MATEMATICA 1 2025-3

30 de noviembre de 2025

Luis Llumiquinga
Introducción a Python

Ing. Santiago Mosquera
MOVILIS

Resumen

El presente informe recopila las soluciones y aprendizajes obtenidos durante el desarrollo de los retos 1 al 11 del curso *Introducción a Python*. Cada actividad permitió reforzar conceptos fundamentales como impresión en consola, variables, tipos de datos, entrada de usuario, operaciones aritméticas, depuración de código y manipulación básica de datos. A través de la práctica guiada, fue posible comprender la utilidad de Python para resolver tareas sencillas, estructurar programas básicos y fortalecer la lógica de programación. Este documento analiza cada reto individualmente, destacando el propósito, el proceso de solución y los conocimientos adquiridos.

Introducción

Python es uno de los lenguajes de programación más utilizados a nivel mundial debido a su facilidad de aprendizaje, sintaxis clara y versatilidad. El curso *Introducción a Python* propone una serie de retos diseñados para que el estudiante adquiera habilidades esenciales en la construcción de scripts, manejo de variables, interacción con el usuario, depuración y operaciones matemáticas.

Este informe presenta un análisis de los retos 1 al 11, abordando la solución implementada y los aprendizajes obtenidos en cada uno. El objetivo es dejar documentada la evolución del proceso formativo y destacar la importancia de cada ejercicio dentro del desarrollo de competencias básicas en programación.

Desarrollo

Reto 1: Crear y modificar páginas HTML

Aunque es un reto introductorio no relacionado directamente con Python, permitió familiarizarse con entornos de trabajo y edición de archivos. Se aprendió a modificar código, actualizar contenido y comprender la estructura general de un archivo HTML. Esto facilitó el manejo de editores antes de comenzar con Python.

Aprendizaje clave: familiarización con herramientas de desarrollo y estructura básica de archivos.

Reto 2: Impresión de un mensaje con print()

El reto consistió en crear un archivo llamado *reto2.py* e imprimir el mensaje:
“Mis primeros pasos en Python.”

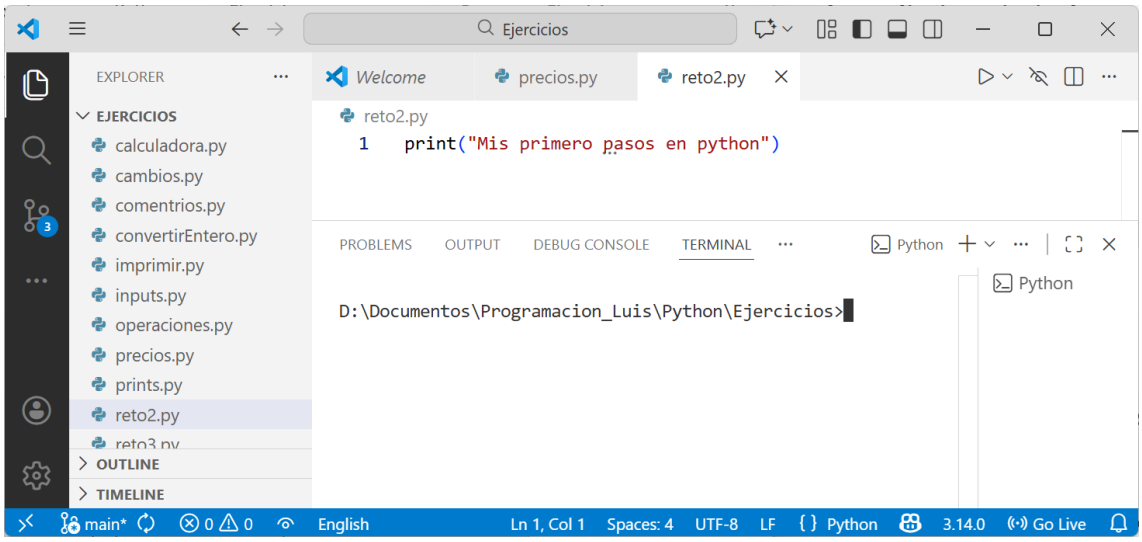
Se utilizó la función print(), una de las herramientas fundamentales de Python.

Aprendizajes principales:

- Uso básico de la función print().
- Creación y ejecución de archivos *.py*.
- Comprensión del flujo básico de un programa.

Figura 1

Primeros pasos en Python



Reto 3: Declaración de variables y tipos de datos

En *reto3.py* se crearon variables de tipo *string* y *entero*:

- **nombre = "Leo"**
- **apellido = "Messi"**
- **apodo = "La Pulga"**
- **balonesDeOro = 5**

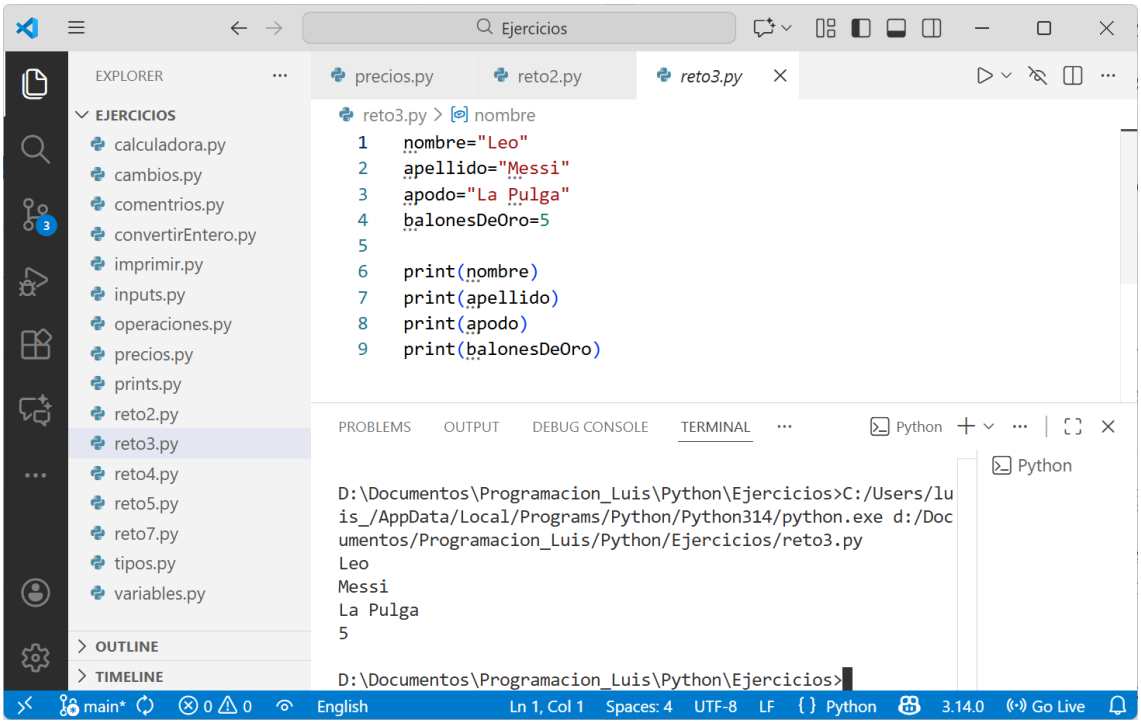
Posteriormente se imprimieron cada una de ellas.

Aprendizajes principales:

- Declaración de variables en Python.
- Diferenciación entre tipos de datos (str y int).
- Impresión múltiple de valores.

Figura 2

Uso de variales



Reto 4: Concatenación de variables

En *reto4.py*, se crearon variables con nombre, apellido y edad del estudiante, y se imprimió un mensaje con el formato:

tienes años

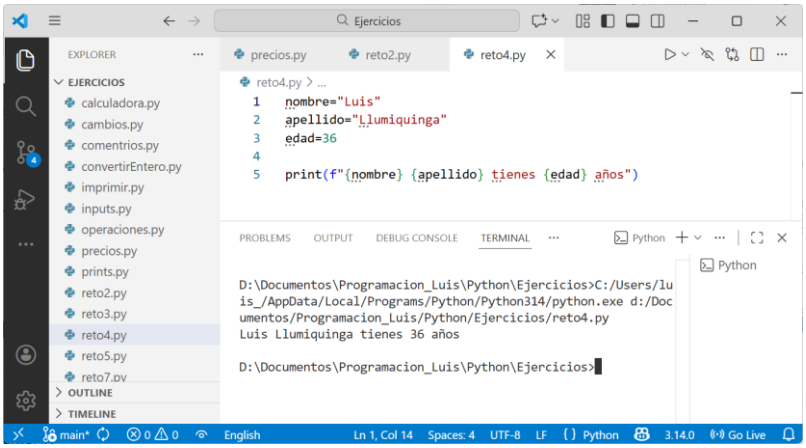
La solución requirió concatenar texto y variables numéricas usando print().

Aprendizajes principales:

- Concatenación de strings.
- Conversión automática de enteros al imprimir.
- Formatos básicos de impresión en consola.

Figura 3

Impresión de variales con print



Reto 5: Entrada de usuario con input()

Se solicitó al usuario ingresar su nombre y apellido y mostrar un mensaje como:

eres un crack

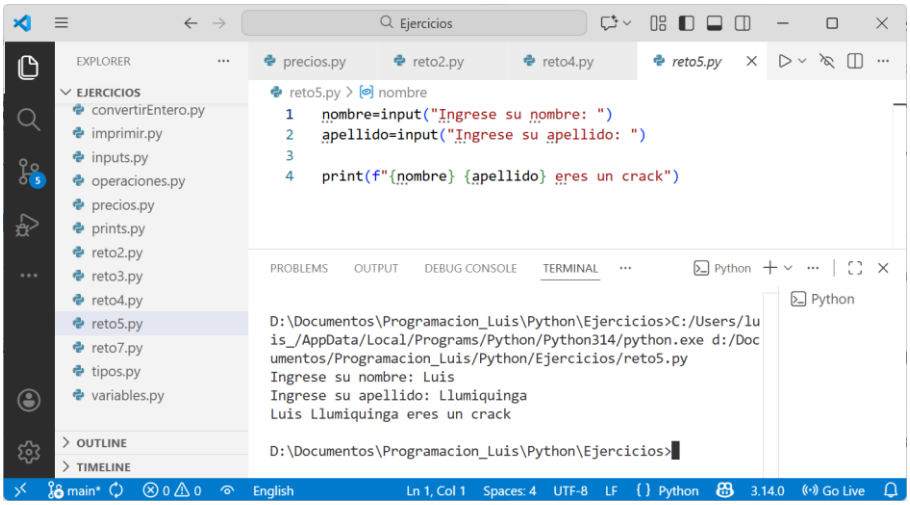
Este reto introdujo la interacción usuario–programa mediante input().

Aprendizajes principales:

- Lectura de datos desde teclado.
- Almacenamiento de entradas en variables.
- Construcción de mensajes personalizados.

Figura 4

Uso de input



Reto 6: Uso del depurador (debug) y breakpoints

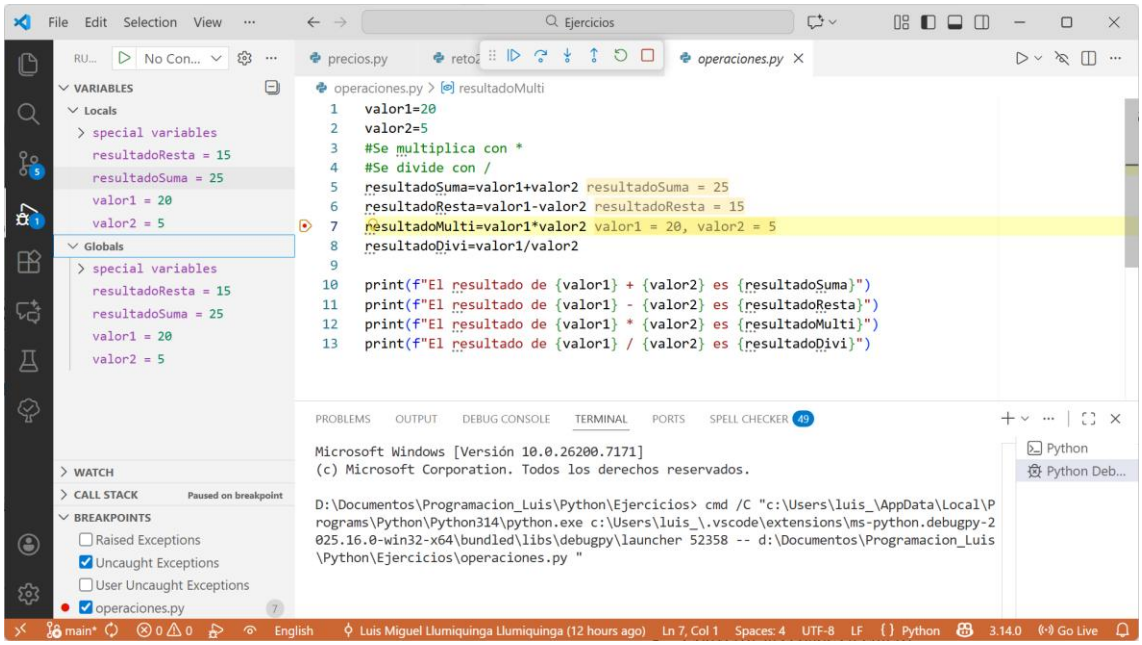
En este reto se colocó un breakpoint en un archivo llamado *operaciones.py* para observar el valor de la suma antes de imprimirse.

Aprendizajes principales:

- Funcionamiento del modo depuración en un IDE.
- Inspección de valores de variables.
- Análisis paso a paso del flujo de ejecución.
- Importancia del debugging para corregir errores.

Figura 5

Uso del breakpoint en Python



Reto 7: Operaciones aritméticas con entradas del usuario

El reto solicitó:

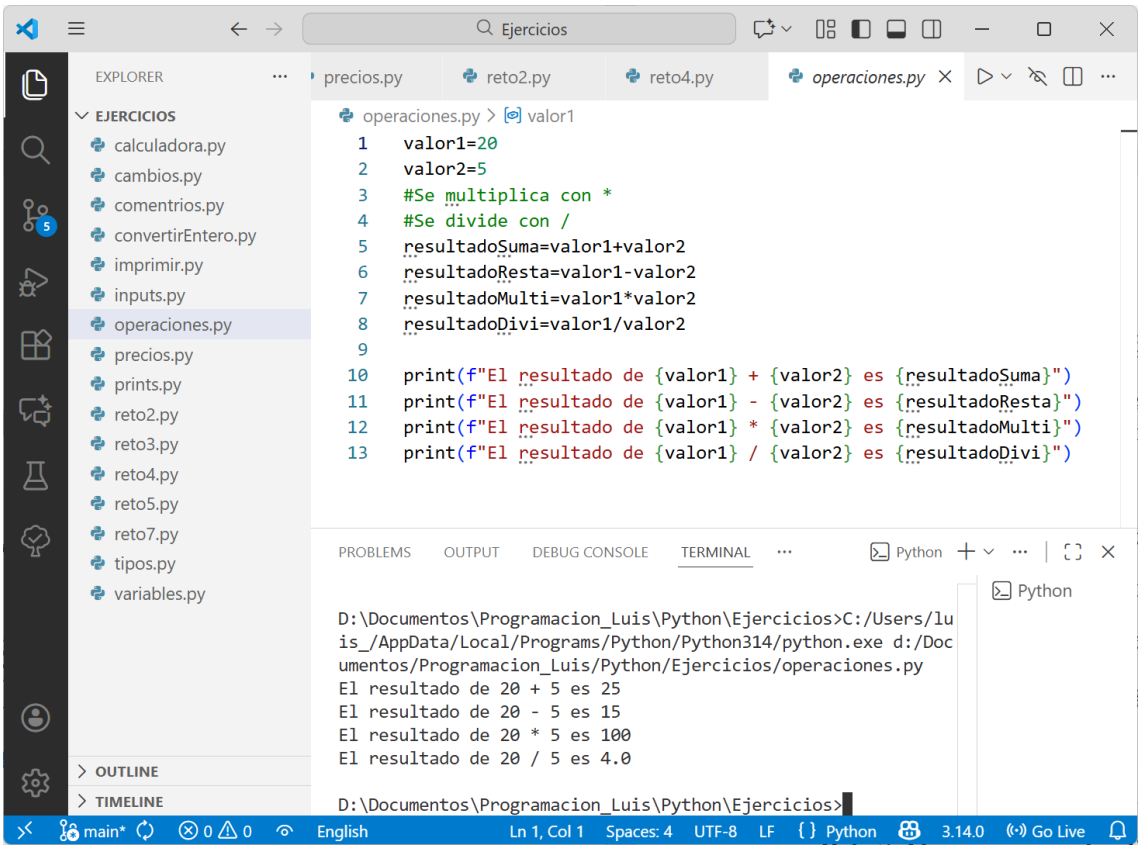
- Pedir nombre y edad del usuario.
- Pedir nombre y edad de un amigo.
- Convertir las edades a entero.
- Calcular la suma y diferencia.
- Imprimir mensajes como:
    - *"La suma de las edades de Juan y Pedro es 40"*
    - *"La diferencia de edades es…"*

Aprendizajes principales:

- Conversión de datos (casting) con int().
- Operaciones aritméticas básicas.
- Interacción avanzada con el usuario.
- Buenas prácticas en estructuración de mensajes.

Figura 6

Operaciones básicas en Python



Reto 8: Cálculo del valor total de una compra

En *precios.py* se pidió:

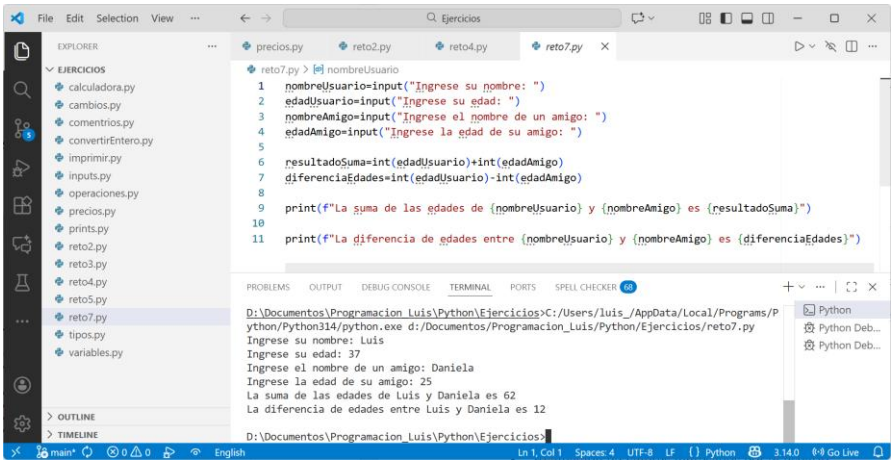
- Nombre del producto.
- Precio (float).
- Cantidad (entero).
- Cálculo del total = precio × cantidad.
- Impresión del mensaje:
  “El valor de es ”

Aprendizajes principales:

- Diferencia entre int() y float().
- Multiplicación de valores numéricos ingresados por el usuario.
- Formatos de mensajes aplicados a compras.

Figura 7

Ingreso de datos y operaciones básicas



Reto 9: Validación de resultados en un juego de dados

En este reto se modificó el archivo *dados.py* para implementar una lógica básica de juego utilizando números aleatorios. Luego del bloque import, se añadió la impresión de dos mensajes introductorios:

- **“** PRUEBA TU SUERTE **”**
- **“Si obtienes un valor mayor a 3, ganas”**

A continuación, se mantuvo el input() original que solicita al usuario presionar una tecla para lanzar los dados.

También se conservó la instrucción que genera un número aleatorio entre 1 y 6 utilizando la función correspondiente de la librería random, así como la impresión del valor obtenido.

La parte central del reto consistió en agregar una validación mediante un condicional if: Si el valor obtenido es mayor a 3, el programa imprime dos mensajes adicionales:

- **“Felicidades Ganaste!!”**
- **“Estás de suerte!!”**

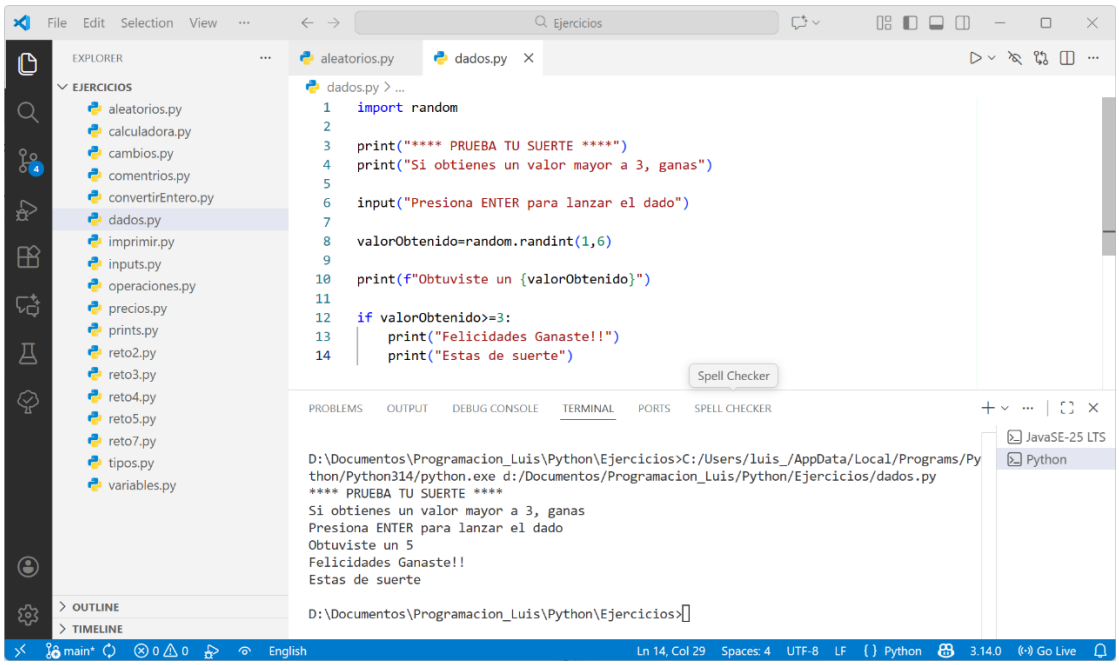
Este reto permitió reforzar tanto el uso de librerías externas como el manejo de estructuras condicionales.

Aprendizajes principales:

- Orden adecuado del código colocando siempre los imports al inicio del archivo.
- Uso de la librería random para generar valores aleatorios.
- Interacción con el usuario a través de input().
- Implementación de condiciones para determinar resultados dentro de un programa.
- Construcción de lógica de flujo para juegos simples basados en valores numéricos.

Figura 8

Uso del condicional IF



Reto 10: Modificación de condiciones para ganar en el juego de dados

En este reto se trabajó nuevamente sobre el archivo *dados.py*, ajustando la lógica del juego para que el usuario únicamente gane si obtiene el número **6** al lanzar el dado. Para lograrlo, fue necesario modificar tanto los mensajes mostrados en pantalla como la validación que determina el resultado.

Primero, se actualizó el mensaje introductorio para indicar claramente la nueva condición de victoria, reemplazando la regla anterior. El programa debía informar al usuario que **solo ganaría si obtenía un 6**, lo cual refuerza el uso de mensajes dinámicos y adecuados al objetivo del programa.

Posteriormente, se mantuvo el funcionamiento que permite al usuario presionar una tecla para realizar el lanzamiento, así como la generación del número aleatorio entre 1 y

6 mediante la librería random. El valor obtenido se imprimió inmediatamente después, tal como en el reto anterior.

La parte central del reto fue la modificación de la estructura condicional.
En lugar de validar si el valor era mayor a 3, la nueva condición quedó de la siguiente forma:

- Si el valor obtenido es **igual a 6**, el programa imprime:
    - **"¡Felicidades, ganaste!"**
    - **"¡Sacaste un 6!"**
- Si el valor es diferente de 6, se puede informar simplemente que **no ganó**, reforzando la retroalimentación al usuario (opcional según diseño del reto).

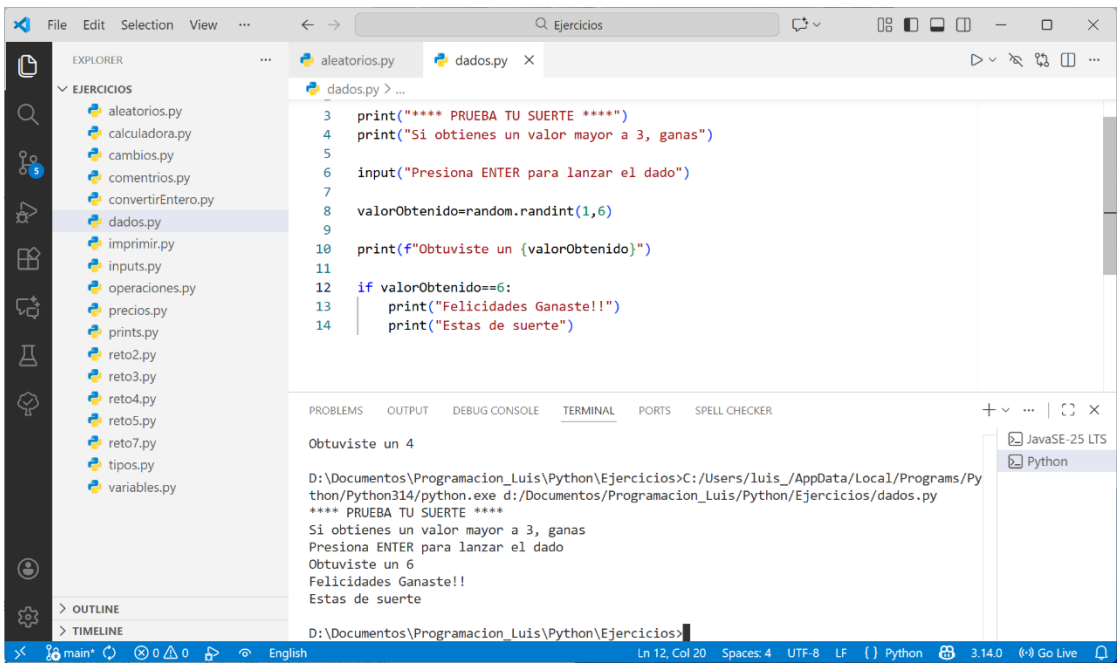
Este reto permitió comprender cómo la lógica de un programa puede modificarse fácilmente ajustando las condiciones evaluadas por las estructuras if.

Aprendizajes principales:

- Actualización de reglas dentro de un programa ya existente.
- Uso de operadores lógicos de igualdad (==).
- Mejora en la claridad de mensajes mostrados al usuario.
- Adaptación del flujo del programa según nuevos requisitos.
- Comprensión del concepto de "condición estricta" para determinar resultados.

Figura 9

Modificación de la condición en el IF



Reto 11: Implementación de un juego con lanzamiento de dos dados

En este último reto se creó un archivo independiente llamado *juegoDados.py*, utilizando como referencia la estructura previamente trabajada en *dados.py*. El objetivo consistió en implementar un juego en el que el usuario lanza dos dados simultáneamente y debe obtener un valor superior a 8 para "salvarse".

Al inicio del programa, se imprimió el mensaje principal que define la regla del juego:

- **"Debes obtener más de 8 para salvarte"**

Posteriormente, al igual que en los retos anteriores, se incluyó un input() para solicitar al usuario que presione la tecla Enter y así realizar el lanzamiento de los dos dados. Para simularlo, se realizaron **dos llamadas independientes** a la función randint de la librería random, guardando cada resultado en una variable distinta. Esto permitió representar el comportamiento realista de dos dados físicos lanzados al mismo tiempo.

En una primera fase, el programa mostraba únicamente los valores obtenidos utilizando el formato:

- **"Ha obtenido <valorDado1> y <valorDado2>"**

Luego de verificar su correcto funcionamiento, se añadió la lógica para **sumar los valores de ambos dados**, almacenando el resultado en una variable llamada total. Tras esto, el mensaje anterior fue modificado para incluir también el valor de la suma:

- **"Ha obtenido <valorDado1> y <valorDado2>, total <total>"**

Finalmente, se incorporó la validación central del reto. Si la suma de los dos dados era **8 o menos**, el programa debía mostrar:

- **"Lo siento no es tu día de suerte, no te salvas"**

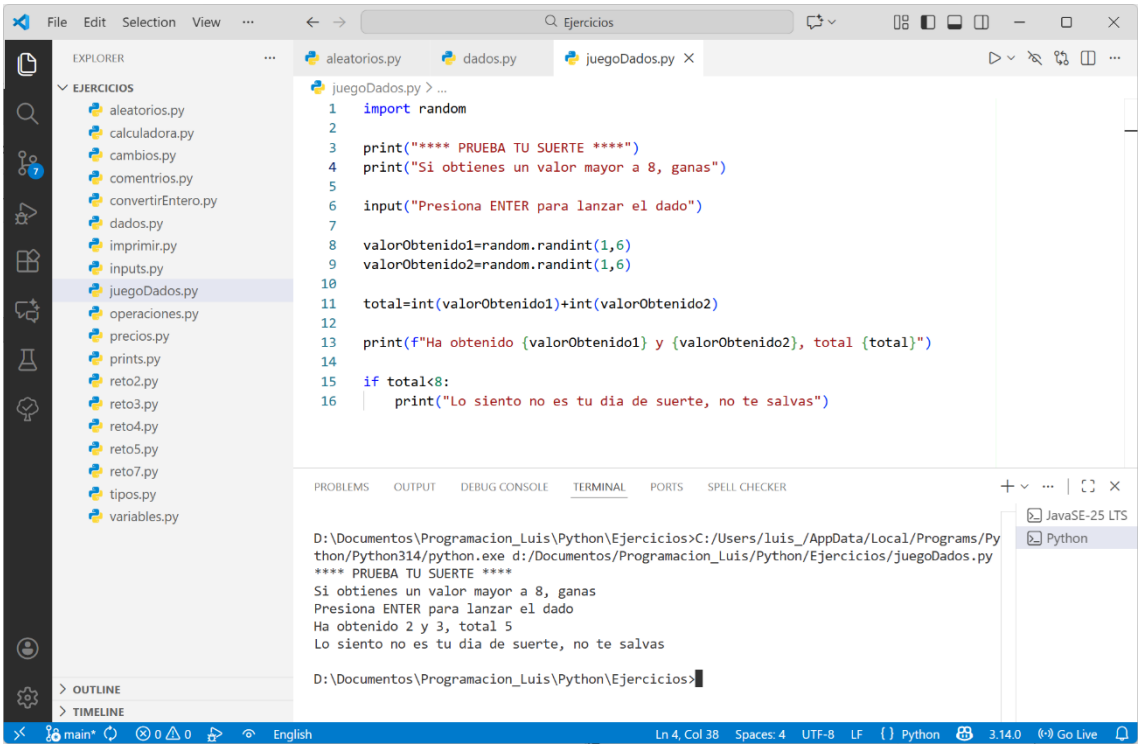
En caso contrario (es decir, si la suma era mayor a 8), el usuario se consideraba "salvado", reforzando así el aprendizaje sobre estructuras condicionales aplicadas a múltiples variables.

Aprendizajes principales:

- Uso simultáneo de múltiples llamadas a funciones aleatorias.
- Almacenamiento y uso de varias variables numéricas relacionadas entre sí.
- Construcción de mensajes dinámicos con múltiples valores.
- Implementación de condiciones con sumas y comparaciones.
- Desarrollo progresivo del programa probando cada etapa antes de avanzar.
- Comprensión de la lógica detrás de juegos simples basados en azar.

Figura 9

Modificación de la condición en el IF



Conclusiones

El desarrollo de los retos permitió adquirir una comprensión sólida de los fundamentos de Python. Cada ejercicio fue diseñado para introducir un concepto clave del lenguaje: desde la impresión de mensajes y manejo de variables hasta operaciones matemáticas, entrada de usuario, condiciones, ciclos, listas y depuración. Esta progresión facilitó el aprendizaje gradual y aplicado, generando una base firme para avanzar hacia niveles más complejos de programación.

Gracias a la práctica constante, fue posible desarrollar habilidades de lógica, resolución de problemas y pensamiento estructurado, esenciales en el campo de la programación.

Referencias

Course. (2025). *Curso: Introducción a Python*. Plataforma Course.

Python Software Foundation. (2024). *Python Documentation*. https://docs.python.org