



## Ejercicios de Python

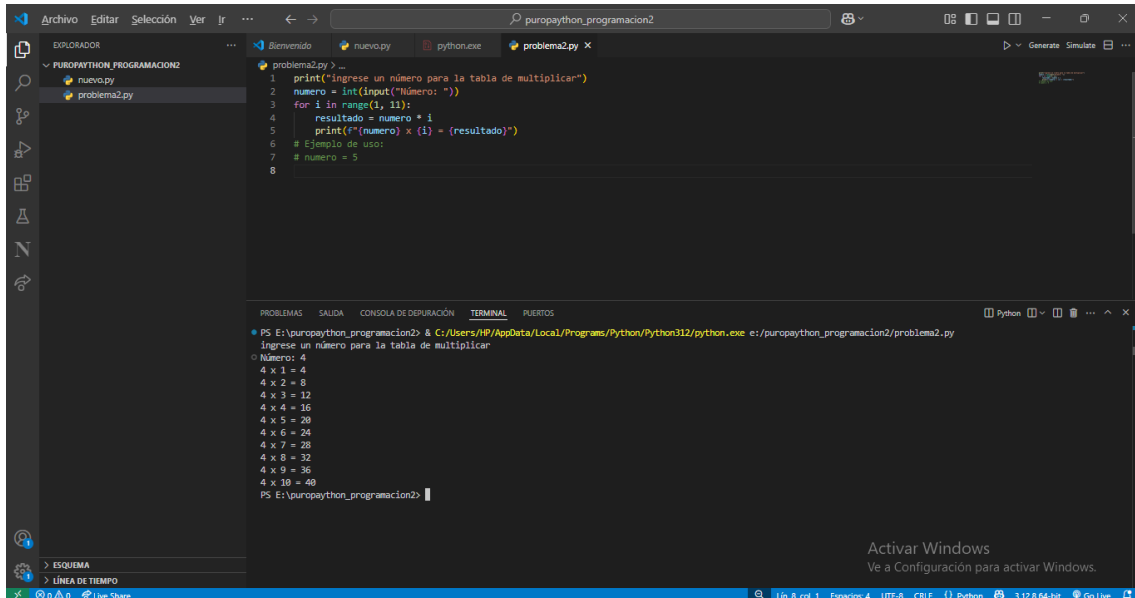
**Ing.** REQUENA LLORENTTY JIMMY NATANIEL

**Integrantes:**

- Marco Aurelio Barriga
- Adaniel Balderrama Orellana
- Carlos Ademar Suárez

## Ejercicio 1

Este es un ejercicio clásico para principiantes en programación, y enseña varios conceptos fundamentales como ser Entrada del usuario, Bucles, Operaciones matemáticas



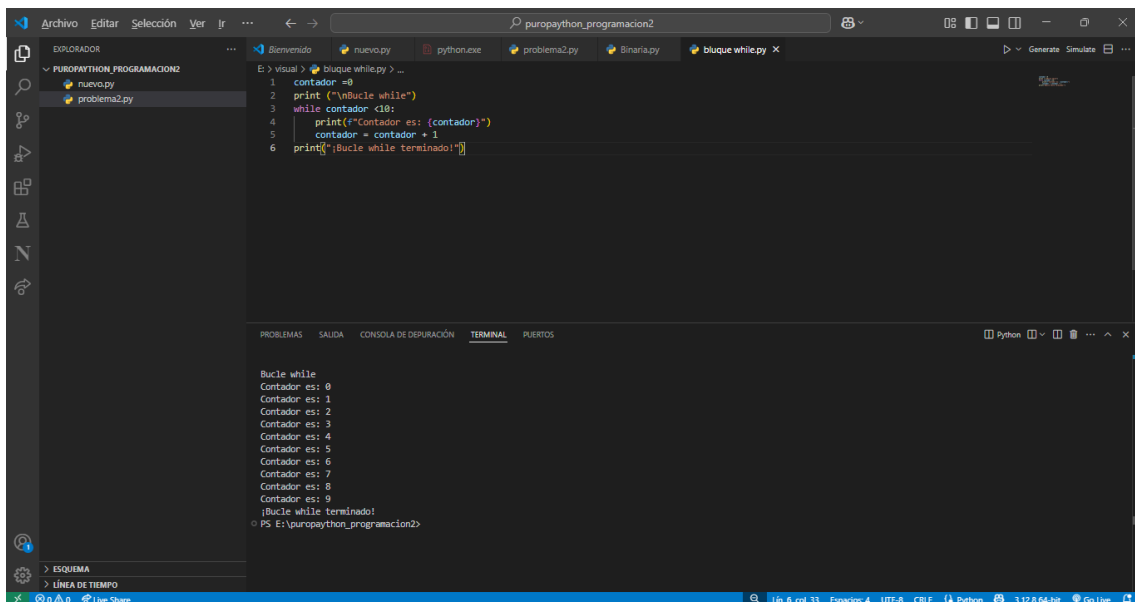
The screenshot shows a Python IDE with a file explorer on the left containing 'nuevo.py' and 'problema2.py'. The main editor displays the code for 'problema2.py', which prompts the user for a number and prints a multiplication table from 1 to 10. The terminal at the bottom shows the execution output, where the user has entered '4', resulting in a 4x10 multiplication table.

```
problema2.py > ...
1 print("ingrese un número para la tabla de multiplicar")
2 numero = int(input("Número: "))
3 for i in range(1, 11):
4     resultado = numero * i
5     print(f"{numero} x {i} = {resultado}")
6 # Ejemplo de uso:
7 # numero = 5
8
```

```
PS E:\puropaython_programacion2> E:\Users\AP\AppData\Local\Programs\Python\Python312\python.exe e:\puropaython_programacion2\problema2.py
ingrese un número para la tabla de multiplicar
Número: 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
PS E:\puropaython_programacion2>
```

## Ejercicio 2

Este ejercicio está diseñado para enseñarte el uso del bucle while, que es otro tipo de estructura de repetición en Python, Variables y control del bucle



The screenshot shows a Python IDE with a file explorer on the left containing 'nuevo.py', 'problema2.py', 'Binaria.py', and 'bloque while.py'. The main editor displays the code for 'bloque while.py', which uses a while loop to print the value of a counter from 0 to 9. The terminal at the bottom shows the execution output, displaying the counter values and a termination message.

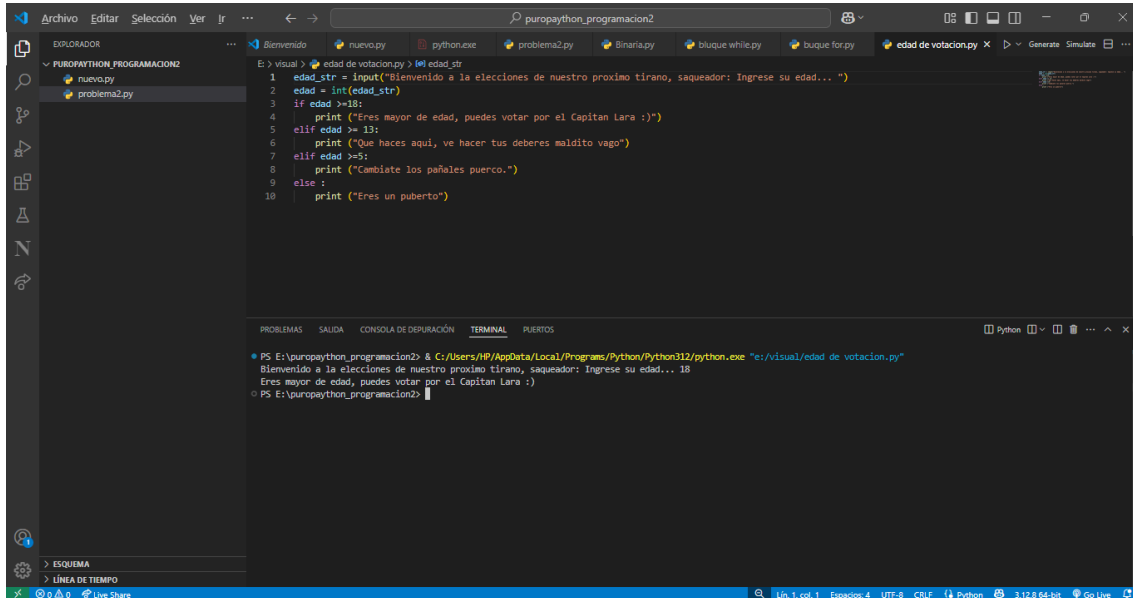
```
E:\visual > E:\visual\bloque while.py > ...
1 contador = 0
2 print("\nBucle while")
3 while contador < 10:
4     print(f"Contador es: {contador}")
5     contador = contador + 1
6 print("\n¡Bucle while terminado!")

```

```
Bucle while
Contador es: 0
Contador es: 1
Contador es: 2
Contador es: 3
Contador es: 4
Contador es: 5
Contador es: 6
Contador es: 7
Contador es: 8
Contador es: 9
¡Bucle while terminado!
PS E:\puropaython_programacion2>
```

### Ejercicio 3

Este ejercicio está diseñado para trabajar con: Entrada del usuario, Condicionales, Rangos en comparaciones



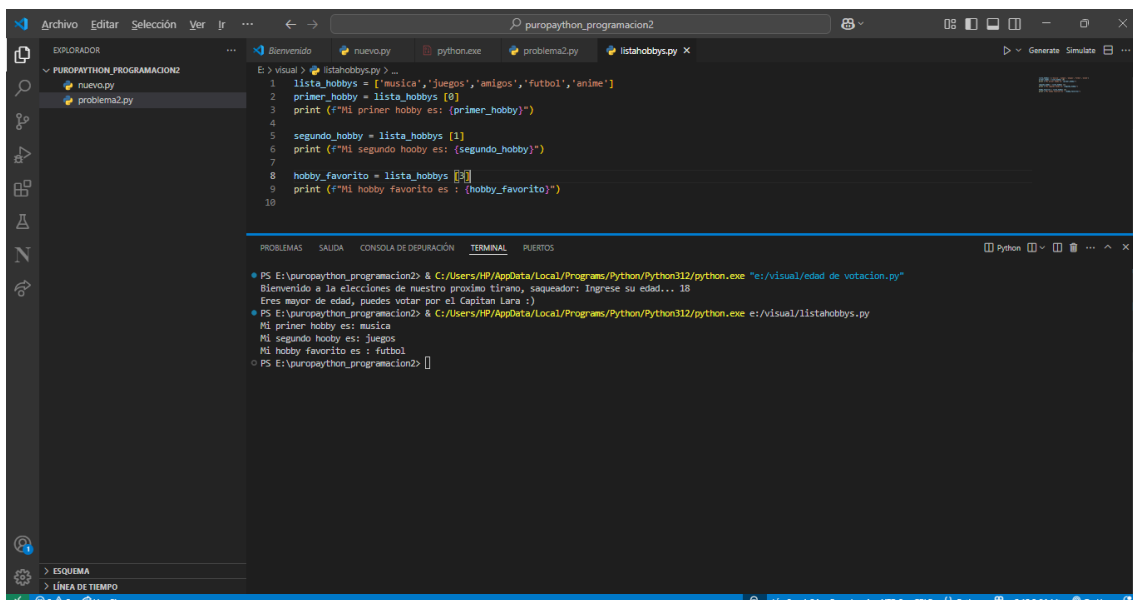
```
E:\visual > cd edad de votacion.py > python.exe
1 edad_str = input("Bienvenido a la elecciones de nuestro proximo tirano, saqueador: Ingrese su edad... ")
2 edad = int(edad_str)
3 if edad >= 18:
4     print ("Eres mayor de edad, puedes votar por el Capitan Lara :)")
5 elif edad >= 13:
6     print ("Que haces aqui, ve hacer tus deberes maldito vago")
7 elif edad >= 5:
8     print ("Cambiate los pañales puerco.")
9 else:
10    print ("Eres un puberto")

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
Python Python 3.12.8 64-bit Go Live

PS E:\puropaython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe "e:/visual/edad de votacion.py"
Bienvenido a la elecciones de nuestro proximo tirano, saqueador: Ingrese su edad... 18
Eres mayor de edad, puedes votar por el Capitan Lara :)
PS E:\puropaython_programacion2>
```

### Ejercicio 4

Este ejercicio está orientado a enseñar el **manejo de listas y acceso a sus elementos**. Aquí están los aprendizajes principales: Cómo crear una lista en Python, Imprimir usando formato, Cómo acceder a elementos usando índices.



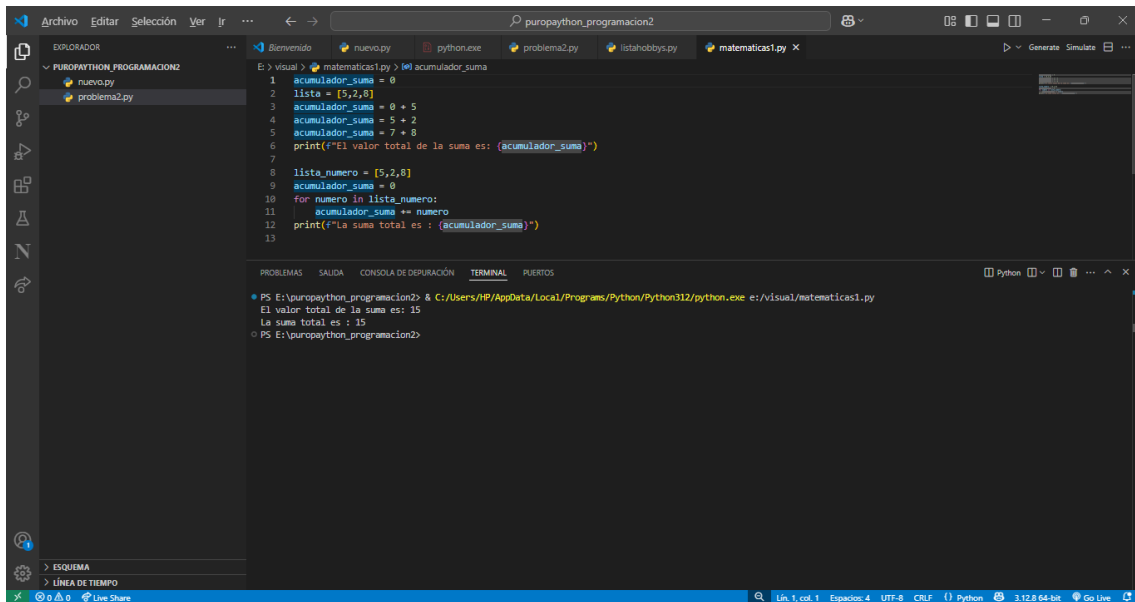
```
E:\visual > cd listahobbies.py > python.exe
1 lista_hobbys = ["musica", "juegos", "amigos", "futbol", "anime"]
2 primer_hobby = lista_hobbys [0]
3 print ("Mi primer hobby es: (primer_hobby)")
4
5 segundo_hobby = lista_hobbys [1]
6 print ("Mi segundo hooby es: (segundo_hobby)")
7
8 hobby_favorito = lista_hobbys [3]
9 print ("Mi hobby favorito es : (hobby_favorito)")
10

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
Python Python 3.12.8 64-bit Go Live

PS E:\puropaython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe "e:/visual/edad de votacion.py"
Bienvenido a la elecciones de nuestro proximo tirano, saqueador: Ingrese su edad... 18
Eres mayor de edad, puedes votar por el Capitan Lara :)
PS E:\puropaython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/listahobbies.py
Mi primer hobby es: musica
Mi segundo hooby es: juegos
Mi hobby favorito es : futbol
PS E:\puropaython_programacion2>
```

## Ejercicio 5

Este ejercicio enseña **cómo recorrer una lista y acumular valores**, una habilidad esencial para procesar datos en programación, Recorrido de una lista con for, Suma acumulativa, **Inicialización del acumulador**

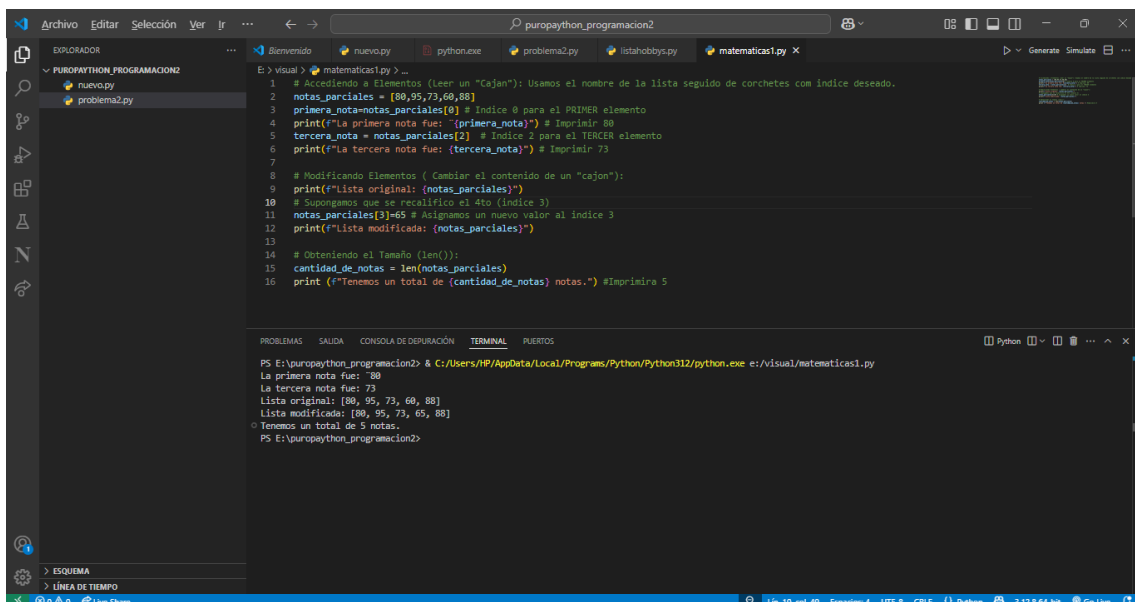


```
E:\visual > python matemáticas1.py
1 acumulador_suma = 0
2 lista = [5,2,8]
3 acumulador_suma = 0 + 5
4 acumulador_suma = 5 + 2
5 acumulador_suma = 7 + 8
6 print("El valor total de la suma es: (acumulador_suma)")
7
8 lista_numero = [5,2,8]
9 acumulador_suma = 0
10 for numero in lista_numero:
11     acumulador_suma += numero
12 print("La suma total es : (acumulador_suma)")
13
```

```
PS E:\puropaython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py
El valor total de la suma es: 15
La suma total es : 15
PS E:\puropaython_programacion2>
```

## Ejercicio 6

Este código abarca varios conceptos clave de listas en Python, que son como "cajas" donde se almacenan datos, Imprimir el contenido de una lista, Contar cuántos elementos hay en la lista



```
E:\visual > python matemáticas1.py
1 # Accediendo a Elementos (Leer un "Cajón"): Usamos el nombre de la lista seguido de corchetes con índice deseado.
2 notas_parciales = [80,95,73,60,88]
3 primera_notas=notas_parciales[0] # Índice 0 para el PRIMER elemento
4 print("La primera nota fue: (primera_notas)") # Imprimir 80
5 tercera_notas = notas_parciales[2] # Índice 2 para el TERCER elemento
6 print("La tercera nota fue: (tercera_notas)") # Imprimir 73
7
8 # Modificando Elementos (Cambiar el contenido de un "cajón"):
9 print("Lista original: (notas_parciales)")
10 # Supongamos que se recalifico el 4to (índice 3)
11 notas_parciales[3]=65 # Asignamos un nuevo valor al índice 3
12 print("Lista modificada: (notas_parciales)")
13
14 # Obteniendo el Tamaño (len()):
15 cantidad_de_notas = len(notas_parciales)
16 print ("Tenemos un total de (cantidad_de_notas) notas.") #Imprimira 5
```

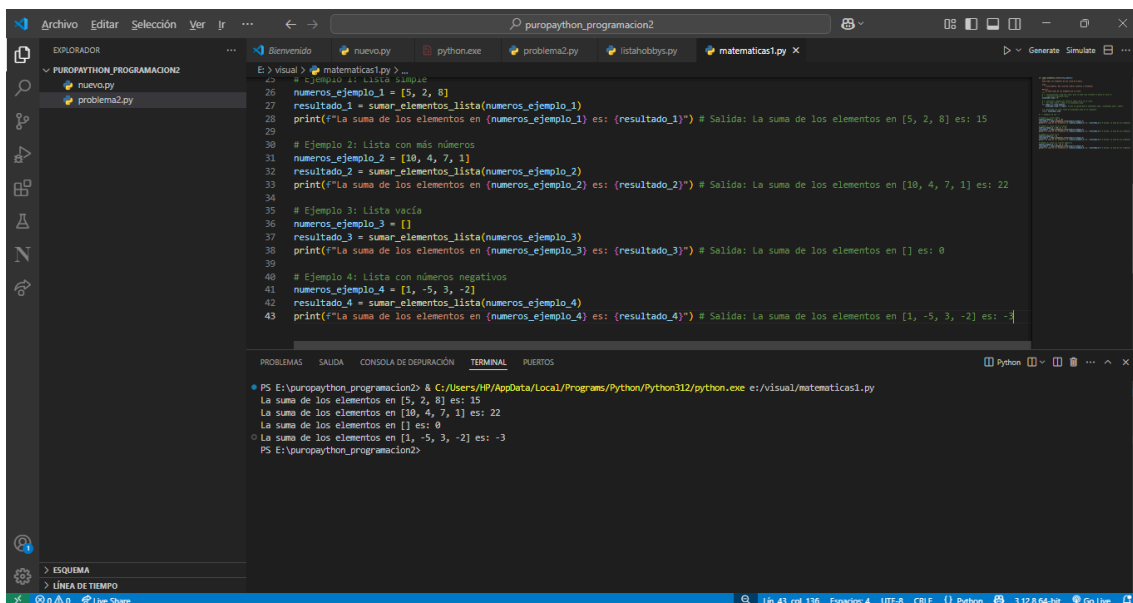
```
PS E:\puropaython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py
La primera nota fue: 80
La tercera nota fue: 73
Lista original: [80, 95, 73, 60, 88]
Lista modificada: [80, 95, 73, 65, 88]
Tenemos un total de 5 notas.
PS E:\puropaython_programacion2>
```

## Ejercicio 7

Las funciones permiten reutilizar código para realizar tareas comunes (como sumar una lista), Variables acumuladoras. Uso de bucle for para recorrer listas,

### Manejo de distintos tipos de listas

- Listas con pocos elementos
- Listas más largas
- Listas vacías
- Listas con números negativos



```
E:\visual > cd matematicas1.py > ...
42 # Ejemplo 1: Lista con pocos elementos
43 numeros_ejemplo_1 = [5, 2, 8]
44 resultado_1 = sumar_elementos_lista(numeros_ejemplo_1)
45 print(f"La suma de los elementos en (numeros_ejemplo_1) es: (resultado_1)") # Salida: La suma de los elementos en [5, 2, 8] es: 15
46
47 # Ejemplo 2: Lista con más números
48 numeros_ejemplo_2 = [10, 4, 7, 1]
49 resultado_2 = sumar_elementos_lista(numeros_ejemplo_2)
50 print(f"La suma de los elementos en (numeros_ejemplo_2) es: (resultado_2)") # Salida: La suma de los elementos en [10, 4, 7, 1] es: 22
51
52 # Ejemplo 3: Lista vacía
53 numeros_ejemplo_3 = []
54 resultado_3 = sumar_elementos_lista(numeros_ejemplo_3)
55 print(f"La suma de los elementos en (numeros_ejemplo_3) es: (resultado_3)") # Salida: La suma de los elementos en [] es: 0
56
57 # Ejemplo 4: Lista con números negativos
58 numeros_ejemplo_4 = [1, -5, 3, -2]
59 resultado_4 = sumar_elementos_lista(numeros_ejemplo_4)
60 print(f"La suma de los elementos en (numeros_ejemplo_4) es: (resultado_4)") # Salida: La suma de los elementos en [1, -5, 3, -2] es: -3
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropaython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py
La suma de los elementos en [5, 2, 8] es: 15
La suma de los elementos en [10, 4, 7, 1] es: 22
La suma de los elementos en [] es: 0
La suma de los elementos en [1, -5, 3, -2] es: -3
PS E:\puropaython_programacion2>
```



The screenshot shows the Visual Studio Code interface with a project named 'puropaython\_programacion2'. The Explorer panel on the left shows the project structure with files 'nuevo.py' and 'problema2.py'. The main editor displays the code in 'problema2.py':

```
1 # numero = 3
2
3 print("Contando hasta 3 (sin incluirlo):")
4 for numero in range(3): # range(3) genera 0, 1, 2
5     print(numero)
6
7 print("\nRecorriendo un string:")
8 nombre = "PYTHON"
9 for letra in nombre:
10     print(letra)
```

The TERMINAL panel at the bottom shows the output of the script:

```
PS E:\puropaython_programacion2> & C:/Users/HP/AppData/Local/Programs/Python/Python312/python.exe e:/puropaython_programacion2/problema2.py
Contando hasta 3 (sin incluirlo):
0
1
2

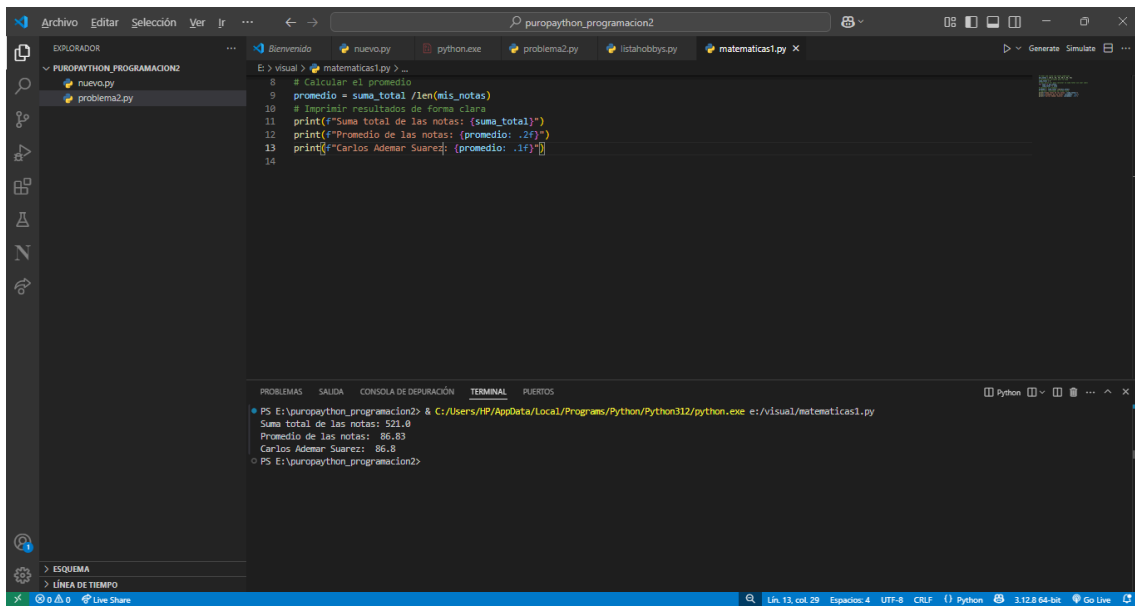
Recorriendo un string:
P
Y
T
H
O
N

PS E:\puropaython_programacion2>
```

The status bar at the bottom indicates the file is 'Line 14, col 17', uses 'Spaces: 4', 'UTF-8' encoding, 'CRLF' line endings, and is a 'Python 3.12.0 64-bit' file.

## Ejercicio 11

1. **Listas:** mis\_notas es una lista de números.
2. **Funciones útiles:**
  - sum(): Suma los elementos de una lista.
  - len(): Cuenta el número de elementos.
3. **Operaciones matemáticas:** Cálculo de promedio (suma / cantidad).
4. print() **correcto:** Uso de comas para separar texto y variables.



```
E:\visual > python.exe matematicas1.py
8 # Calcular el promedio
9 promedio = suma_total / len(mis_notas)
10 # Imprimir resultados de forma clara
11 print(f"Suma total de las notas: {suma_total}")
12 print(f"Promedio de las notas: {promedio:.2f}")
13 print(f"Carlos Ademar Suarez: {promedio:.1f}")
14
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py
Suma total de las notas: 521.0
Promedio de las notas: 86.83
Carlos Ademar Suarez: 86.8
PS E:\puropython_programacion2>
```

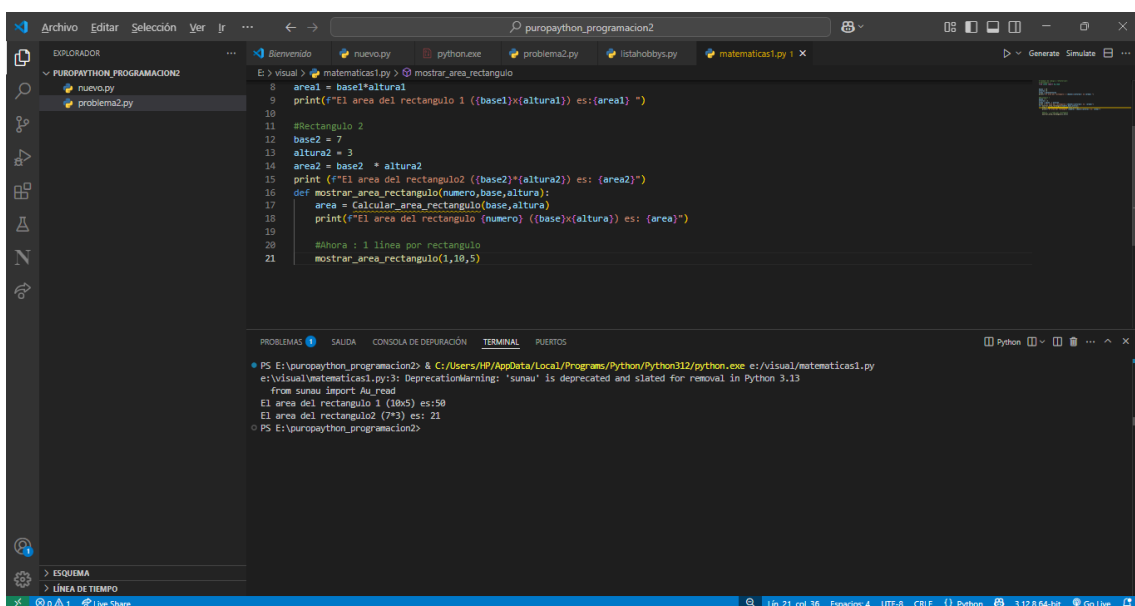
## Ejercicio 12

1. **Fórmula del área de un rectángulo:**  $\text{base} * \text{altura}$ .
2. **Funciones en Python:**

definir una función con `def`.

Paso de parámetros (numero, base, altura).

3. f-strings: Para formatear texto con variables (ej: `f"El área es {area}"`).
4. **Reutilización de código:** La función evita repetir la misma lógica múltiples veces.



```
E:\visual > python.exe matematicas1.py
8 area1 = base1*altura1
9 print(f"El area del rectangulo 1 {(base1)*altura1} es: {area1}")
10
11 #Rectangulo 2
12 base2 = 7
13 altura2 = 3
14 area2 = base2 * altura2
15 print(f"El area del rectangulo2 {(base2)*altura2} es: {area2}")
16 def mostrar_area_rectangulo(numero,base,altura):
17     area = calcular_area_rectangulo(base,altura)
18     print(f"El area del rectangulo {numero} {(base)*altura} es: {area}")
19
20 #Ahora : 1 linea por rectangulo
21 mostrar_area_rectangulo(1,10,5)
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py
e:/visual/matematicas1.py:3: DeprecationWarning: 'sunau' is deprecated and slated for removal in Python 3.13
from sunau import Au_read
El area del rectangulo 1 (10*5) es: 50
El area del rectangulo2 (7*3) es: 21
PS E:\puropython_programacion2>
```



```
1 # clase06_búsquedas.py
2
3 def busqueda_lineal(lista, clave):
4     """
5     Implementa el algoritmo de búsqueda lineal.
6
7     Args:
8         lista (list): La lista en la que buscar (puede estar desordenada).
9         clave: El elemento a buscar.
10
11     Returns:
12         int: El índice de la clave si se encuentra, o -1 si no.
13     """
14     # Usa un bucle for con range(len(lista)) para obtener los índices i.
15     for i in range(len(lista)):
16         # Dentro del bucle, si lista[i] == clave: return i.
17         if lista[i] == clave:
18             return i
19     # Si el bucle termina sin encontrar nada, return -1 (después del bucle).
20     return -1
21
22 # Prueba tu función con assert:
23 if __name__ == "__main__":
24     mi_lista_desordenada = [10, 5, 42, 8, 17, 30, 25]
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

PS E:\puropaython\_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py

Probando busqueda\_lineal...

¡Pruebas para busqueda\_lineal pasaron! ✓

Ejemplos adicionales:

Buscando 8 en [10, 5, 42, 8, 17, 30, 25]: 3

Buscando 17 en [10, 5, 42, 8, 17, 30, 25]: 4

Buscando 100 en [10, 5, 42, 8, 17, 30, 25]: -1

PS E:\puropaython\_programacion2>

```
14 print(letra)*****
15 contador = 0
16 print ("\nBucle while:")
17 while contador < 3:
18     print(f"Contador es: {contador}")
19     contador = contador + 1
20     # ¡MUY IMPORTANTE! Actualizar la variable de control
21     # o se para evitar un bucle infinito. print("Bucle while terminado!")
22     # ¡Cuidado con los Bucles Infinitos en while!
23     # Asegúrate de que la condición eventualmente se vuelva False.
24
25
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

PS E:\puropaython\_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/puropaython\_programacion2/problema2.py

Bucle while:

Contador es: 0

Contador es: 1

Contador es: 2

PS E:\puropaython\_programacion2>

```
8 print("2. General Patán")
9 print("3. Señor de los Memes")
10
11 opcion = input("Ingresa el número de tu candidato favorito: ")
12
13 if opcion == "1":
14     print("Has votado por Capitán Lara. ¡Buena suerte con eso!")
15 elif opcion == "2":
16     print("Has votado por General Patán. Esperemos que sobrevivas.")
17 elif opcion == "3":
18     print("Has votado por el Señor de los Memes. El caos te bendiga.")
19 else:
20     print("Opción no válida. ¡Has desperdiciado tu voto como buen ciudadano!")
21
22 if edad >= 13:
23     print("¿Qué haces aquí? ¡Ve a hacer tus deberes, maldito vago!")
24 elif edad >= 5:
25     print("¡Cámbiate los pañales, puerco!")
26 else:
27     print("Eres un puerbo. Vuelve cuando sepas atarte los zapatos.")
```

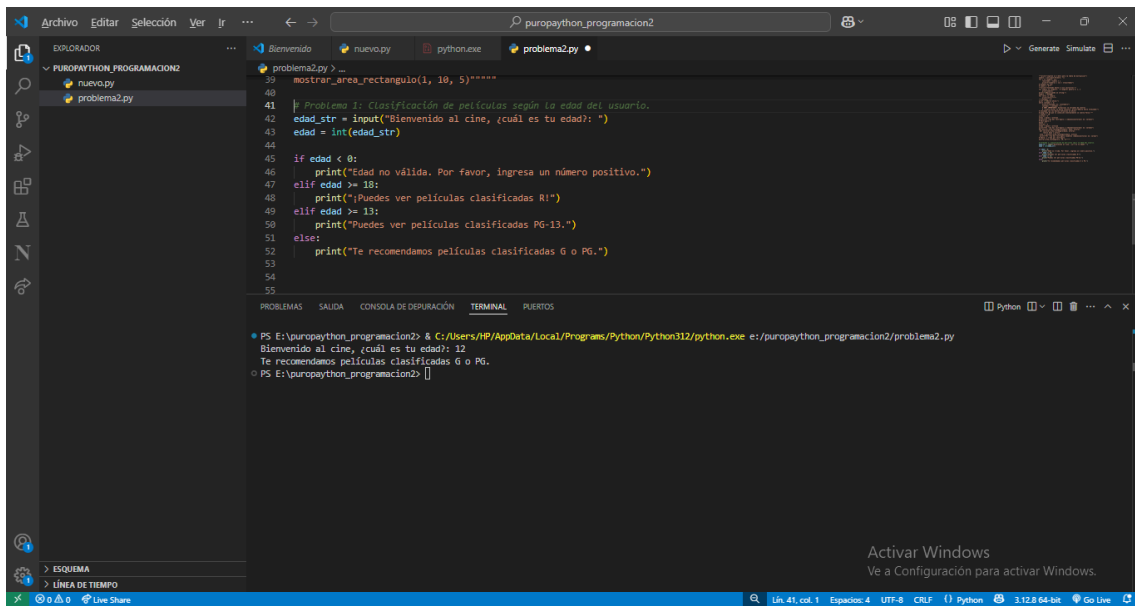
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

PS E:\puropython\_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/visual/matematicas1.py  
Bienvenido a las elecciones de nuestro próximo tirano, saqueador. Ingresa su edad: 18  
Eres mayor de edad, puedes votar por el próximo dictador :D  
Candidatos disponibles:  
1. Capitán Lara  
2. General Patán  
3. Señor de los Memes  
Ingresa el número de tu candidato favorito: 3  
Has votado por el Señor de los Memes. El caos te bendiga.  
PS E:\puropython\_programacion2>

```
24 base1 = 10
25 altura1 = 5
26 area1 = base1 * altura1
27 print("El área del rectángulo 1 ((base1)x(altura1)) es: {area1}")
28 # Rectángulo 2
29 base2 = 7
30 altura2 = 3
31 area2 = base2 * altura2
32 print("El área del rectángulo 2 ((base2)x(altura2)) es: {area2}")
33 def mostrar_area_rectangulo(numero, base, altura):
34     def calcular_area_rectangulo(base, altura):
35         return base * altura
36     area = calcular_area_rectangulo(base, altura)
37     print(f"El área del rectángulo {numero} ((base)x(altura)) es: {area}")
38 # Ahora: 1 línea por rectángulo
39 mostrar_area_rectangulo(1, 10, 5)
40
41
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

PS E:\puropython\_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:/puropython\_programacion2/problema2.py  
El área del rectángulo 1 (10x5) es: 50  
El área del rectángulo 2 (7x3) es: 21  
El área del rectángulo 1 (10x5) es: 50  
PS E:\puropython\_programacion2>



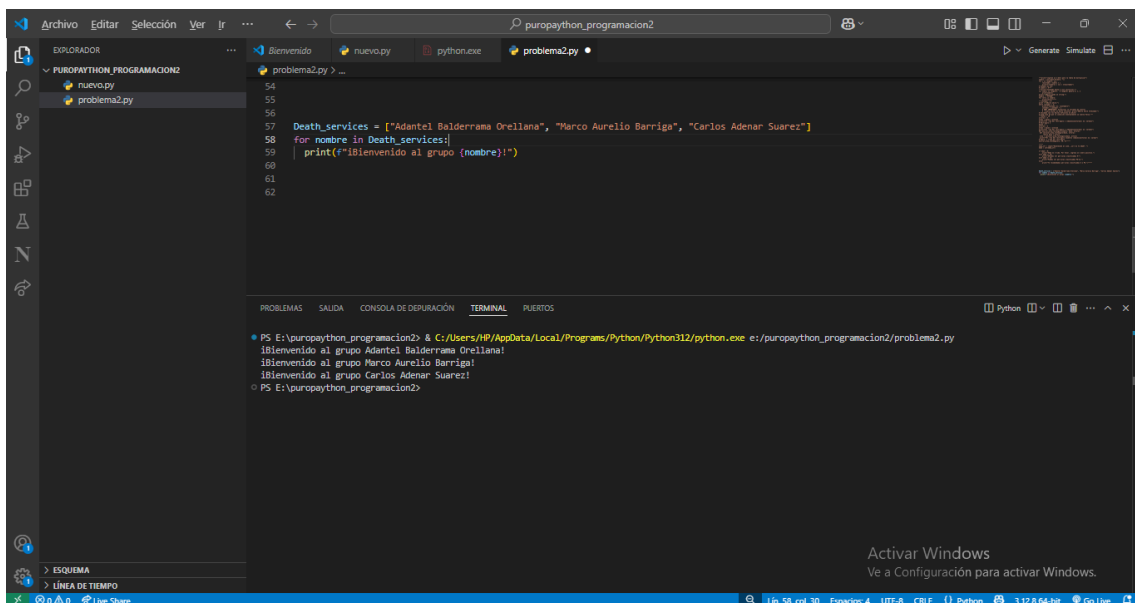
```
39 mostrar_area_rectangulo(1, 10, 5)"""
40
41 # Problema 1: Clasificación de películas según la edad del usuario.
42 edad_str = input("Bienvenido al cine, ¿cuál es tu edad?: ")
43 edad = int(edad_str)
44
45 if edad < 0:
46     print("Edad no válida. Por favor, ingresa un número positivo.")
47 elif edad >= 18:
48     print("Puedes ver películas clasificadas R")
49 elif edad >= 13:
50     print("Puedes ver películas clasificadas PG-13.")
51 else:
52     print("Te recomendamos películas clasificadas G o PG.")
53
54
55
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropaython_programacion2> & C:/Users/HP/AppData/Local/Programs/Python/Python312/python.exe e:/puropaython_programacion2/problema2.py
Bienvenido al cine, ¿cuál es tu edad?: 12
Te recomendamos películas clasificadas G o PG.
PS E:\puropaython_programacion2>
```

## Ejercicio 18

Conceptos aprendidos en este código: Listas (arrays), Formateo de strings con variables usando f"texto {variable}"



```
54
55
56
57 Death_services = ["Adantel Balderrama Orellana", "Marco Aurelio Barriga", "Carlos Adenar Suarez"]
58 for nombre in Death_services:
59     print(f"¡Bienvenido al grupo {nombre}!")
60
61
62
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropaython_programacion2> & C:/Users/HP/AppData/Local/Programs/Python/Python312/python.exe e:/puropaython_programacion2/problema2.py
¡Bienvenido al grupo Adantel Balderrama Orellana!
¡Bienvenido al grupo Marco Aurelio Barriga!
¡Bienvenido al grupo Carlos Adenar Suarez!
PS E:\puropaython_programacion2>
```

The screenshot shows the Visual Studio Code interface with a project named 'puropaython\_programacion2'. The Explorer sidebar on the left shows two files: 'nuevo.py' and 'problema2.py'. The main editor displays the content of 'nuevo.py', which contains a Python script for summing a list. The script includes comments in Spanish explaining the algorithm (La Receta) and a function 'acumulador\_suma' that takes a list and returns the sum. The terminal at the bottom shows the command 'python.exe' being executed, and the output displays the sum of the list [1, 2, 3, 4, 5], which is 15.

```
1 #El Algoritmo (La Receta):
2 #1. Inicialización: Crea una "caja" para la suma (una variable) y ponle el valor 0. Llámosla acumulador_suma.
3 #2. Iteración: Toma el primer número de la lista y súmalo a tu acumulador_suma.
4 #3. Repetición: Toma el siguiente número de la lista y súmalo a tu acumulador_suma.
5 #4. Condición de fin: Repite el paso 3 hasta que hayas procesado todos los números de la lista.
6 #5. Resultado: El valor final en acumulador_suma es tu respuesta.****
7 def acumulador_suma(lista):
8     acumulador_suma = 0
9     for numero in lista:
10         acumulador_suma += numero
11     return acumulador_suma
12 print(acumulador_suma([1,2,3,4,5]))
13
```

TERMINAL

```
PS E:\puropaython_programacion2> C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:\puropaython_programacion2\nuevo.py
69
PS E:\puropaython_programacion2> C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:\puropaython_programacion2\nuevo.py
15
PS E:\puropaython_programacion2>
```

The screenshot shows the Visual Studio Code interface with the same project. The main editor displays the content of 'nuevo.py', which contains a Python script for finding the maximum number in a list. The script includes comments in Spanish explaining the algorithm and a function 'mayor' that takes a list and returns the maximum value. The terminal at the bottom shows the command 'python.exe' being executed, and the output displays the maximum value of the list [23, 45, 24, 76, 57], which is 76.

```
21 #* Si el elemento actual es más grande que mayor_temporal, ¡desecha el valor antiguo y actualiza
22 #mayor_temporal con este nuevo número más grande!
23 #* Si no, no hagas nada y continúa.
24 #4. Condición de fin: Repite los pasos 2 y 3 hasta que hayas revisado todos los elementos.
25 #5. Resultado: El valor final en mayor_temporal es el número más grande de toda la lista.
26 # Pedir al usuario que ingrese los números separados por comas
27 entrada = input("Ingresa varios números separados por comas: ")
28 numeros = entrada.split(",")
29 mayor = int(numeros[0])
30 for n in numeros:
31     numero = int(n)
32     if numero > mayor:
33         mayor = numero
34 print("El número más grande es:", mayor)
35
```

TERMINAL

```
PS E:\puropaython_programacion2> C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:\puropaython_programacion2\nuevo.py
Ingresa varios números separados por comas: 23, 45, 24, 76, 57
El número más grande es: 76
PS E:\puropaython_programacion2>
```

```
new.py > ...
37 #1. Inicialización: Crea un contador y ponle el valor 0.
38 #2. Iteración: Recorre cada elemento de la lista, uno por uno.
39 #3. Comparación: En cada paso, pregunta: "¿Es este elemento igual al que estoy buscando?"
40 # • Si la respuesta es "Sí", incrementa tu contador en 1. • Si la respuesta es "No", no hagas nada.
41 #4. Condición de Fin: Continúa hasta haber revisado todos los elementos.
42 #5. Resultado: El número final en tu contador es la respuesta.
43 # Deberá imprimir la porque hay tres '1' en la lista
44 # Paso 1: Inicialización
45 contador = 0
46 lista = [1, 3, 1, 5, 1, 7, 9]
47 buscar = 1
48 for elemento in lista:
49     if elemento == buscar:
50         contador += 1
51 print(f"El número, buscar, 'aparece', contador, 'veces.'")
52
53
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropaython_programacion2> & C:/Users/HP/AppData/Local/Programs/Python/Python312/python.exe e:/puropaython_programacion2/nuevo.py
El número 1 aparece 3 veces.
PS E:\puropaython_programacion2>
```

Activar Windows  
Ve a Configuración para activar Windows.

```
new.py > ...
54 #Problema 4: Invertir el orden de una lista (creando una lista nueva).
55 #El Algoritmo (La Receta):
56 #1. Inicialización: Crea una nueva lista vacía para el resultado, llámémosla lista_invertida.
57 #2. Iteración (Especial): Recorre la lista original pero ¡comenzando desde el último elemento y yendo hacia el primero!
58 #3. Construcción: En cada paso, toma el elemento actual de la lista original y añádelo al final de tu lista_invertida.
59 #4. Condición de Fin: Continúa hasta que hayas procesado todos los elementos (desde el último hasta el primero).
60 #5. Resultado: lista_invertida contendrá todos los elementos de la original, pero en orden inverso.
61
62 lista_original = [1, 2, 3, 4, 5]
63 print("Lista original:", lista_original)
64 lista_original.reverse()
65 print("Lista invertida:", lista_original)
66
67
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS E:\puropaython_programacion2> & C:/Users/HP/AppData/Local/Programs/Python/Python312/python.exe e:/puropaython_programacion2/nuevo.py
Lista original: [1, 2, 3, 4, 5]
Lista invertida: [5, 4, 3, 2, 1]
PS E:\puropaython_programacion2>
```

Activar Windows  
Ve a Configuración para activar Windows.

```
E:\visual> python Binariapy.py
3 def busqueda_binaria(lista_ordenada, clave):
34     # Si el bucle termina, return -1.
35     return -1
36
37 # Prueba tu función:
38 if __name__ == "__main__":
39     lista_ordenada = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]
40     print("\nProbando busqueda_binaria...")
41
42 # Test cases using assert
43 assert busqueda_binaria(lista_ordenada, 23) == 5, "Test Falló: 23 debería estar en el índice 5"
44 assert busqueda_binaria(lista_ordenada, 91) == 9, "Test Falló: 91 debería estar en el último índice"
45 assert busqueda_binaria(lista_ordenada, 2) == 0, "Test Falló: 2 debería estar en el primer índice"
46 assert busqueda_binaria(lista_ordenada, 3) == -1, "Test Falló: 3 no debería existir"
47 assert busqueda_binaria(lista_ordenada, 100) == -1, "Test Falló: 100 no debería existir (fuera de rango mayor)"
48 assert busqueda_binaria([], 5) == -1, "Test Falló: Búsqueda binaria en lista vacía"
49 assert busqueda_binaria([1], 1) == 0, "Test Falló: Búsqueda binaria con un solo elemento encontrado"
50 assert busqueda_binaria([1], 99) == -1, "Test Falló: Búsqueda binaria con un solo elemento no encontrado"
51
52 print("\nPruebas para busqueda_binaria pasaron! 🎉")
53
54
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

Probando busqueda_binaria...
¡Pruebas para busqueda_binaria pasaron! 🎉

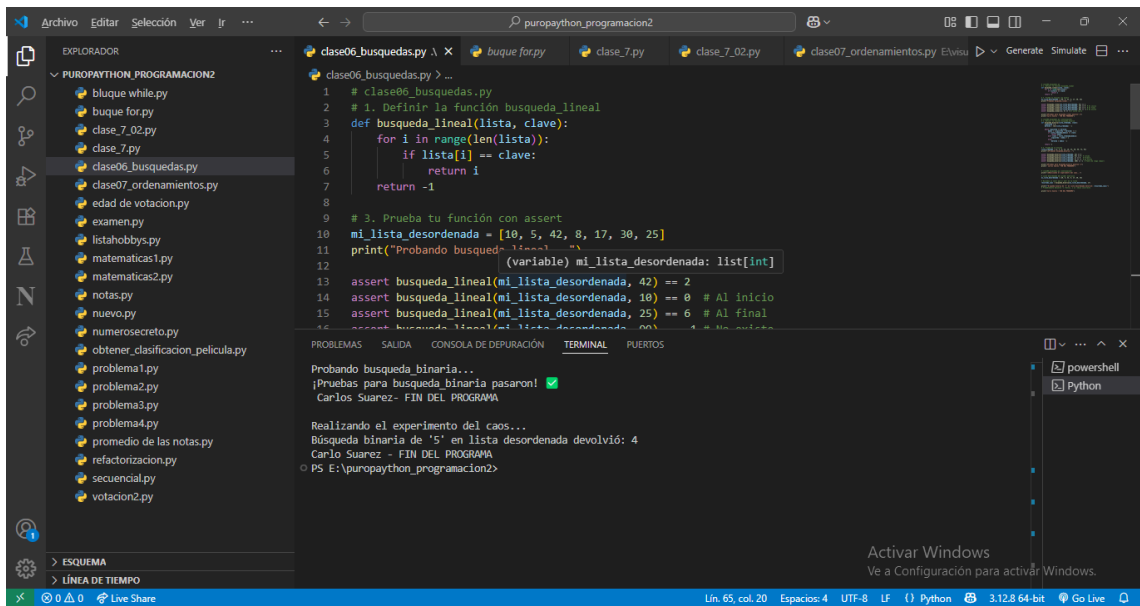
Ejemplos adicionales:
Buscando 12 en [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]: 3
Buscando 5 en [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]: 1
Buscando 70 en [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]: -1
PS E:\puropython_programacion2>
```

```
E:\visual> python matematicas1.py
5 adivinanza = int(input("Adivina el número secreto entre 1 y 10: "))
6
7 # Paso 3: Bucle hasta que lo adivine
8 while adivinanza != numero_secreto:
9     if adivinanza < numero_secreto:
10         print("Demasiado bajo. Intenta de nuevo.")
11     else:
12         print("Demasiado alto. Intenta de nuevo.")
13
14 # Volver a pedir el número
15 adivinanza = int(input("Adivina otra vez: "))
16
17 # Paso 4: Mensaje final
18 print(f"¡Correcto! El número era {numero_secreto}.")

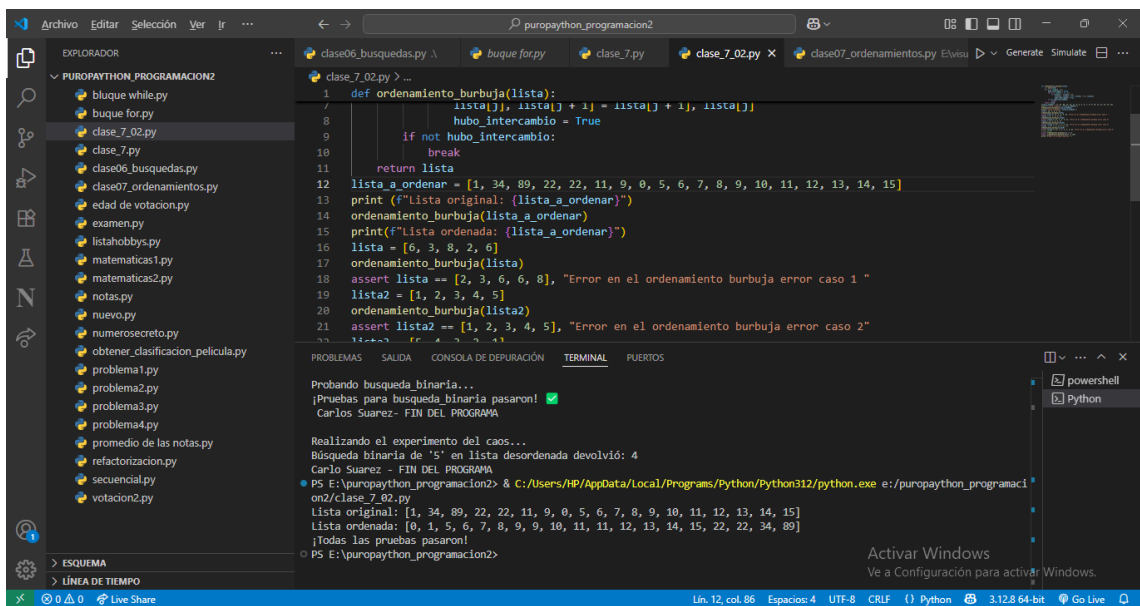
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

PS E:\puropython_programacion2> & C:\Users\HP\AppData\Local\Programs\Python\Python312\python.exe e:\visual\matematicas1.py
Adivina el número secreto entre 1 y 10: 3
Demasiado bajo. Intenta de nuevo.
Adivina otra vez: 5
Demasiado bajo. Intenta de nuevo.
Adivina otra vez: 7
¡Correcto! El número era 7.
PS E:\puropython_programacion2>
```

Ejercicio de la búsqueda binaria y búsqueda lineal



## Ejercicio del ordenamiento burbuja



Aprendí sobre la lista y los repositorios de github, una recomendación que nos enseñé también sobre git que es una herramienta bastante buena para los programadores