



Práctica 2b:

Python y Visual Studio Code



¿Qué es Python?

- Lenguaje de programación de alto nivel, fácil de leer y escribir.
- Muy usado en: desarrollo web, ciencia de datos, automatización, inteligencia artificial, etc.



¿Qué es Visual Studio Code?

- Editor de código fuente gratuito de Microsoft.
- Soporta múltiples lenguajes (Python, JavaScript, C++, etc.)
- Integración con Git, terminal, depuración, y más extensiones.



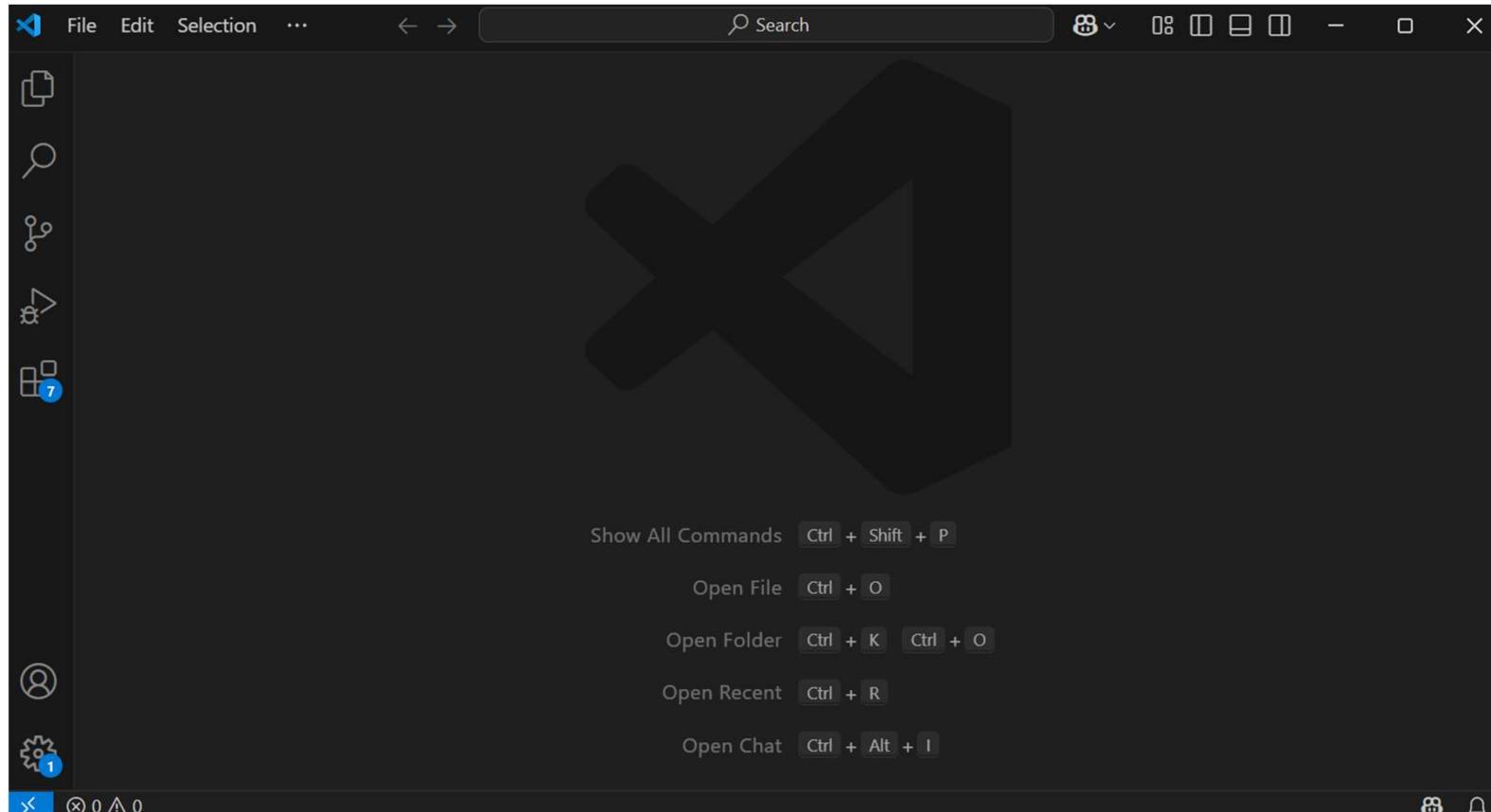
Instalación

- Python
 - <https://python.org>
- Visual Studio Code
 - <https://code.visualstudio.com/>



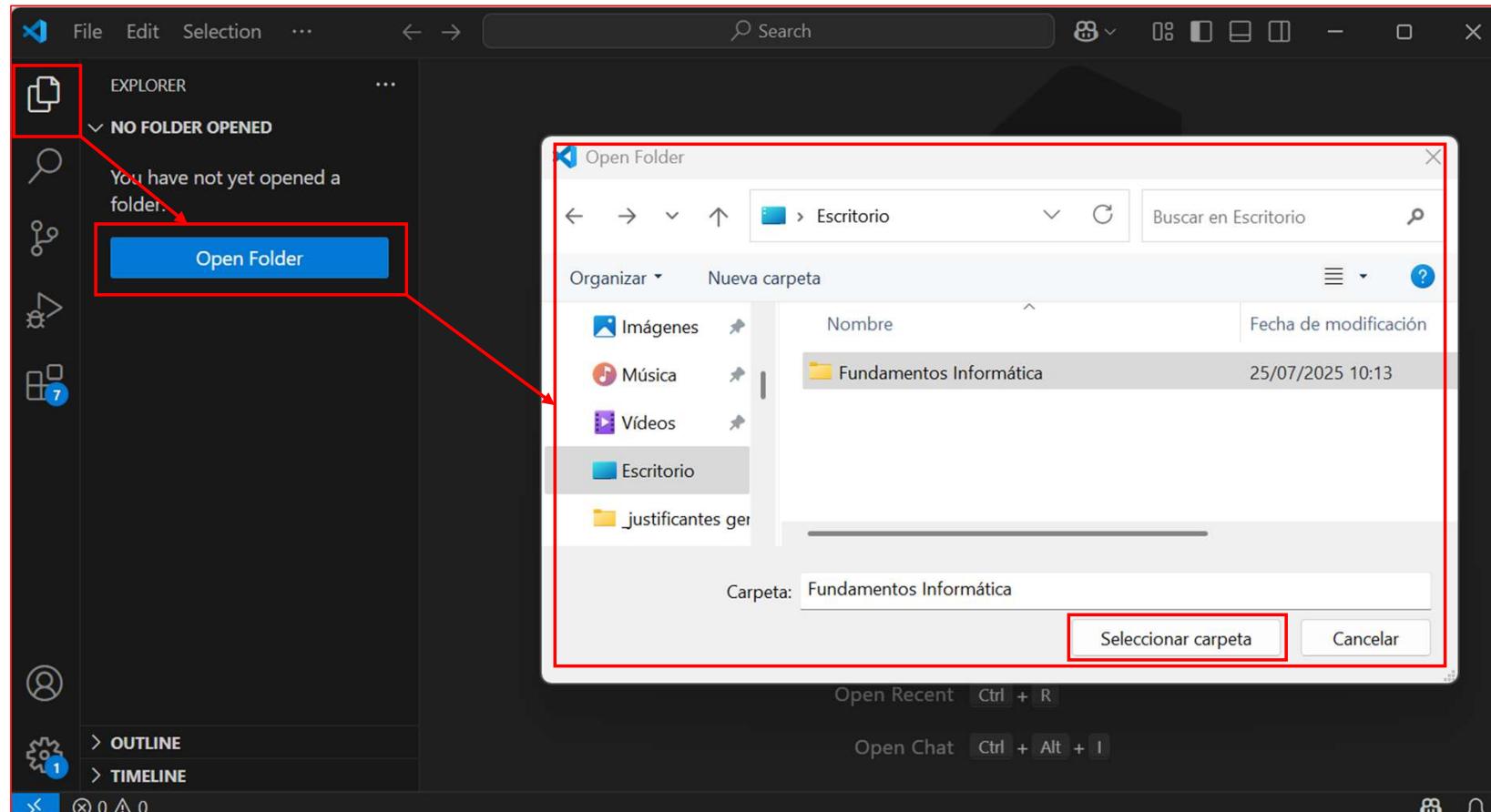


Introducción a Visual Studio Code



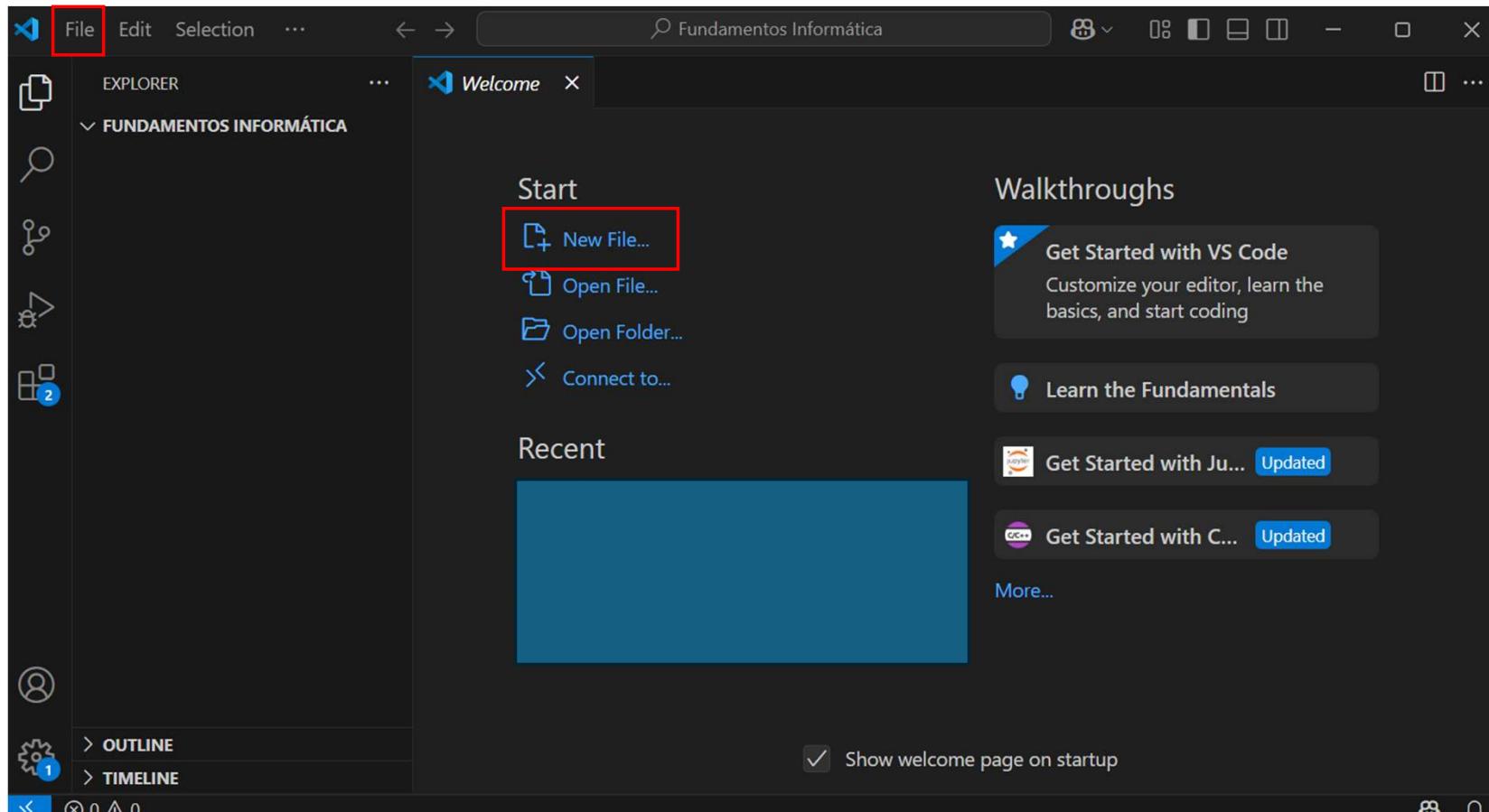


Seleccionar un directorio de trabajo



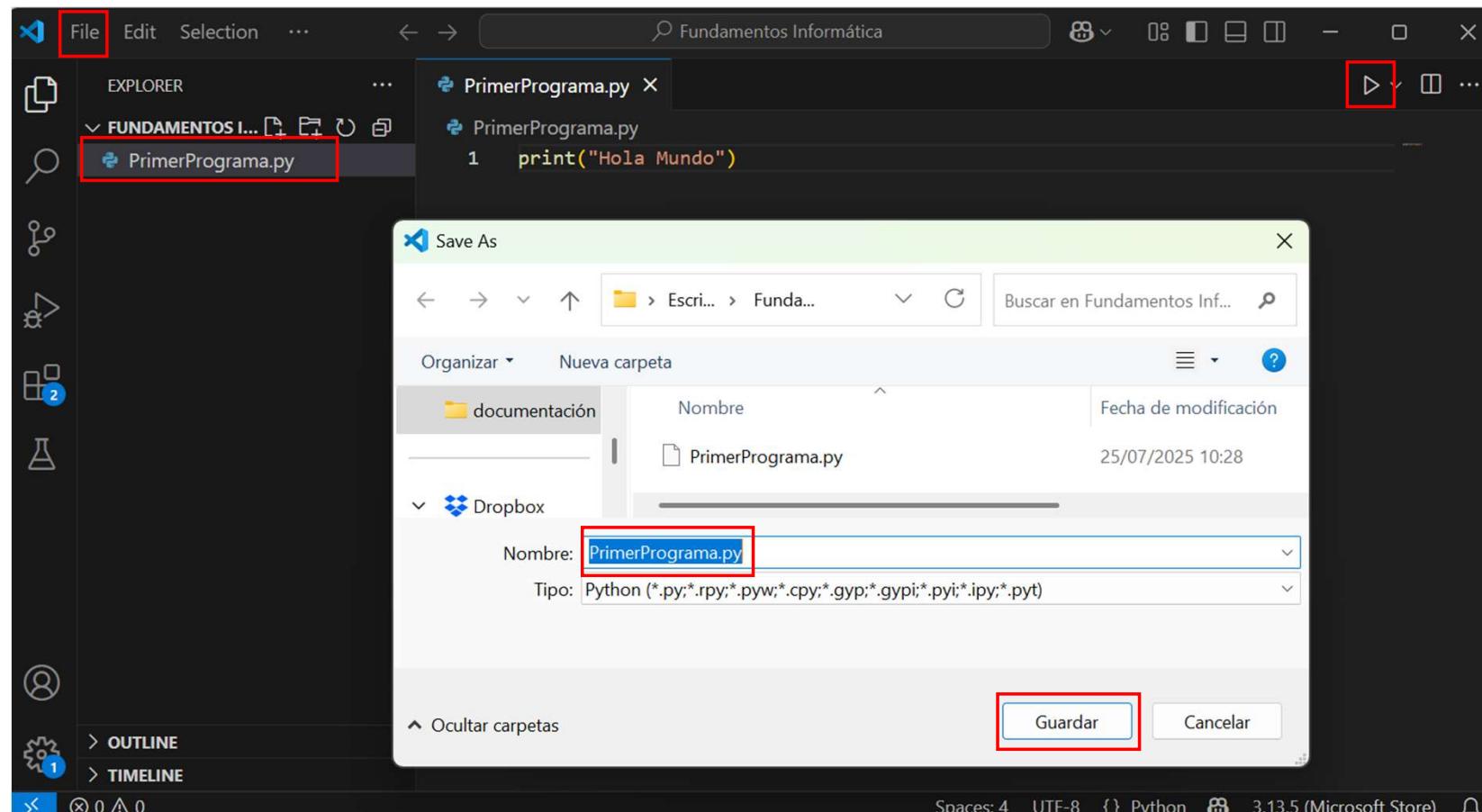


Crear un primer programa “Hola Mundo”





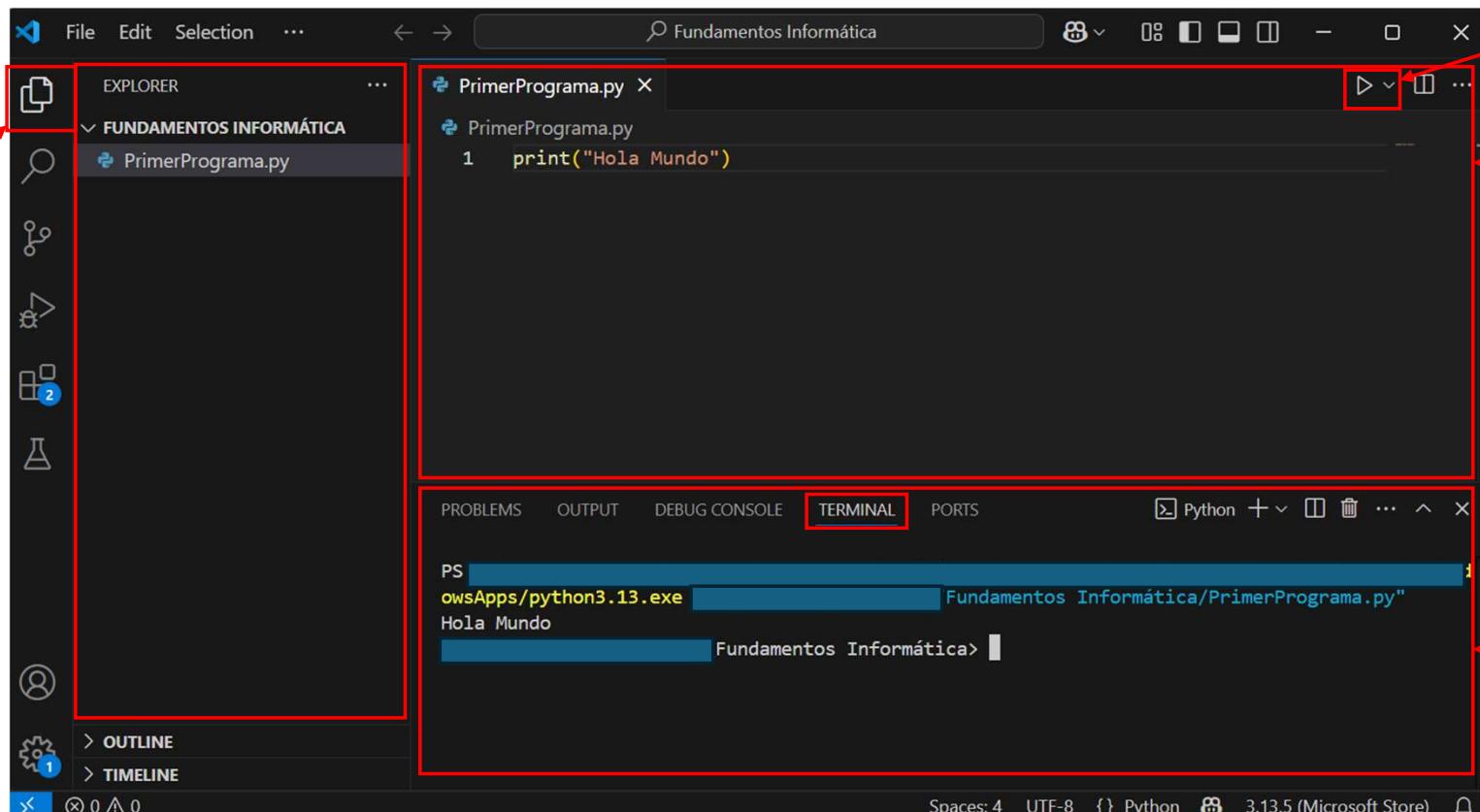
Hola Mundo





Hola Mundo

Navegador



The screenshot shows the Visual Studio Code (VS Code) interface. On the left is the Explorer sidebar, which has a red box around it and an arrow pointing from the 'Navegador' label. It displays a folder named 'FUNDAMENTOS INFORMÁTICA' containing a file named 'PrimerPrograma.py'. The main workspace shows the code for 'PrimerPrograma.py':

```
1 print("Hola Mundo")
```

. To the right of the code is a terminal window with a red box around it and an arrow pointing from the 'Ejecutar el programa.' label. The terminal output shows:

```
PS owsApps/python3.13.exe Fundamentos Informática/PrimerPrograma.py
Hola Mundo Fundamentos Informática>
```

. Below the terminal are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, with 'TERMINAL' being the active tab. The status bar at the bottom indicates 'Spaces: 4' and 'Python 3.13.5 (Microsoft Store)'.

Ejecutar el
programa.

El editor

La consola,
terminal, o
intérprete



Primer ejercicio

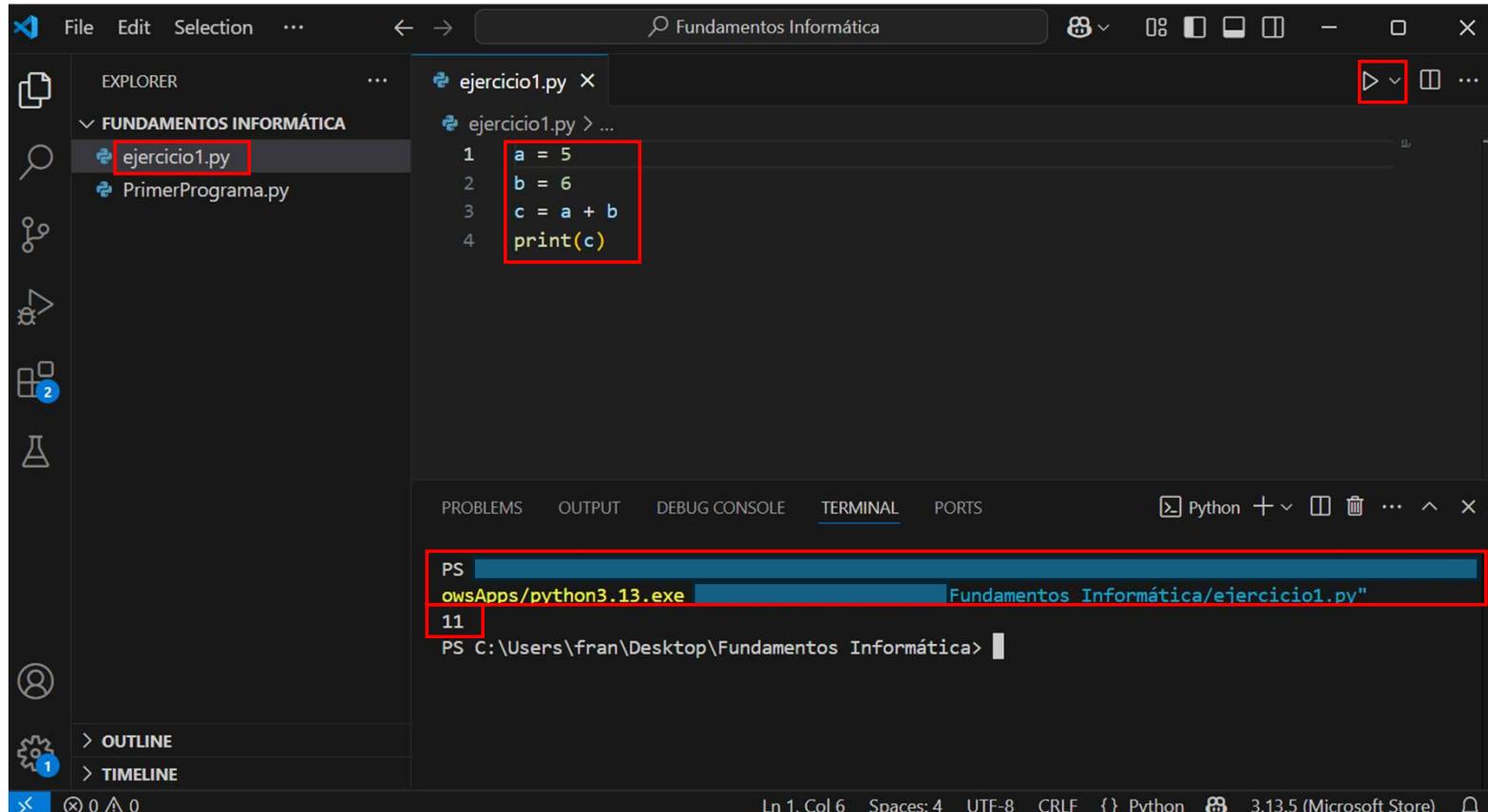
Realiza un programa que consista en asignar el valor 5 a la variable “a”, el valor 6 a la variable “b” y que la suma de ambas variables se guarde en la variable “c”.

A continuación, muestra el resultado por consola.

Guarda el programa como “ejercicio1.py” y ejecútalo.



Primer ejercicio



The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. On the left is the Explorer sidebar with a dark theme. It lists two files under the folder 'FUNDAMENTOS INFORMÁTICA': 'ejercicio1.py' (selected) and 'PrimerPrograma.py'. The 'ejercicio1.py' file contains the following code:

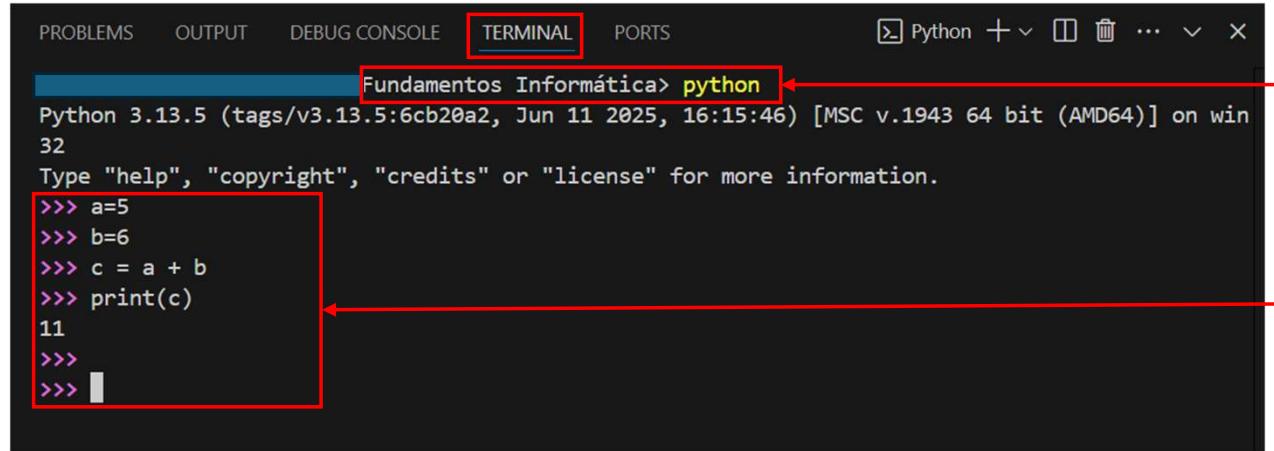
```
a = 5
b = 6
c = a + b
print(c)
```

The code editor has a red box around the first four lines. In the top right corner of the editor area, there is a dropdown menu with a red box around the 'Run' icon (a play button).

At the bottom of the screen is the terminal window. It has a red box around the title bar which reads 'PS owsApps/python3.13.exe Fundamentos Informática/ejercicio1.py'. The terminal prompt shows '11' and the path 'PS C:\Users\fran\Desktop\Fundamentos Informática>'. The status bar at the bottom indicates the file is 'ejercicio1.py', encoding is 'UTF-8', and the Python version is '3.13.5 (Microsoft Store)'.



La consola



The screenshot shows a terminal window with the following interface elements:

- Top bar with tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (highlighted with a red box), and PORTS.
- Top right: Python icon with dropdown menu, plus sign, minus sign, trash can, three dots, and close button.
- Text area:
 - Header: Fundamentos Informática > python
 - Python version: Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
 - Type "help", "copyright", "credits" or "license" for more information.
 - Code input area (highlighted with a red box):

```
>>> a=5
>>> b=6
>>> c = a + b
>>> print(c)
11
>>>
>>> 
```

Ejecutar código sin crear un fichero desde la consola.

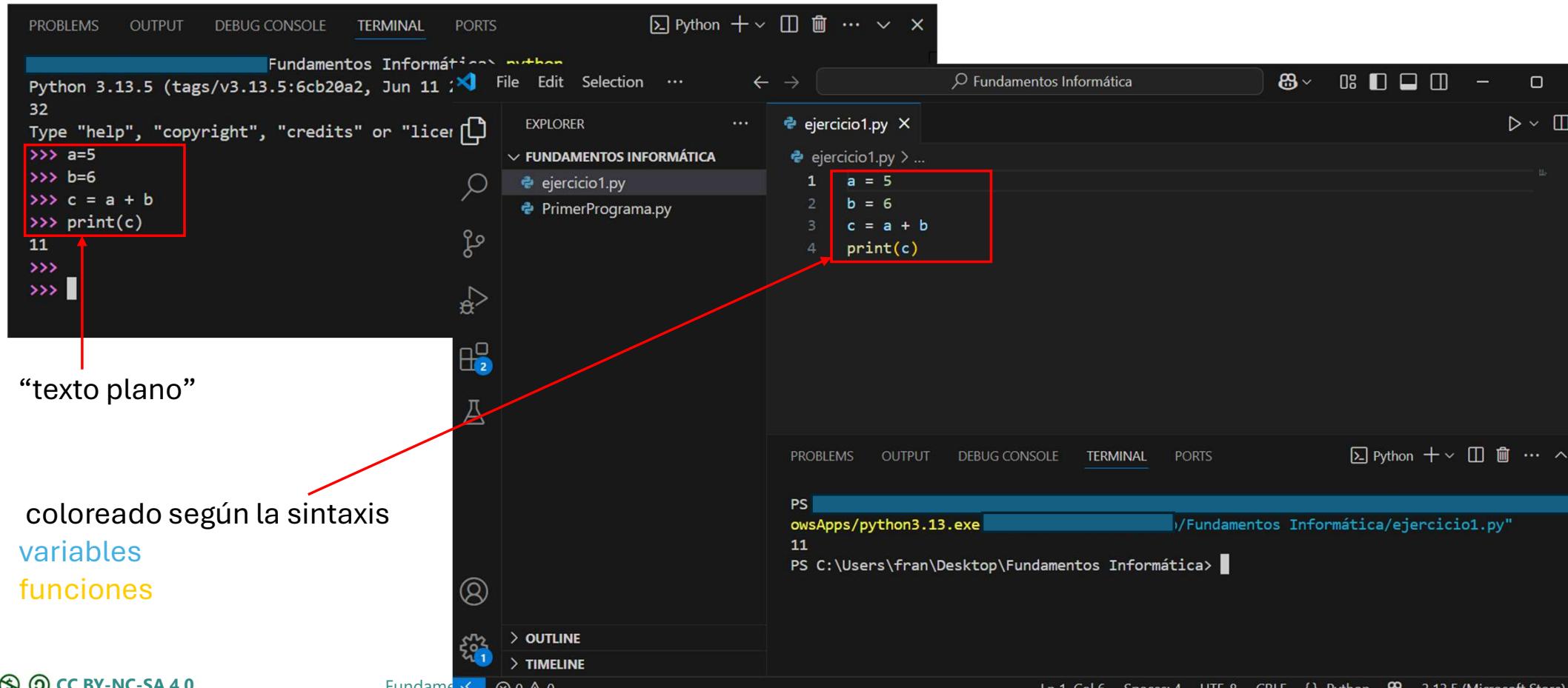
Se escribe la línea y se pulsa “intro”. Si no se escribe nada es un salto de línea.

Problemas:

- 1- No se guarda en el código generado.
- 2- El código es “texto plano”.
- 3- No integra algunas herramientas como depurador...



La consola vs Editor



The screenshot illustrates the comparison between a terminal window and a code editor in a development environment.

Terminal View: Shows a Python 3.13.5 session with the following input and output:

```
>>> a=5
>>> b=6
>>> c = a + b
>>> print(c)
11
```

A red box highlights the first four lines of code, and a red arrow points from this box to the code editor below, labeled "text plano".

Code Editor View: Shows the same code in a file named `ejercicio1.py`:

```
1 a = 5
2 b = 6
3 c = a + b
4 print(c)
```

A red box highlights the same four lines, and a red arrow points from this box to the terminal window above, labeled "coloreado según la sintaxis".

Bottom Terminal View: Shows the command-line interface with the following output:

```
PS C:\Users\fran\Desktop\Fundamentos Informática>
```

The status bar at the bottom indicates the following information:

Ln 1, Col 6 Spaces: 4 UTF-8 CRLF {} Python 8 3.13.5 (Microsoft Store)



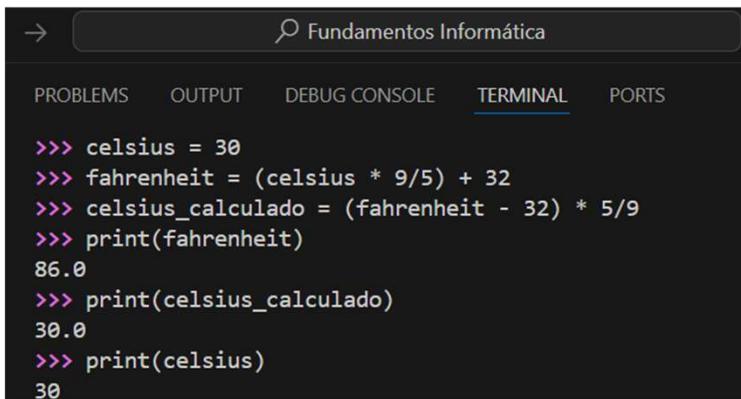
Python como calculadora

Utilizando la consola de Python:

1. Asigna una temperatura de 38 grados Celsius a una variable.
2. Convierte ese valor a Fahrenheit.
$$F = (C \times \frac{9}{5}) + 32$$
3. Convierte el resultado nuevamente a Celsius (variable C2): .
$$C = (F - 32) \times \frac{5}{9}$$
4. Muestra ambos resultados.

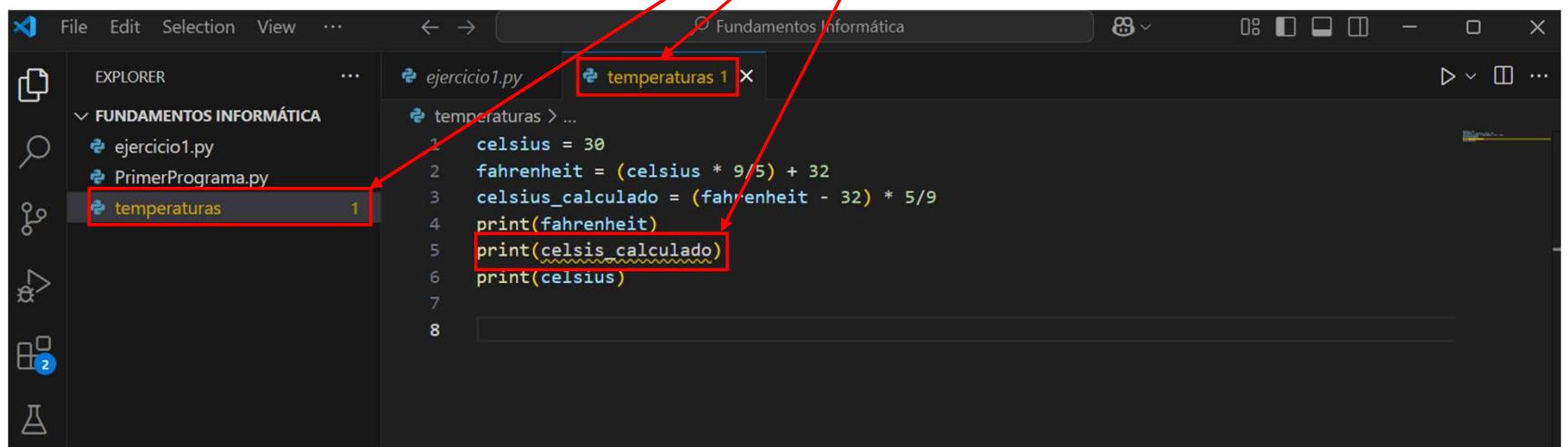


Python como calculadora



```
→ Fundamentos Informática
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
>>> celsius = 30
>>> fahrenheit = (celsius * 9/5) + 32
>>> celsius_calculado = (fahrenheit - 32) * 5/9
>>> print(fahrenheit)
86.0
>>> print(celsius_calculado)
30.0
>>> print(celsius)
30
```

Más sencillo localizar errores
usando el editor



```
File Edit Selection View ...
EXPLORER ...
FUNDAMENTOS INFORMÁTICA
ejercicio1.py
PrimerPrograma.py
temperaturas 1
temperaturas > ...
1 celsius = 30
2 fahrenheit = (celsius * 9/5) + 32
3 celsius_calculado = (fahrenheit - 32) * 5/9
4 print(fahrenheit)
5 print(celsis_calculado)
6 print(celsius)
7
8
```



Operadores básicos en Python

- **Aritméticos:**
+ (suma), - (resta), * (multiplicación), / (división), // (división entera), % (módulo o resto), ** (potencia o exponente)
- **Comparación:**
== (igual a), != (distinto de), > (mayor que), < (menor que), >= (mayor o igual que), <= (menor o igual que)
- **Lógicos:**
and (y lógico), or (o lógico), not (negación lógica)
- **Asignación:**
= (asignación), += (suma y asignación), -= (resta y asignación), *= (multiplicación y asignación), /= (división y asignación), //=(división entera y asignación), %= (módulo y asignación), **= (potencia y asignación)
- **Identidad:**
is (es el mismo objeto), is not (no es el mismo objeto)
- **Pertenencia:**
in (está en), not in (no está en)
- **Bit a Bit:**
& (AND bit a bit), | (OR bit a bit), ^ (XOR bit a bit), ~ (NOT bit a bit), << (desplazamiento a la izquierda), >> (desplazamiento a la derecha)



Funciones básicas en Python

- **Aritméticas:**
 - `abs()` → Valor absoluto
 - `round()` → Redondea un número
 - `pow(x, y)` → Potencia x y y
 - `max() / min()` → Máximo / Mínimo entre varios valores
 - ...
- **Lógicos:**
 - `and` → Verdadero si ambas condiciones son verdaderas
 - `or` → Verdadero si una o ambas condiciones son verdaderas
 - `not` → Invierte el valor lógico (negación)
 - ...
- **Entrada / salida:**
 - `input()` → Lee datos desde la consola como texto
 - `print()` → Muestra datos en la consola
 - ...
- **Tipos:**
 - `type()` → Muestra el tipo de dato
 - `int()` → Convierte a entero
 - `float()` → Convierte a número decimal
 - `str()` → Convierte a texto
 - ...



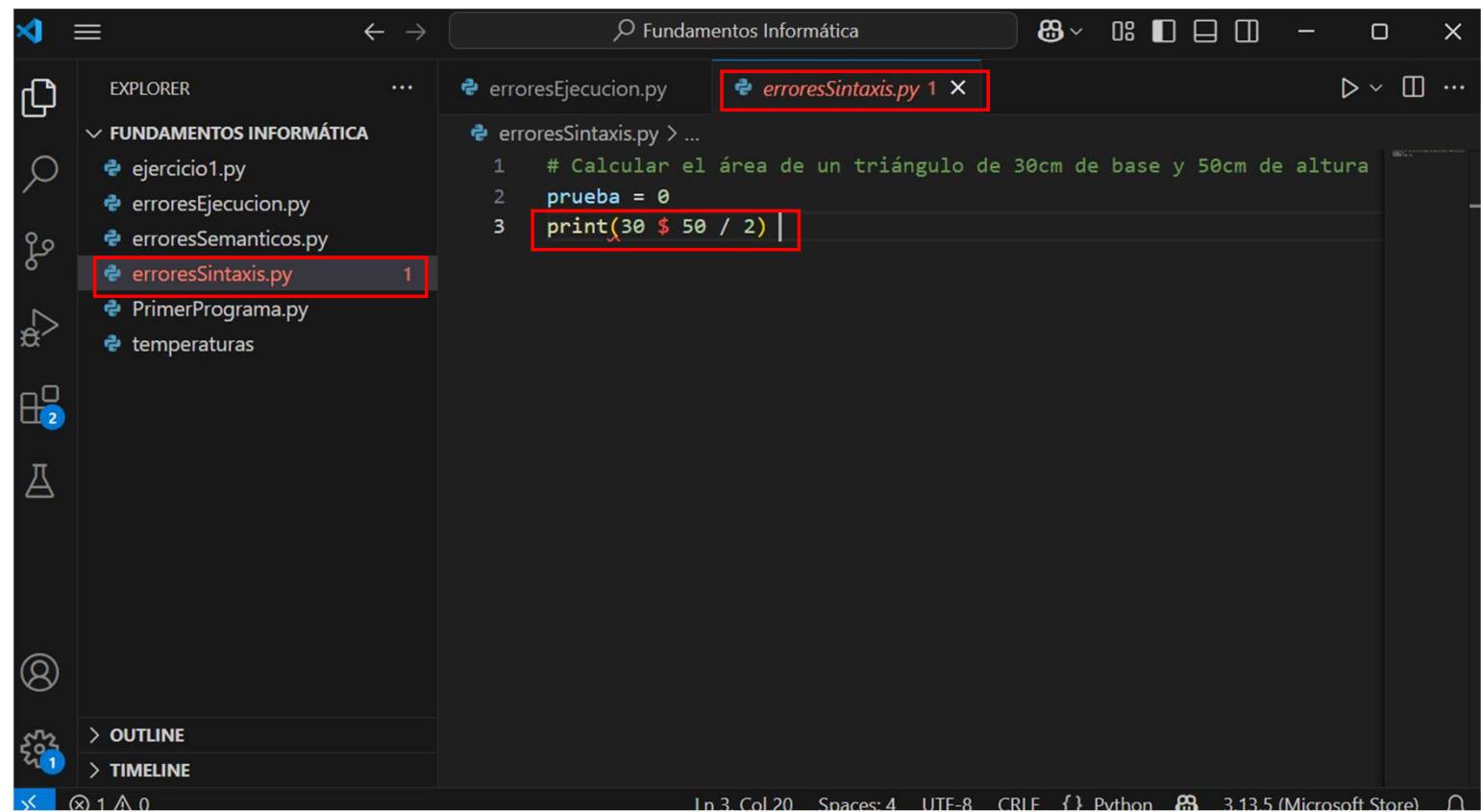
Errores

1. Los errores en programación se denominan "gazapos" ("bugs" en inglés), y encontrar esos errores se llama "depuración" ("debugging" en inglés).
2. Podemos catalogar los errores en tres tipos, por orden creciente de su dificultad de detección



Errores sintácticos

- Son fáciles de encontrar porque Python los señala.
- El error de sintaxis de poner “\$” en lugar de “*”.



The screenshot shows a Microsoft Visual Studio Code interface. The left sidebar has icons for Explorer, Search, Open, and Outline/Timeline. The Explorer panel shows a folder named 'FUNDAMENTOS INFORMÁTICA' containing files: ejercicio1.py, erroresEjecucion.py, erroresSemanticos.py, erroresSintaxis.py (which is selected and highlighted with a red border), PrimerPrograma.py, and temperaturas. The main editor area shows the content of 'erroresSintaxis.py':

```
# Calcular el área de un triángulo de 30cm de base y 50cm de altura
prueba = 0
print(30 $ 50 / 2)
```

The line 'print(30 \$ 50 / 2)' contains a syntax error where a dollar sign (\$) is used instead of a multiplication sign (*). This line is also highlighted with a red border. The status bar at the bottom indicates: In 3. Col 20 Spaces: 4 UTF-8 CR LF { Python 3.13.5 (Microsoft Store) □



Errores en tiempo de ejecución

- Python no detectará el problema hasta que no llegue a ejecutar esa línea. (aunque Visual Code a veces si)
- Ocurre un error al utilizar la variable “dos” que no está definida.

The screenshot shows a Visual Studio Code interface with the following code in the editor:

```
# Calcular el área de un triángulo
# de 30cm de base y 50cm de altura
prueba = 2
print(30 * 50 / dos)
```

The line `print(30 * 50 / dos)` is highlighted in yellow. In the bottom right corner of the code editor, there is a small red box highlighting the error message in the terminal:

```
Traceback (most recent call last):
File "Fundamentos Informática\erroresEjecucion.py", line 4, in <module>
    print(30 * 50 / dos)
          ^
NameError: name 'dos' is not defined
```

- Otros no se pueden detectar hasta que el código se ejecuta.
- Ocurre un error al dividir entre cero.

The screenshot shows a Visual Studio Code interface with the same code as the previous screenshot. However, the variable `prueba` has been set to 0:

```
# Calcular el área de un triángulo
# de 30cm de base y 50cm de altura
prueba = 0
print(30 * 50 / prueba)
```

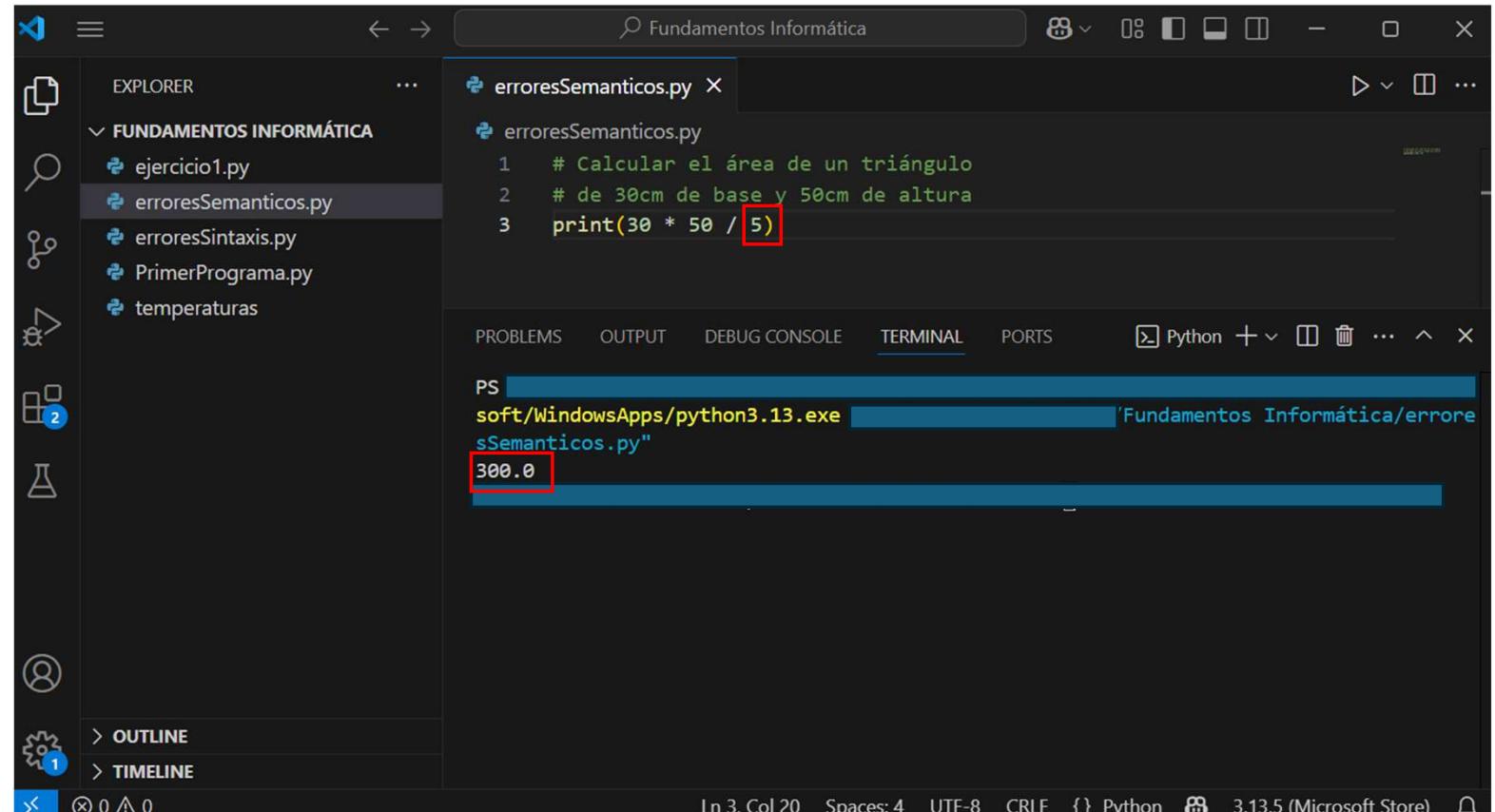
The line `print(30 * 50 / prueba)` is highlighted in yellow. The terminal output shows the error:

```
Traceback (most recent call last):
File "Fundamentos Informática\erroresEjecucion.py", line 4, in <module>
    print(30 * 50 / prueba)
          ^
ZeroDivisionError: division by zero
```



Errores semánticos

- Son difíciles de encontrar.
- El error semántico al calcular el área de un triángulo.
- En lugar de dividir por 2, hemos dividido por 5.



The screenshot shows the Visual Studio Code (VS Code) interface. On the left is the Explorer sidebar with a tree view of files in the 'FUNDAMENTOS INFORMÁTICA' folder: ejercicio1.py, erroresSemanticos.py (selected), erroresSintaxis.py, PrimerPrograma.py, and temperaturas. The main editor area displays the code for 'erroresSemanticos.py':

```
# Calcular el área de un triángulo
# de 30cm de base y 50cm de altura
print(30 * 50 / 5)
```

A red box highlights the division operation `/ 5`. In the bottom right corner of the code editor, there is a small red box around the output `300.0` in the terminal tab, which corresponds to the highlighted division in the code. The terminal tab is active and shows the command `soft/WindowsApps/python3.13.exe` running the script. The status bar at the bottom indicates the code is in Python mode, version 3.13.5 from the Microsoft Store.



Ejercicios

1. Encuentra cuál es el mayor de estos números: 2^{15} , 3^{12} y 5^{10} .
2. Usando las funciones chr y ord, encuentra cuál es la letra que está 10 posiciones más adelante de la A
3. Usando las funciones chr y hex (y tu cabeza) encuentra cuál es el código binario de la letra 'a' y de la letra 'A' y verifica que se diferencian tan solo en un bit
4. Escribe una expresión que calcule el resto de dividir 500 entre 7. Comprueba que sale 3.
5. Escribe un programa que haga salir en pantalla una línea formada por 80 asteriscos.
6. Calcula la raíz cuadrada de 2 con cinco decimales.
7. ¿Qué crees que debería salir al poner type(1/2)? Comprueba qué sale. Comprueba también el resultado de la operación. ¿Y con type(1//2)? ¿Cuál es la diferencia entre el operador / y el //?
8. Prepara una variable n con el valor 5. Calcular la raíz n-sima de 2. Debe salir 1.1486983549970351.