
```

Observe como se utilizan las llaves para reservar el espacio donde deben aparecer los valores de las variables especificadas en el `format` respectivamente.

### 4. *Mostrar usando `f-string`*

Ejemplo:

```
11 # Mostrando datos usando f-string
12 print(f"Hola {nombre} tienes {edad} años")
```

Observe que se utiliza una letra `f` antes de la cadena de texto de salida y como las variables van entre llaves directamente en el lugar donde se quiere que aparezcan.

Otra forma de aplicar formatos a otros tipos de datos sería:

```
13 # Mostrando datos con formato espedifico
14 resultado = 10 / 3
15 print("El resultado es {r}".format(r=resultado))
16 #Muestra: El resultado es 3.3333333333335
17 print("El resultado es {r:1.3f}".format(r=resultado))
18 #Muestra: El resultado es 3.333
```

### GUIA DE EJERCICIOS

1. **Calculadora de volumen de una esfera.** La fórmula es:  $V = (4/3) * (3.1416) * r^3$ .
2. **Calculadora de área de un círculo:** Escribe un programa que pida al usuario el radio de un círculo y calcule su área. Recuerda que la fórmula para calcular el área de un círculo es  $\pi r^2$ .
3. **Conversor de grados Celsius a Fahrenheit:** Escribe un programa que pida al usuario una temperatura en grados Celsius y la convierta a grados Fahrenheit.  
La fórmula para convertir grados Celsius a Fahrenheit es:  $F = C * 9/5 + 32$ .
4. **Calculadora de volumen de un cubo:** Escribe un programa que pida al usuario la longitud de un lado de un cubo y calcule su volumen. Recuerda que el volumen de un cubo es  $lado^3$ .
5. **Calculadora de la hipotenusa de un triángulo rectángulo:** Escribe un programa que pida al usuario las longitudes de los dos catetos de un triángulo rectángulo y calcule la longitud de la hipotenusa. Recuerda que la fórmula para calcular la hipotenusa es:  $\sqrt{cateto1^2 + cateto2^2}$ .
6. **Calculadora de velocidad media:** Escribe un programa que pida al usuario la distancia que ha recorrido y el tiempo que ha tardado, y calcule la velocidad media utilizando la fórmula  $velocidad = distancia / tiempo$ .
7. **Calculadora de tiempo de cocción:** Escribe un programa que pida al usuario el peso de un pollo y calcule el tiempo de cocción. Suponer que el tiempo de cocción es de 45 minutos para el primer kilogramo y de 30 minutos adicionales por cada kilogramo extra.
8. **Calculadora de consumo de agua:** Escribe un programa que pida al usuario la duración de su ducha en minutos y calcule el consumo de agua. Suponer que el consumo de agua es de 9.5 litros por minuto.
9. **Calculadora de calorías quemadas al caminar:** Escribe un programa que pida al usuario su peso y la distancia que ha caminado, y calcule las calorías quemadas. Suponer que se queman 0.53 calorías por kilogramo de peso y por kilómetro recorrido.
10. **Calculadora de tiempo de lectura:** Escribe un programa que pida al usuario el número de páginas de un libro y su velocidad de lectura, y calcule el tiempo que tardará en leer el libro. Suponer que la velocidad de lectura se da en páginas por hora.