



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.

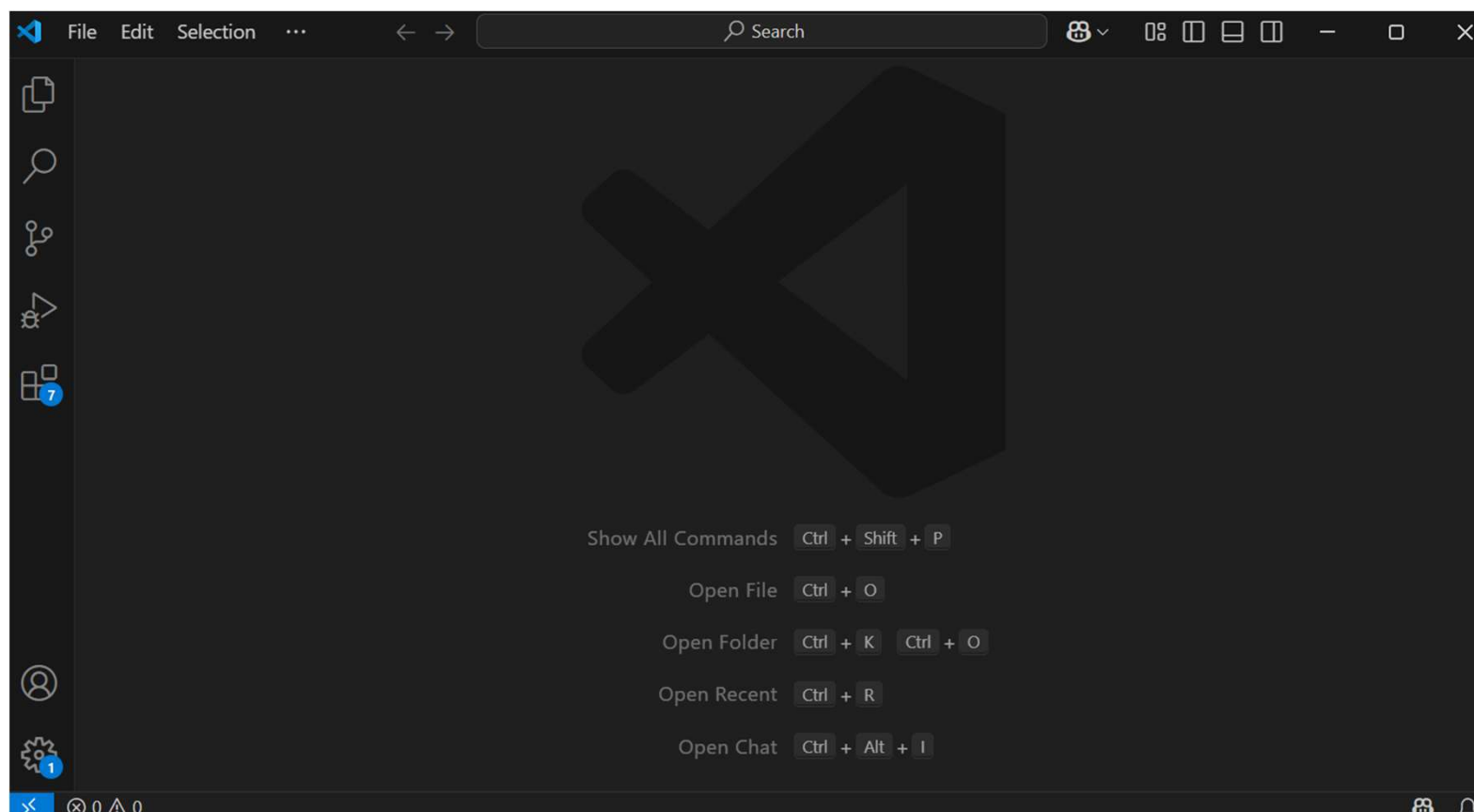


Instalación

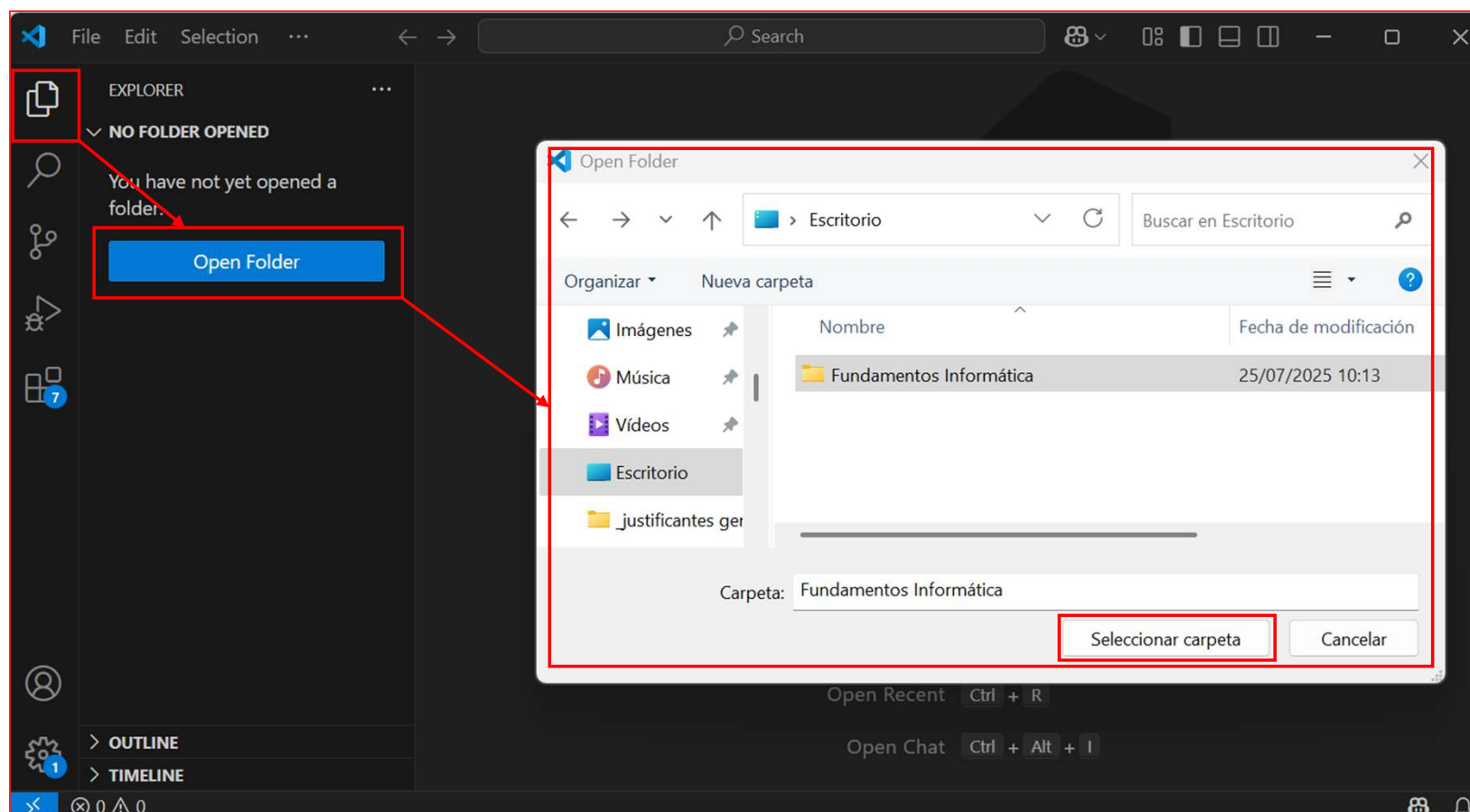
- Python
 - <https://python.org>
- Visual Studio Code
 - <https://code.visualstudio.com/>



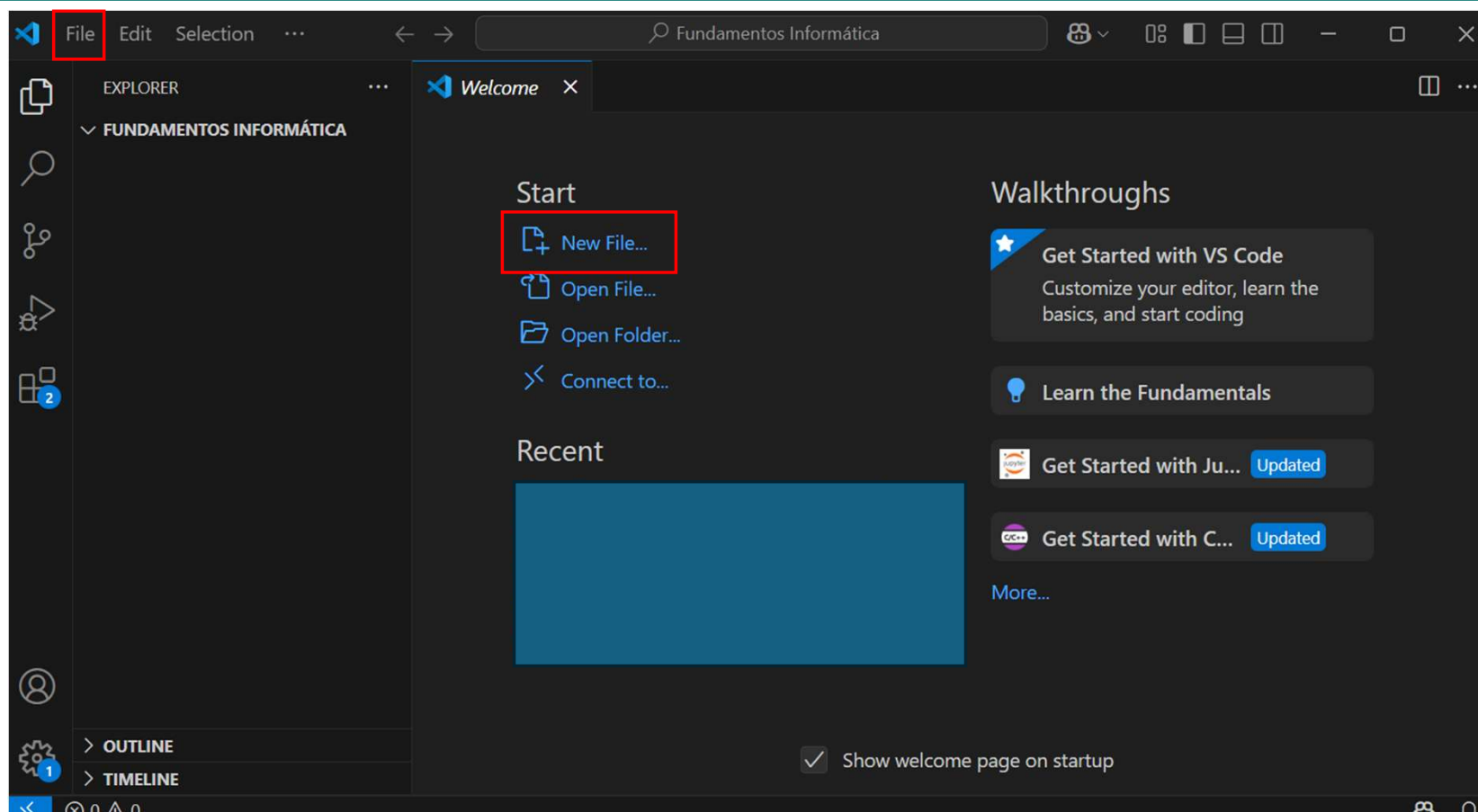
Introducción a Visual Studio Code



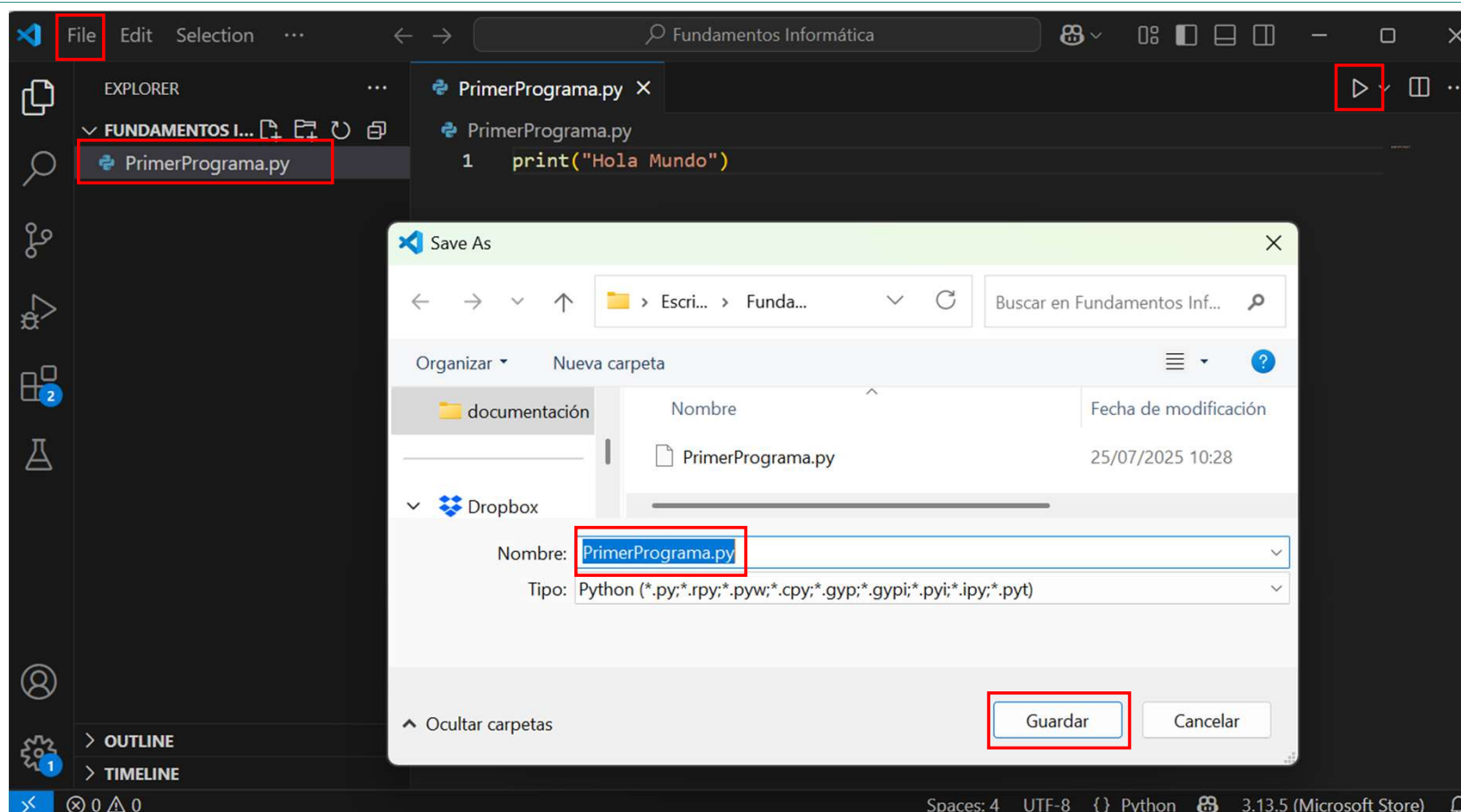
Seleccionar un directorio de trabajo



Crear un primer programa "Hola Mundo"

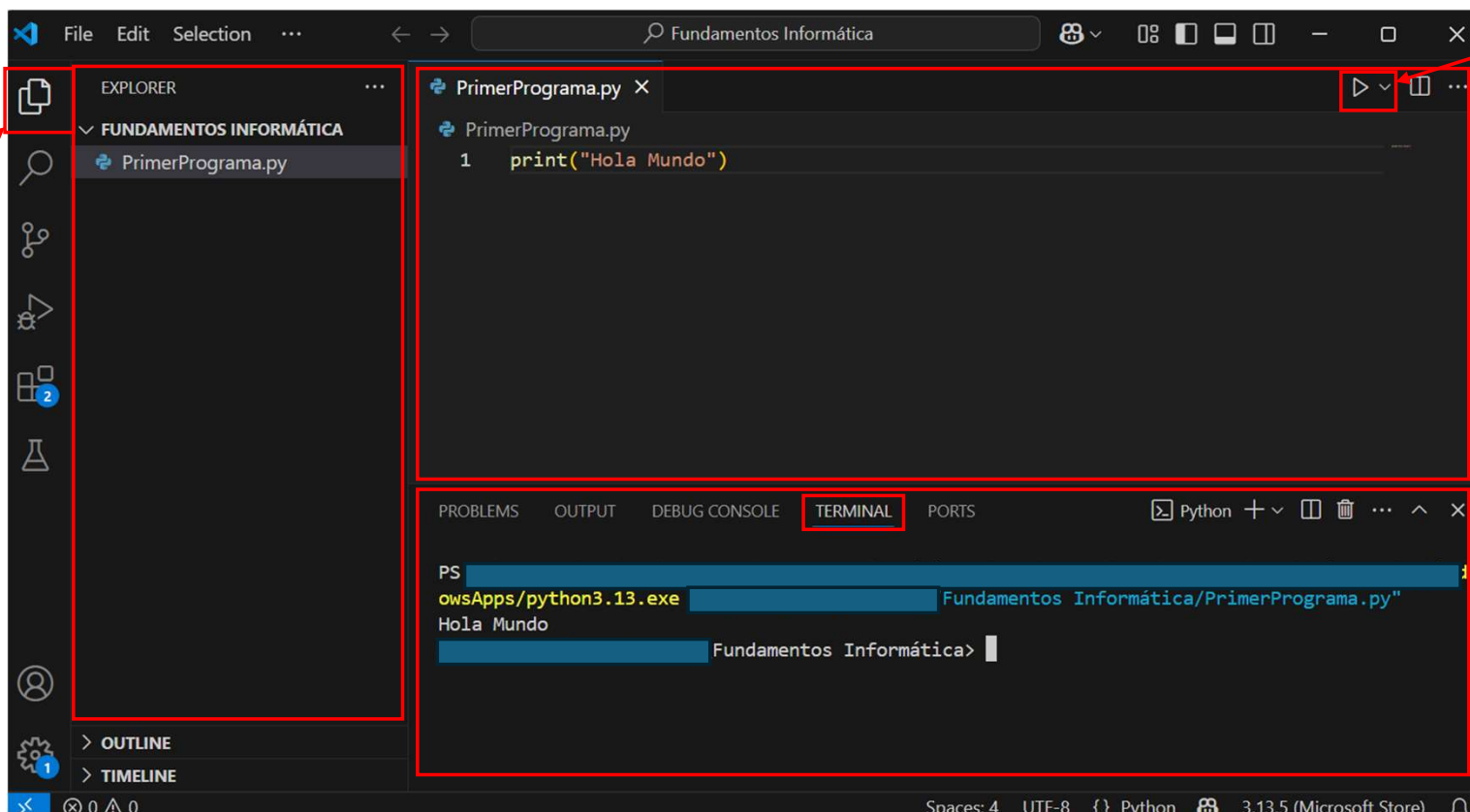


Hola Mundo



Hola Mundo

Navegador



Ejecutar el programa.

El editor

La consola, terminal, o intérprete

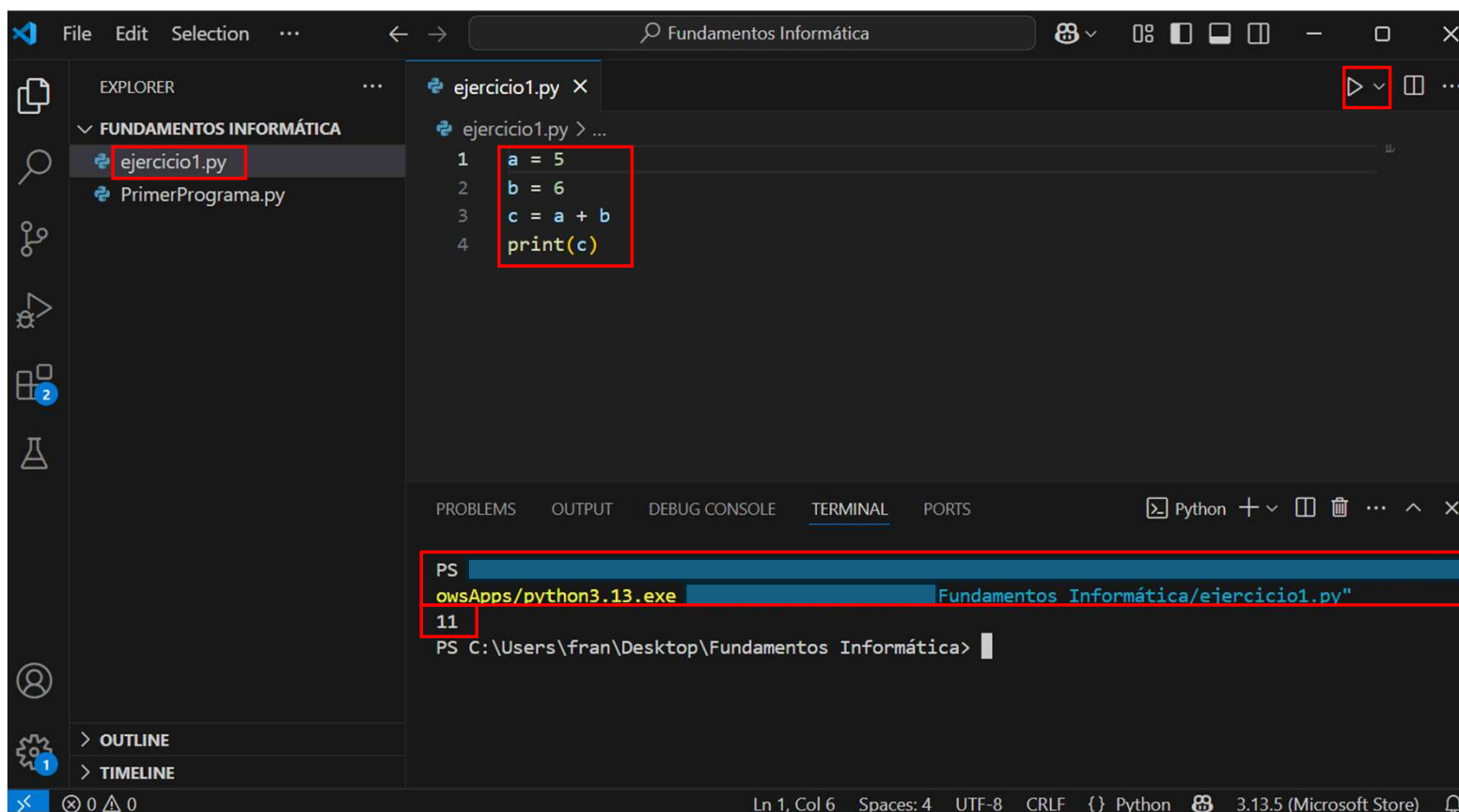


Primer ejercicio

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



```
File Edit Selection ... Fundamentos Informática
```

EXPLORER

- FUNDAMENTOS INFORMÁTICA
 - ejercicio1.py
 - PrimerPrograma.py

ejercicio1.py X

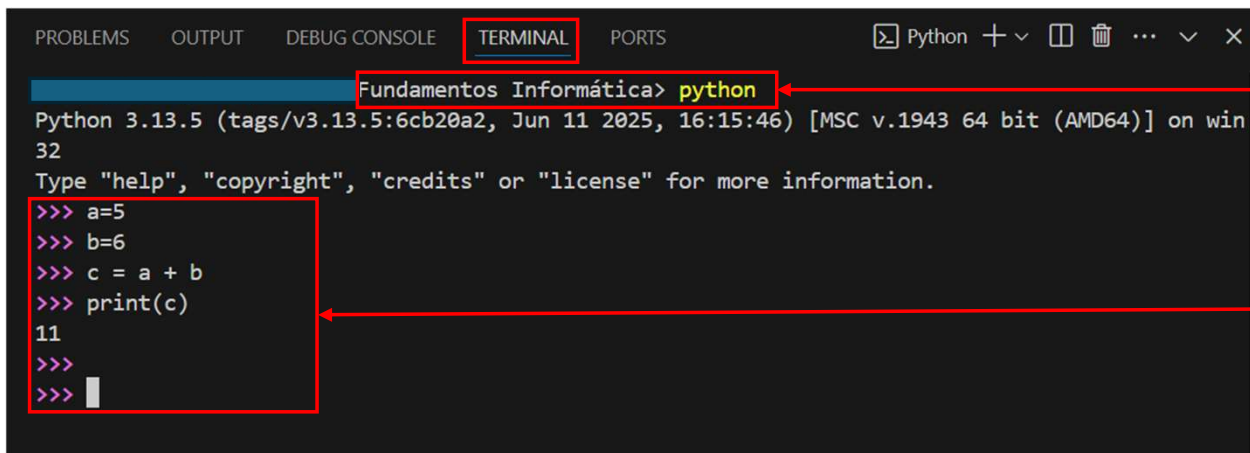
```
1 a = 5
2 b = 6
3 c = a + b
4 print(c)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + -

```
PS
owsApps/python3.13.exe Fundamentos Informática/ejercicio1.py"
11
PS C:\Users\fran\Desktop\Fundamentos Informática>
```

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

La consola



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
Python  + v  [icon]  [icon]  ...  v  x

Fundamentos Informática> python
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.
>>> 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 image shows a side-by-side comparison of code execution in a console versus a code editor. On the left, a Python 3.13.5 terminal window shows a prompt where the code `a=5`, `b=6`, `c = a + b`, and `print(c)` is entered line by line. On the right, a code editor window shows the same code in a file named `ejercicio1.py`. A red box highlights the code in both the console and the editor. A red arrow points from the console to the editor, indicating the flow of code from the console to the editor. The console output shows the code being executed line by line, with the prompt `>>>` and the output `11` visible. The editor window shows the code being edited, with line numbers 1 through 4 visible. The code in the editor is `a = 5`, `b = 6`, `c = a + b`, and `print(c)`. The console window also shows the prompt `Type "help", "copyright", "credits" or "license()"` and the prompt `>>>`. The editor window shows the file `ejercicio1.py` and the file `PrimerPrograma.py` in the Explorer pane. The console window shows the prompt `Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2024)` and the prompt `>>>`. The editor window shows the file `ejercicio1.py` and the file `PrimerPrograma.py` in the Explorer pane. The console window shows the prompt `Type "help", "copyright", "credits" or "license()"` and the prompt `>>>`. The editor window shows the code being edited, with line numbers 1 through 4 visible. The code in the editor is `a = 5`, `b = 6`, `c = a + b`, and `print(c)`. The console window also shows the prompt `>>>` and the output `11`.

“texto plano”

coloreado según la sintaxis

variables

funciones



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): .

4. Muestra ambos resultados.

$$C = (F - 32) \times \frac{5}{9}$$

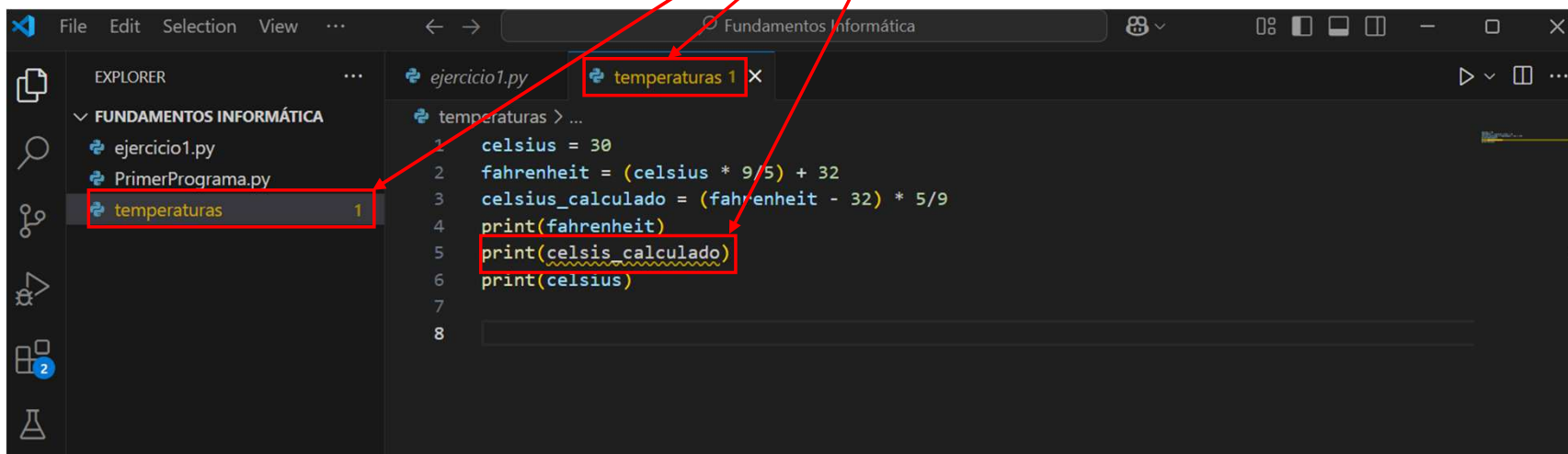
Python como calculadora

```
→ 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 ... Fundamentos Informática

EXPLORER
FUNDAMENTOS INFORMÁTICA
  ejercicio1.py
  PrimerPrograma.py
  temperaturas 1

ejercicio1.py
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 x
 - `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 “*”.

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

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

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

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

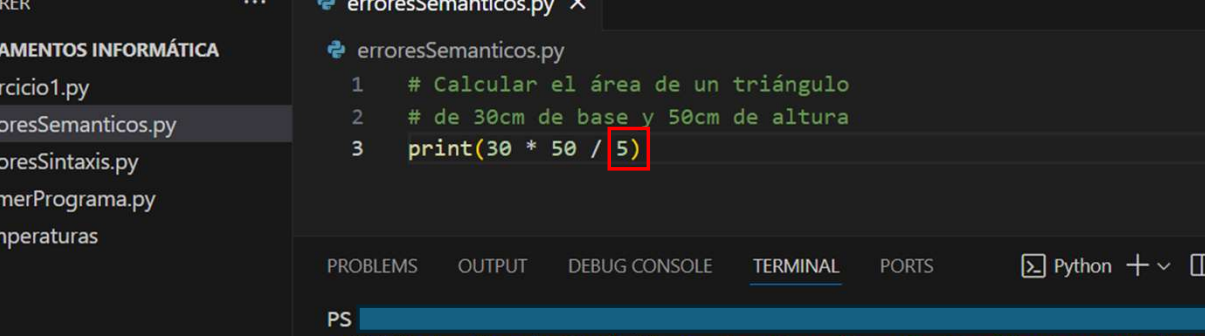
```
1 # Calcular el área de un triángulo
2 # de 30cm de base y 50cm de altura
3 prueba = 0
4 print(30 * 50 / prueba)
```

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 interface. On the left is the Explorer sidebar with a file tree containing 'ejercicio1.py', 'erroresSemanticos.py' (selected), 'erroresSintaxis.py', 'PrimerPrograma.py', and 'temperaturas'. Below the Explorer are the Outline and Timeline views. The main editor displays the file 'erroresSemanticos.py' with the following code:

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

The number '5' in the print statement is highlighted with a red box. At the bottom, the TERMINAL panel shows the command prompt output:

```
PS
soft/WindowsApps/python3.13.exe 'Fundamentos Informática/erroresSemanticos.py'
300.0
```

The output '300.0' is highlighted with a red box. The status bar at the bottom indicates the current cursor position is 'Ln 3, Col 20' and the Python version is '3.13.5 (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.