

Universidad privada Domingo Savio

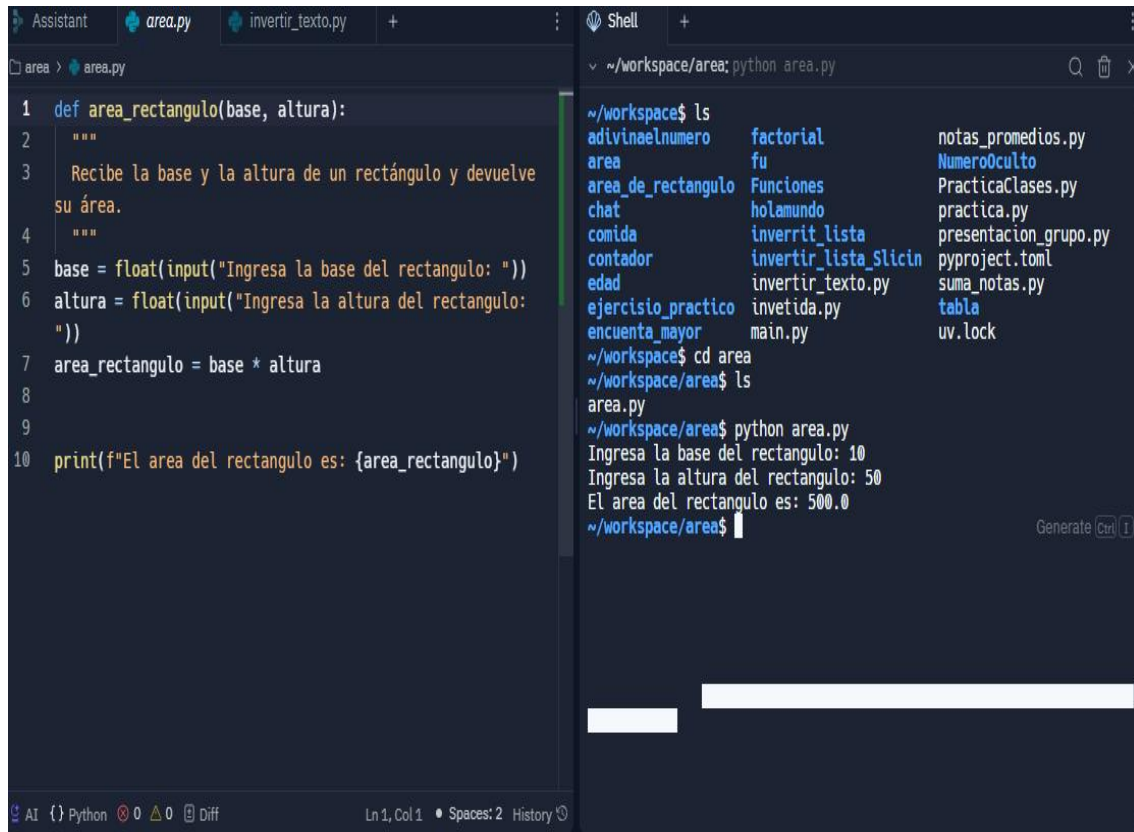


Grupos Los Magios

Materia: Programación II
Docente: Jimmy Nataniel Requena Llorentty
Integrantes del grupo: Limber David Quispe Osco
Beymar Ferrufino Peredo
Daniel Alanes
Yery Torrico Ribera
Salomon Leon Pesoa

Santa Cruz- Bolivia

Código para calcular área de un rectángulo, en este código utilizamos la función “def” para poder definir la base y altura.

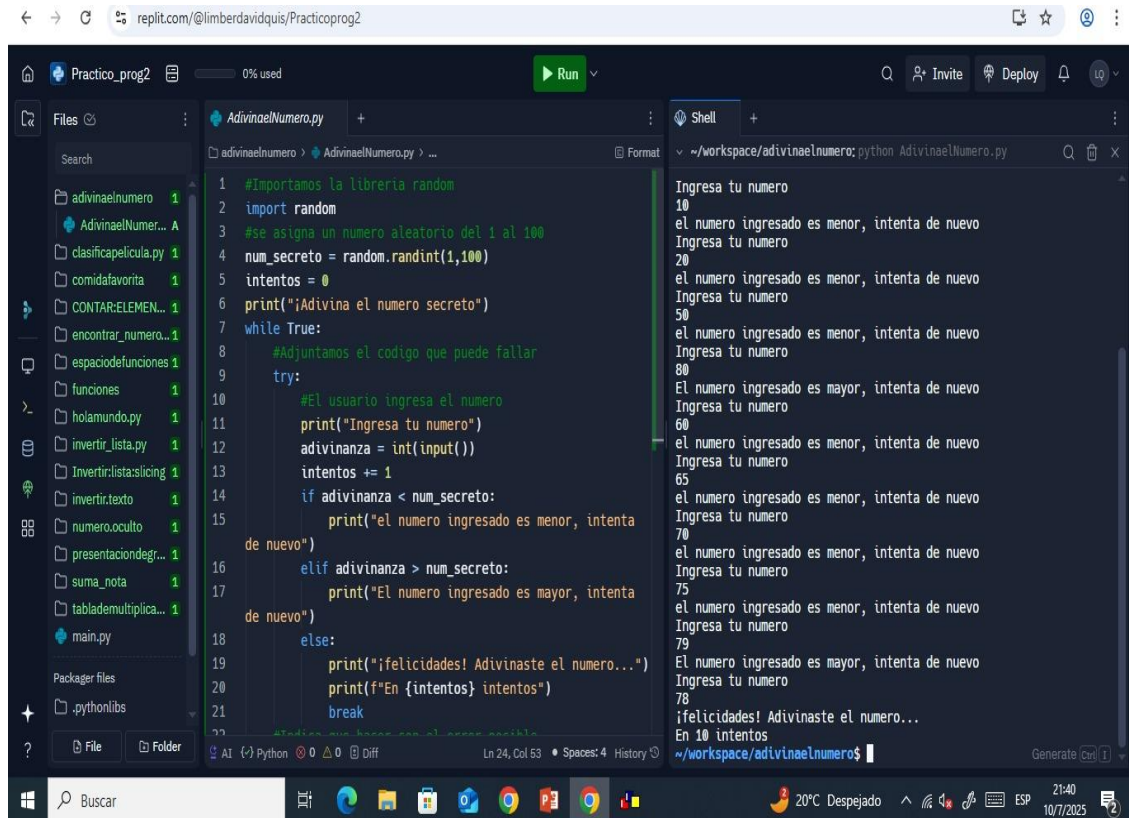


```
1 def area_rectangulo(base, altura):
2     """
3     Recibe la base y la altura de un rectángulo y devuelve
4     su área.
5     """
6     base = float(input("Ingresa la base del rectángulo: "))
7     altura = float(input("Ingresa la altura del rectángulo: "))
8     area_rectangulo = base * altura
9
10    print(f"El area del rectangulo es: {area_rectangulo}")
```

```
~/workspace$ ls
adivinaelnúmero  factorial  notas_promedios.py
area             fu         NumeroOculto
area_de_rectangulo  Funciones  PracticaClases.py
chat             holamundo  practica.py
comida          invertir_lista  presentacion_grupo.py
contador        invertir_lista_Slicing  pyproject.toml
edad           invertir_texto.py  suma_notas.py
ejercicio_practico  invetida.py  tabla
encuesta_mayor    main.py      uv.lock

~/workspace$ cd area
~/workspace/area$ ls
area.py
~/workspace/area$ python area.py
Ingresa la base del rectángulo: 10
Ingresa la altura del rectángulo: 50
El area del rectangulo es: 500.0
~/workspace/area$
```

Código Adivina el número, en esta código importamos una librería de Python la cual es "random" lo cual sirve para que nos dé un numero aleatorio en un rango que nosotros elegimos.



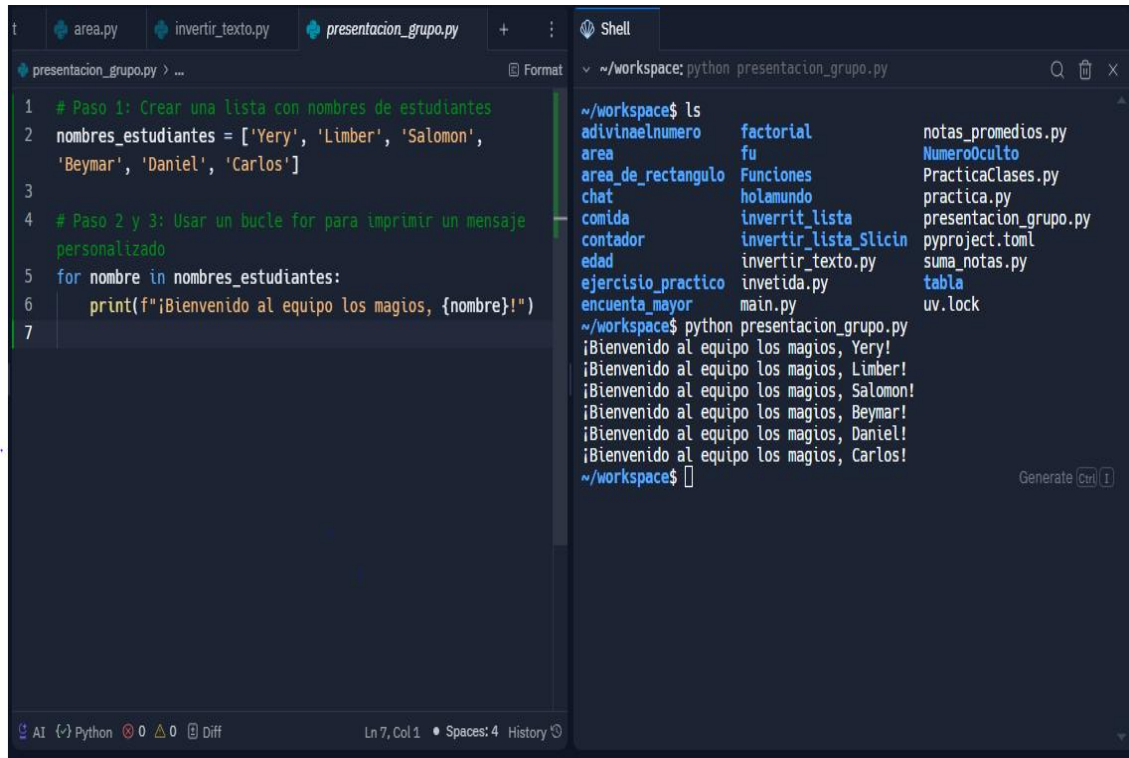
The screenshot displays a Replit IDE interface for a project named "Practico_prog2". The left sidebar shows a file explorer with various Python files, including "AdivinaelNumero.py". The main editor window shows the code for "AdivinaelNumero.py", which is a number guessing game. The code imports the "random" library, generates a random number between 1 and 100, and enters a loop where the user guesses the number. The program provides feedback on whether the guess is too low or too high and counts the number of attempts. The right sidebar shows the output of the program, which displays the user's input and the program's responses.

```
1 #Importamos la libreria random
2 import random
3 #se asigna un numero aleatorio del 1 al 100
4 num_secreto = random.randint(1,100)
5 intentos = 0
6 print("¡Adivina el numero secreto")
7 while True:
8     #Adjuntamos el codigo que puede fallar
9     try:
10         #El usuario ingresa el numero
11         print("Ingresa tu numero")
12         adivinanza = int(input())
13         intentos += 1
14         if adivinanza < num_secreto:
15             print("el numero ingresado es menor, intenta
16 de nuevo")
17         elif adivinanza > num_secreto:
18             print("El numero ingresado es mayor, intenta
19 de nuevo")
20         else:
21             print("¡felicidades! Adivinaste el numero...")
22             print(f"En {intentos} intentos")
23             break
24     except ValueError:
25         print("¡Adios, que bueno que el error sea ahí!")
```

Output:

```
Ingresa tu numero
10
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
20
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
50
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
80
El numero ingresado es mayor, intenta de nuevo
Ingresa tu numero
60
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
65
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
70
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
75
el numero ingresado es menor, intenta de nuevo
Ingresa tu numero
79
El numero ingresado es mayor, intenta de nuevo
Ingresa tu numero
78
¡felicidades! Adivinaste el numero...
En 10 intentos
```

Código presentación de grupo, en este código utilizamos la “listas” y un bucle “for”.



The image shows a code editor with two panels. The left panel displays a Python script named `presentacion_grupo.py`. The script contains the following code:

```
1 # Paso 1: Crear una lista con nombres de estudiantes
2 nombres_estudiantes = ['Yery', 'Limber', 'Salomon',
3 'Beymar', 'Daniel', 'Carlos']
4 # Paso 2 y 3: Usar un bucle for para imprimir un mensaje
5 # personalizado
6 for nombre in nombres_estudiantes:
7     print(f"¡Bienvenido al equipo los magios, {nombre}!")
```

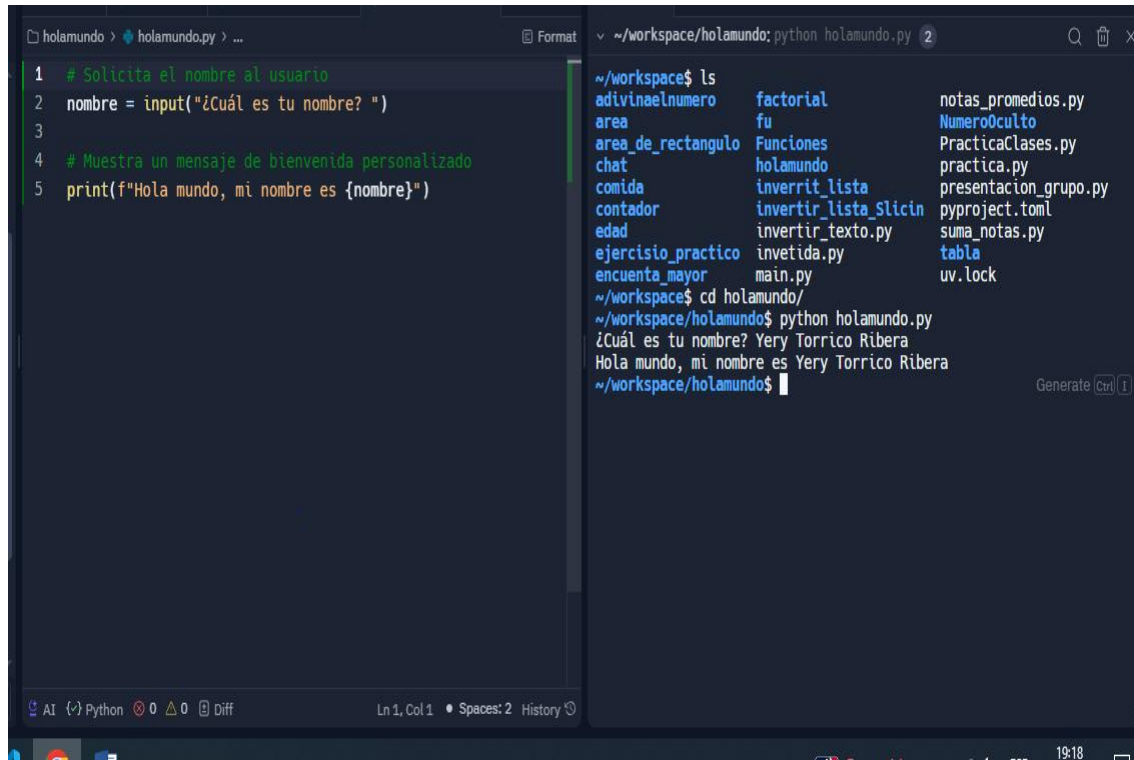
The right panel shows a terminal window with the following commands and output:

```
~/workspace$ ls
adivinaelnumero  factorial  notas_promedios.py
area              fu         NumeroOculto
area_de_rectangulo Funciones  PracticaClases.py
chat             holamundo  practica.py
comida           invertir_lista
contador         invertir_lista_Slicin
edad            invertir_texto.py
ejercicio_practico invetida.py
encuentra_mayor  main.py
~/workspace$ python presentacion_grupo.py
¡Bienvenido al equipo los magios, Yery!
¡Bienvenido al equipo los magios, Limber!
¡Bienvenido al equipo los magios, Salomon!
¡Bienvenido al equipo los magios, Beymar!
¡Bienvenido al equipo los magios, Daniel!
¡Bienvenido al equipo los magios, Carlos!
~/workspace$
```

The status bar at the bottom of the editor indicates the current position is Line 7, Column 1, with 4 spaces and a history icon.

Código Hola mundo con nombre, en este código se utilizó las funciones “input” y la función “print”.

Para poder meter información mediante el teclado usamos la función input y para poder ver usamos la función “print” información que nos da el código.



The image shows a code editor with two panels. The left panel displays a Python script named `holamundo.py` with the following code:

```
1 # Solicita el nombre al usuario
2 nombre = input("¿Cuál es tu nombre? ")
3
4 # Muestra un mensaje de bienvenida personalizado
5 print(f"Hola mundo, mi nombre es {nombre}")
```

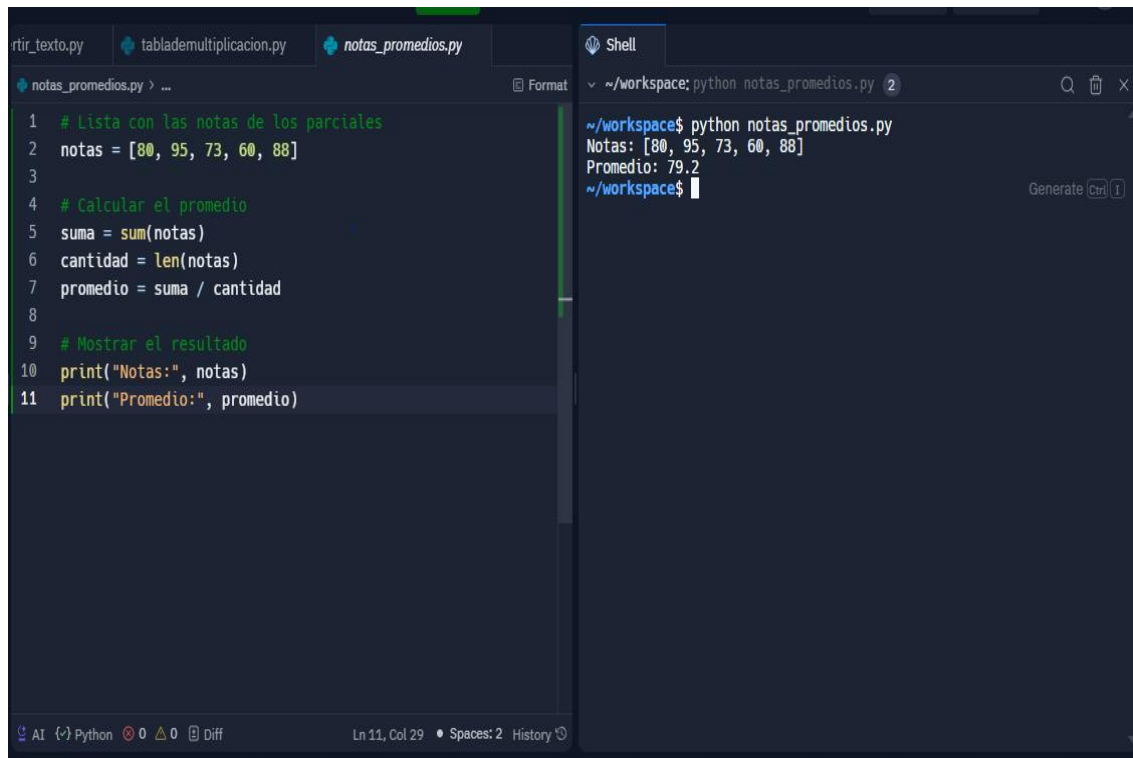
The right panel shows the terminal output of running the script. It lists the files in the workspace, then runs `python holamundo.py`, which prompts for a name. The user enters "Yery Torrico Ribera", and the script outputs "Hola mundo, mi nombre es Yery Torrico Ribera".

```
~/workspace$ ls
adivinaelnumero  factorial      notas_promedios.py
area             fu            NumeroOculto
area_de_rectangulo Funciones     PracticaClases.py
chat             holamundo    practica.py
comida          invertir_lista presentacion_grupo.py
contador        invertir_lista_slicing pyproject.toml
edad           invertir_texto.py suma_notas.py
ejercicio_practico invetida.py   tabla
encuentro_mayor main.py       uv.lock

~/workspace$ cd holamundo/
~/workspace/holamundo$ python holamundo.py
¿Cuál es tu nombre? Yery Torrico Ribera
Hola mundo, mi nombre es Yery Torrico Ribera
~/workspace/holamundo$
```

The status bar at the bottom indicates the current position is Line 1, Column 1, with 2 spaces and a history view available.

Código Notas promedios, en este código utilizamos una lista la cuales son las notas y para poder hacer la función para sacar promedios de dicha notas que estaban en la lista usamos las funciones “sum” y “len” para poder la lista que hicimos.

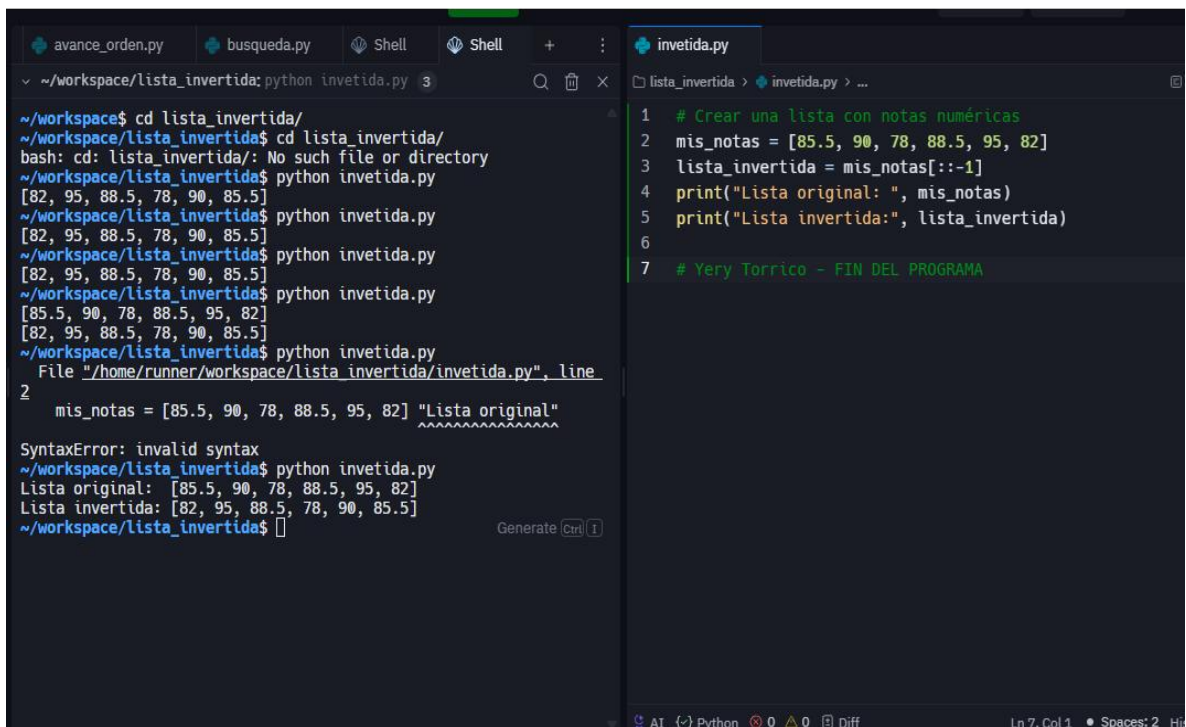


The image shows a code editor with a dark theme. The editor has three tabs at the top: 'rtir_texto.py', 'tablademultiplicacion.py', and 'notas_promedios.py'. The 'notas_promedios.py' tab is active, showing a Python script. The script defines a list 'notas' with values [80, 95, 73, 60, 88], calculates the sum and length, and then prints the list and the average. To the right of the code editor is a 'Shell' terminal window. The terminal shows the command 'python notas_promedios.py' being executed, with the output 'Notas: [80, 95, 73, 60, 88]' and 'Promedio: 79.2'. The status bar at the bottom indicates 'Ln 11, Col 29' and 'Spaces: 2'.

```
1 # Lista con las notas de los parciales
2 notas = [80, 95, 73, 60, 88]
3
4 # Calcular el promedio
5 suma = sum(notas)
6 cantidad = len(notas)
7 promedio = suma / cantidad
8
9 # Mostrar el resultado
10 print("Notas:", notas)
11 print("Promedio:", promedio)
```

```
~/workspace$ python notas_promedios.py
Notas: [80, 95, 73, 60, 88]
Promedio: 79.2
~/workspace$
```

Código Lista invertida, en este código primeramente tenemos crear un lista en este caso numérica lo cual llamamos “mis_notas”, y después usamos Invertir la lista usando “slicing” y después usamos la función “print” para poder mostrar ambas listas.



The image shows a code editor with two panes. The left pane displays the terminal output of running a Python script named `invetida.py`. The right pane shows the source code of `invetida.py`.

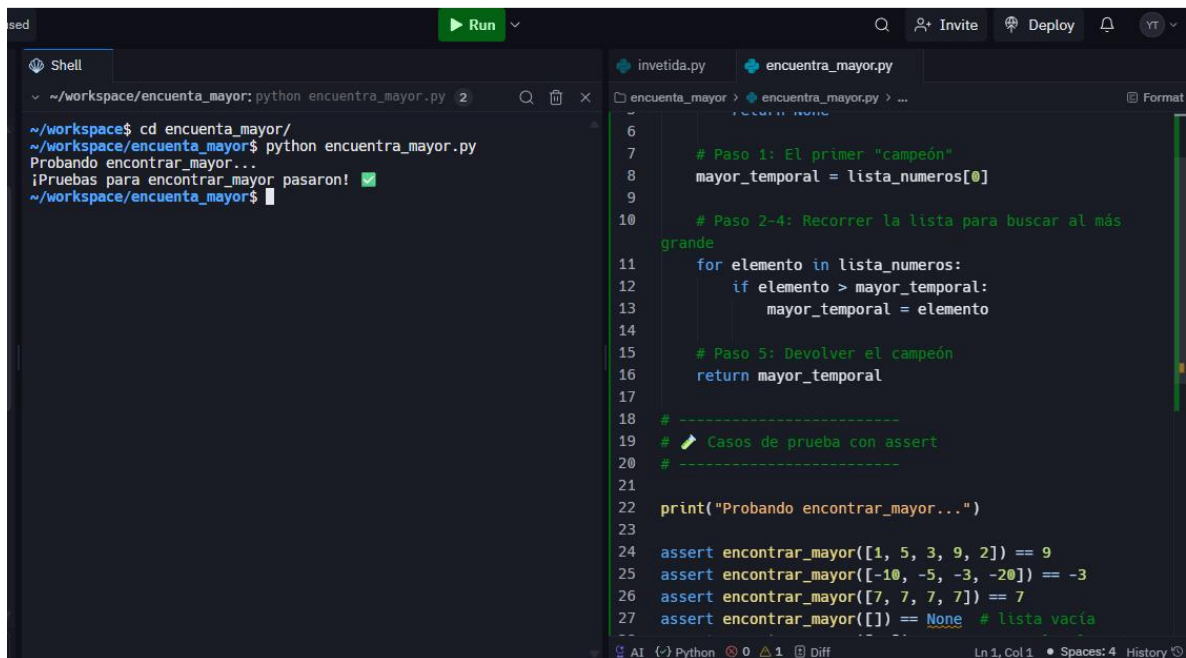
Terminal Output (Left Pane):

```
~/workspace$ cd lista_invertida/
~/workspace/lista_invertida$ cd lista_invertida/
bash: cd: lista_invertida/: No such file or directory
~/workspace/lista_invertida$ python invetida.py
[82, 95, 88.5, 78, 90, 85.5]
~/workspace/lista_invertida$ python invetida.py
[82, 95, 88.5, 78, 90, 85.5]
~/workspace/lista_invertida$ python invetida.py
[82, 95, 88.5, 78, 90, 85.5]
~/workspace/lista_invertida$ python invetida.py
[85.5, 90, 78, 88.5, 95, 82]
~/workspace/lista_invertida$ python invetida.py
[82, 95, 88.5, 78, 90, 85.5]
~/workspace/lista_invertida$ python invetida.py
File "/home/runner/workspace/lista_invertida/invetida.py", line
2
  mis_notas = [85.5, 90, 78, 88.5, 95, 82] "Lista original"
                                         ~~~~~^
SyntaxError: invalid syntax
~/workspace/lista_invertida$ python invetida.py
Lista original: [85.5, 90, 78, 88.5, 95, 82]
Lista invertida: [82, 95, 88.5, 78, 90, 85.5]
~/workspace/lista_invertida$
```

Source Code (Right Pane):

```
1 # Crear una lista con notas numéricas
2 mis_notas = [85.5, 90, 78, 88.5, 95, 82]
3 lista_invertida = mis_notas[::-1]
4 print("Lista original: ", mis_notas)
5 print("Lista invertida:", lista_invertida)
6
7 # Yery Torrico - FIN DEL PROGRAMA
```

Código Encuentra Mayor, la función encontrar_mayor(lista_numeros) recibe una lista de números y devuelve el valor más grandes y usamos “assert” para probar su correcto funcionamiento en distintos casos, como listas vacías o con negativos.

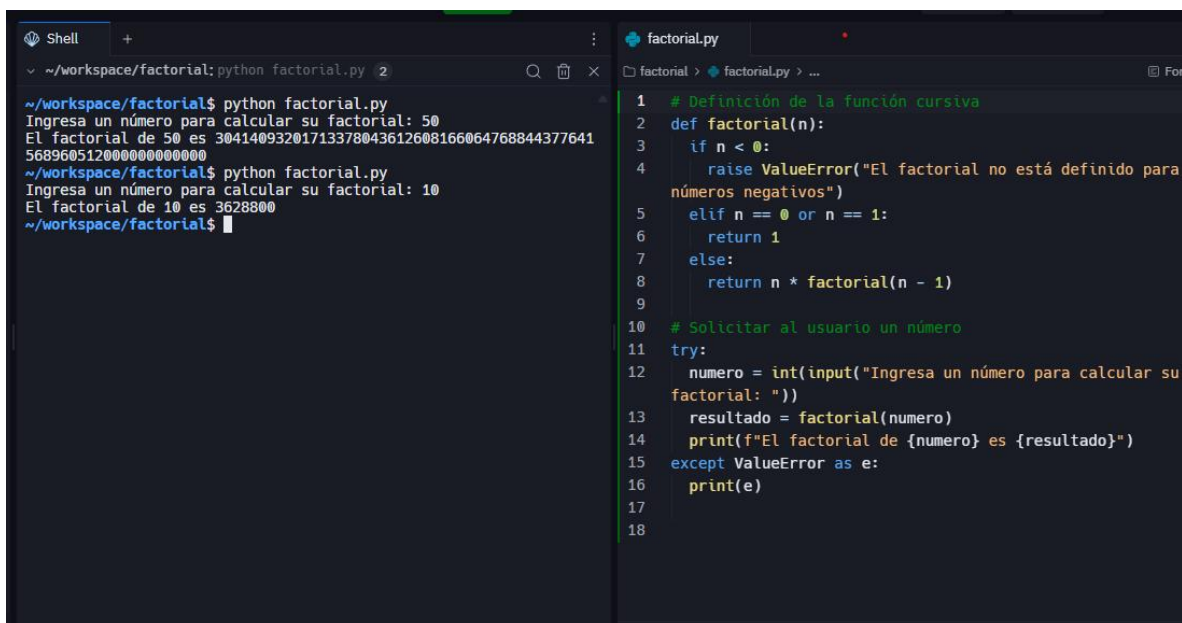


The screenshot shows a code editor with two panels. The left panel displays the terminal output of running a Python script. The right panel shows the source code of the script, which includes a function to find the maximum value in a list and several assert statements for testing.

```
~/workspace$ cd encuesta_mayor/
~/workspace/encuesta_mayor$ python encuesta_mayor.py
Probando encontrar_mayor...
¡Pruebas para encontrar_mayor pasaron! ✓
~/workspace/encuesta_mayor$
```

```
6
7
8 # Paso 1: El primer "campeón"
9 mayor_temporal = lista_numeros[0]
10
11 # Paso 2-4: Recorrer la lista para buscar al más
12 grande
13 for elemento in lista_numeros:
14     if elemento > mayor_temporal:
15         mayor_temporal = elemento
16
17 # Paso 5: Devolver el campeón
18 return mayor_temporal
19
20 # -----
21 # 🟢 Casos de prueba con assert
22 # -----
23
24 print("Probando encontrar_mayor...")
25
26 assert encontrar_mayor([1, 5, 3, 9, 2]) == 9
27 assert encontrar_mayor([-10, -5, -3, -20]) == -3
28 assert encontrar_mayor([7, 7, 7, 7]) == 7
29 assert encontrar_mayor([]) == None # lista vacía
30
```

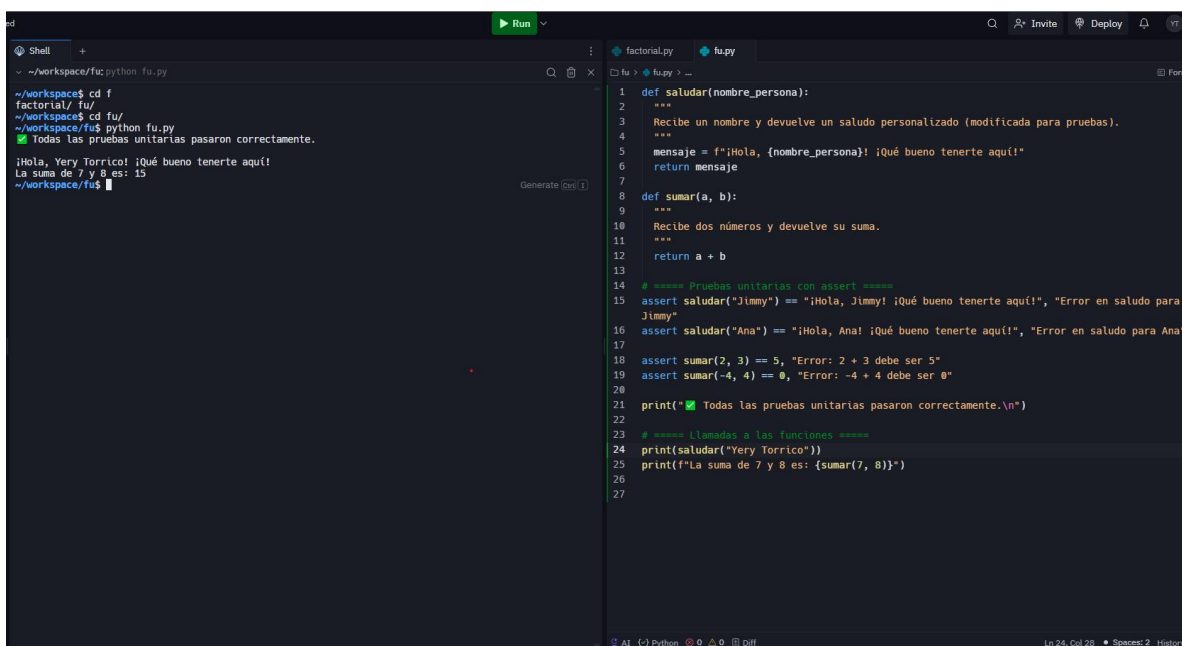

Código Factorial, en este código usamos una función recursiva. Verifica si el número es negativo y lanza un error si lo es. Si es 0 o 1, retorna 1. Para otros casos, multiplica el número por el factorial del anterior. Usa try-except para manejar errores de entrada.



```
~/workspace/factorial:python factorial.py 2
~/workspace/factorial$ python factorial.py
Ingresa un número para calcular su factorial: 50
El factorial de 50 es 30414093201713378043612608166064768844377641
568960512000000000000000
~/workspace/factorial$ python factorial.py
Ingresa un número para calcular su factorial: 10
El factorial de 10 es 3628800
~/workspace/factorial$
```

```
1 # Definición de la función recursiva
2 def factorial(n):
3     if n < 0:
4         raise ValueError("El factorial no está definido para
números negativos")
5     elif n == 0 or n == 1:
6         return 1
7     else:
8         return n * factorial(n - 1)
9
10 # Solicitar al usuario un número
11 try:
12     numero = int(input("Ingresa un número para calcular su
factorial: "))
13     resultado = factorial(numero)
14     print(f"El factorial de {numero} es {resultado}")
15 except ValueError as e:
16     print(e)
17
18
```

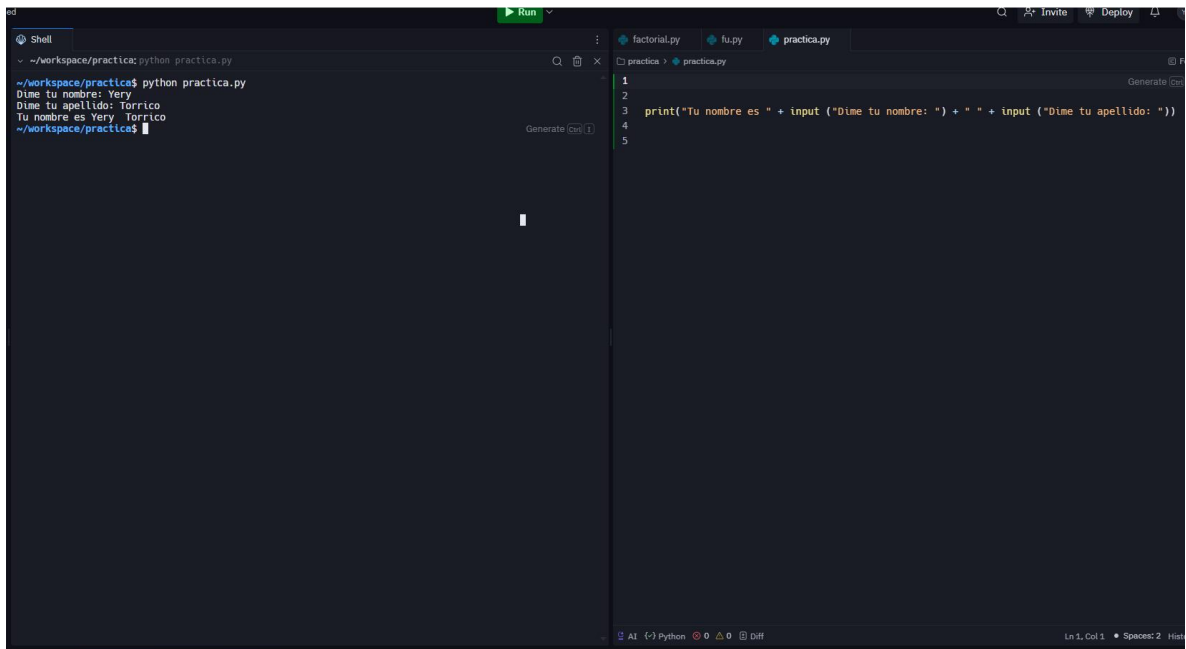
Código Funciones, en este código implementamos los “assert” para poder para verificar si una condición es verdadera.



```
def factorial(fu):  
    """  
    Recibe un número y devuelve su factorial (modificada para pruebas).  
    """  
    mensaje = f"Hola, {nombre_persona}! ¡Qué bueno tenerte aquí!"  
    return mensaje  
  
def sumar(a, b):  
    """  
    Recibe dos números y devuelve su suma.  
    """  
    return a + b  
  
# ===== Pruebas unitarias con assert =====  
assert saludar("Jimmy") == "¡Hola, Jimmy! ¡Qué bueno tenerte aquí!", "Error en saludo para Jimmy"  
assert saludar("Ana") == "¡Hola, Ana! ¡Qué bueno tenerte aquí!", "Error en saludo para Ana"  
  
assert sumar(2, 3) == 5, "Error: 2 + 3 debe ser 5"  
assert sumar(-4, 4) == 0, "Error: -4 + 4 debe ser 0"  
  
print("✅ Todas las pruebas unitarias pasaron correctamente.\n")  
  
# ===== Llamadas a las funciones =====  
print(saludar("Very Torrico"))  
print(f"La suma de 7 y 8 es: {sumar(7, 8)}")
```

~/workspace/fu\$ python fu.py
✅ Todas las pruebas unitarias pasaron correctamente.
¡Hola, Very Torrico! ¡Qué bueno tenerte aquí!
La suma de 7 y 8 es: 15
~/workspace/fu\$

Este es una practica en clases en la cual concatenamos dos “input” dentro de un “print”.



The screenshot shows a code editor with two panels. The left panel is a terminal window showing the execution of a Python script. The right panel shows the source code of the script, `practica.py`.

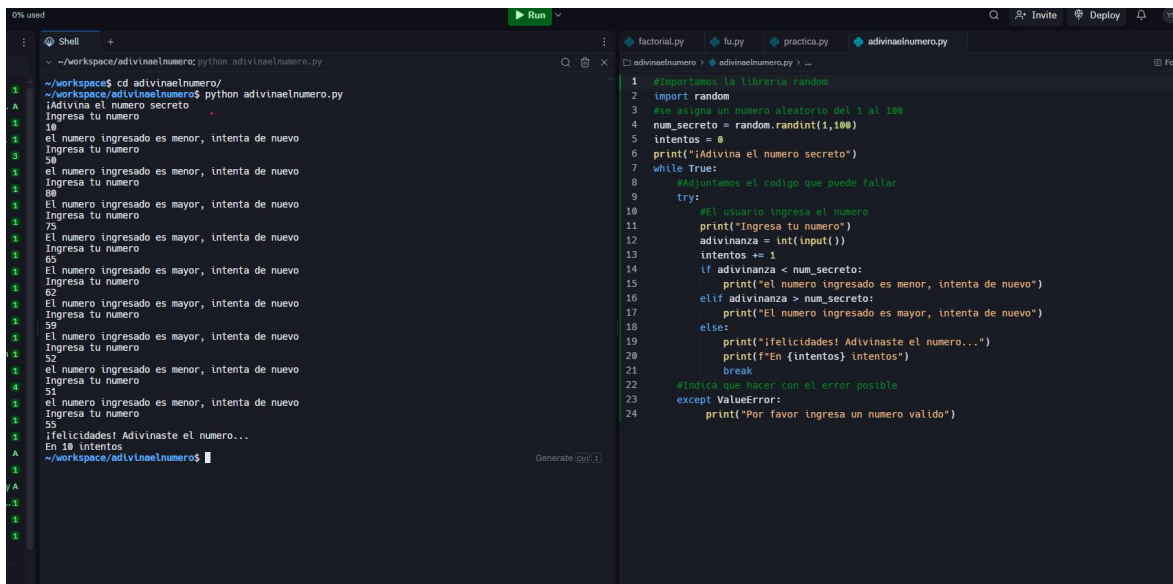
Terminal Output:

```
~/workspace/practica:python practica.py
Dime tu nombre: Yery
Dime tu apellido: Torrico
Tu nombre es Yery Torrico
~/workspace/practica$
```

Source Code (practica.py):

```
1
2
3 print("Tu nombre es " + input("Dime tu nombre: ") + " " + input("Dime tu apellido: "))
4
5
```

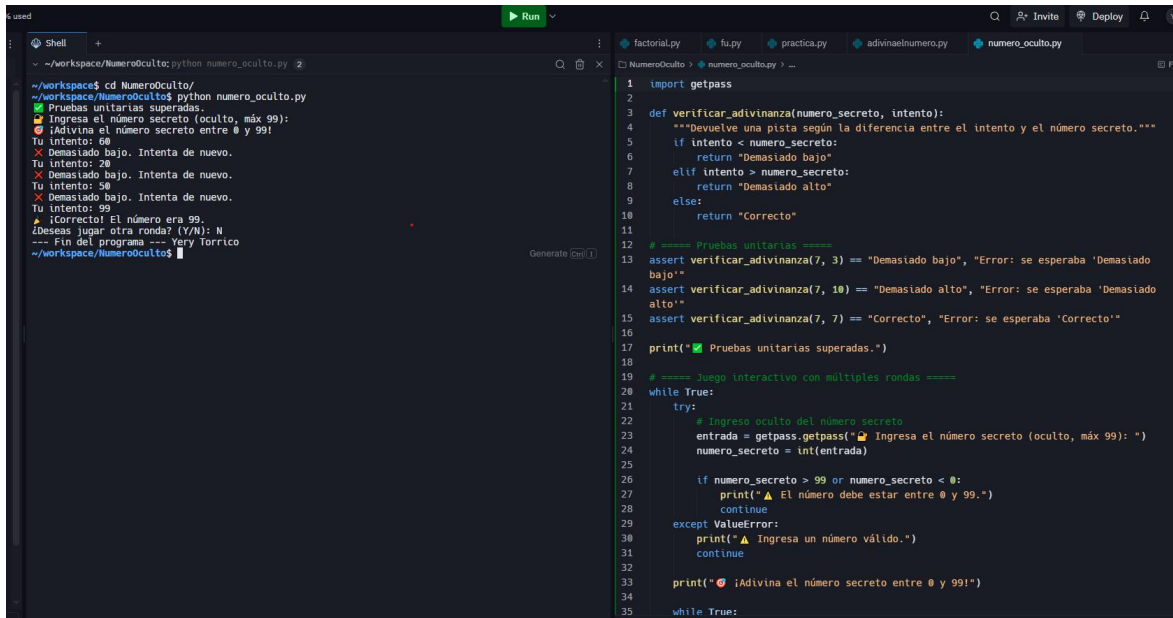
Código Adivina el Numero, para este cogidos importamos la librería “random” lo cual su función es darnos un numero aleatorio de acuerdo al rango que queramos y también usamos un bucle “while” para poder intentar adivinar el numero y ya cuando se adivina el numero termina el programa.



```
0% used
Run
~/workspace/adivinaelnúmero: python adivinaelnúmero.py
~/workspace/adivinaelnúmero$ python adivinaelnúmero.py
Adivina el número secreto
Ingresa tu número
10
El número ingresado es menor, intenta de nuevo
Ingresa tu número
50
El número ingresado es menor, intenta de nuevo
Ingresa tu número
80
El número ingresado es mayor, intenta de nuevo
Ingresa tu número
75
El número ingresado es mayor, intenta de nuevo
Ingresa tu número
65
El número ingresado es mayor, intenta de nuevo
Ingresa tu número
62
El número ingresado es mayor, intenta de nuevo
Ingresa tu número
59
El número ingresado es mayor, intenta de nuevo
Ingresa tu número
52
El número ingresado es menor, intenta de nuevo
Ingresa tu número
51
El número ingresado es menor, intenta de nuevo
Ingresa tu número
59
¡Felicidades! Adivinaste el número...
En 10 intentos
~/workspace/adivinaelnúmeros
```

```
1 #Importamos la libreria random
2 import random
3 #se asigna un numero aleatorio del 1 al 100
4 num_secreto = random.randint(1,100)
5 intentos = 0
6 print("Adivina el numero secreto")
7 while True:
8     #Adjuntamos el codigo que puede fallar
9     try:
10         #El usuario ingresa el numero
11         print("Ingresa tu numero")
12         adivinanza = int(input())
13         intentos += 1
14         if adivinanza < num_secreto:
15             print("el numero ingresado es menor, intenta de nuevo")
16         elif adivinanza > num_secreto:
17             print("El numero ingresado es mayor, intenta de nuevo")
18         else:
19             print("¡felicidades! Adivinaste el numero...")
20             print(f"En {intentos} intentos")
21             break
22     #Indica que hacer con el error posible
23 except ValueError:
24     print("Por favor ingresa un numero valido")
```

Código Numero Oculto, este código es similar al código “adivina el numero” pero en este código implementamos la opción de que ya no sea aleatorio el numero y que uno asigne un numero cualquiera dentro del rango y que otro adivine el numero. En este código importamos la librería “getpass” la cual nos va a permitir ocultar el numero a adivinar



```
~/workspace/NumeroOculto: python numero_oculto.py
~/workspace/NumeroOculto$ python numero_oculto.py
✓ Pruebas unitarias superadas.
🎯 Ingresa el número secreto (oculto, máx 99):
🎯 ¡Adivina el número secreto entre 0 y 99!
Tu intento: 60
✗ Demasiado bajo. Intenta de nuevo.
Tu intento: 20
✗ Demasiado bajo. Intenta de nuevo.
Tu intento: 50
✗ Demasiado bajo. Intenta de nuevo.
Tu intento: 99
🎉 ¡Correcto! El número era 99.
¿Deseas jugar otra ronda? (Y/N): N
--- Fin del programa --- Yury Torrico
~/workspace/NumeroOculto$

1 import getpass
2
3 def verificar_adivinanza(numero_secreto, intento):
4     """Devuelve una pista según la diferencia entre el intento y el número secreto."""
5     if intento < numero_secreto:
6         return "Demasiado bajo"
7     elif intento > numero_secreto:
8         return "Demasiado alto"
9     else:
10        return "Correcto"
11
12 # ===== Pruebas unitarias =====
13 assert verificar_adivinanza(7, 3) == "Demasiado bajo", "Error: se esperaba 'Demasiado bajo'"
14 assert verificar_adivinanza(7, 10) == "Demasiado alto", "Error: se esperaba 'Demasiado alto'"
15 assert verificar_adivinanza(7, 7) == "Correcto", "Error: se esperaba 'Correcto'"
16
17 print("✓ Pruebas unitarias superadas.")
18
19 # ===== Juego interactivo con múltiples rondas =====
20 while True:
21     try:
22         # Ingreso oculto del número secreto
23         entrada = getpass.getpass("🎯 Ingresa el número secreto (oculto, máx 99): ")
24         numero_secreto = int(entrada)
25
26         if numero_secreto > 99 or numero_secreto < 0:
27             print("⚠ El número debe estar entre 0 y 99.")
28             continue
29     except ValueError:
30         print("⚠ Ingresa un número válido.")
31         continue
32
33 print("🎯 ¡Adivina el número secreto entre 0 y 99!")
34
35 while True:
```