

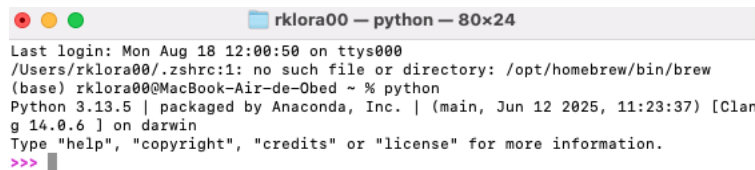
## Tarea 01

18 de Agosto del 2025

Alumno: Lora Marín Obed Ricardo

Profesor: santosg572@gmail.com

Primeramente abrimos la terminal y usamos el lenguaje de programación python.



```
Last login: Mon Aug 18 12:00:50 on ttys000
/Users/rklora00/.zshrc:1: no such file or directory: /opt/homebrew/bin/brew
(base) rklora00@MacBook-Air-de-Obed ~ % python
Python 3.13.5 | packaged by Anaconda, Inc. | (main, Jun 12 2025, 11:23:37) [Clan
g 14.0.6 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

**Ejercicio 1.** Escriba los pasos que hay que seguir para resolver una ecuación de segundo grado:  $ax^2 + bx + c = 0$ , donde  $a, b, c \in \mathbb{R}$

```
>>> import math
... import random
...
... # Datos
... a = 1
... b = -3
... c = 2
...
... # Fórmula general
... discriminante = b**2 - 4*a*c
... raiz_discriminante = math.sqrt(discriminante)
...
... x1 = (-b + raiz_discriminante) / (2 * a)
... x2 = (-b - raiz_discriminante) / (2 * a)
...
... print("Raíces:", x1, x2)
...
Raíces: 2.0 1.0
```

**Ejercicio 2.** Dada la función  $f(x) = 2 - (x - 2)^2$  definida en el intervalo  $[1, 3]$ .

Escriba los pasos necesarios para encontrar el área bajo la curva que define la función y el eje x.

```

>>> # Área bajo f(x) = 2 - (x - 2)^2 entre x = 1 y x = 3
...
... # Evaluamos manualmente
... x0 = 1
... x1 = 2
... x2 = 3
...
... fx0 = 2 - (x0 - 2)**2
... fx1 = 2 - (x1 - 2)**2
... fx2 = 2 - (x2 - 2)**2
...
... # Paso h (distancia entre puntos)
... h = (x2 - x0) / 2
...
... # Aproximación con regla del trapecio compuesta de 3 puntos (Simpson 1/3)
... area = (h / 3) * (fx0 + 4 * fx1 + fx2)
...
... print("Área bajo la curva:", area)
...
Área bajo la curva: 3.333333333333333

```

### Ejercicio 3. Convertir 30° a radianes.

```

>>> grados = 30
... radianes = math.radians(grados)
... print("30 grados =", radianes, "radianes")
...
30 grados = 0.5235987755982988 radianes

```

### Ejercicio 4. Una aproximación para calcular el valor de $e^x$ es utilizar la siguiente expresion:

$$1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!}$$

```

>>> x = 3
... exp_aprox = 1 + x + x**2/math.factorial(2) + x**3/math.factorial(3) + x**4/\
math.factorial(4) + x**5/math.factorial(5)
... print("e^3 aproximado:", exp_aprox)
...
... x = 1
... exp_aprox = 1 + x + x**2/math.factorial(2) + x**3/math.factorial(3) + x**4/\
math.factorial(4) + x**5/math.factorial(5)
... print("e^1 aproximado:", exp_aprox)
...
[e^3 aproximado: 18.4
e^1 aproximado: 2.7166666666666663

```

### Ejercicio 5. Simular el lanzamiento de una moneda, aguilá o sol, { A, S }.

```

>>> import random
...
... # Simular número aleatorio entre 0 y 1
... probabilidad = random.random()
... print("Número aleatorio generado:", probabilidad)
...
... # Si es menor que 0.5 será Águila, si no, Sol
... resultado = "A" * (probabilidad < 0.5) + "S" * (probabilidad >= 0.5)
...
... print("Resultado del lanzamiento:", resultado)
...
Número aleatorio generado: 0.6810946159902671
Resultado del lanzamiento: S
>>>
>>> # Simular número aleatorio entre 0 y 1
... probabilidad = random.random()
... print("Número aleatorio generado:", probabilidad)
...
... # Si es menor que 0.5 será Águila, si no, Sol
... resultado = "Aguila" * (probabilidad < 0.5) + "Sol" * (probabilidad >= 0.5)
... print("Resultado del lanzamiento:", resultado)
...
Número aleatorio generado: 0.4326910123392611
Resultado del lanzamiento: Aguila

```

NOTA1: Para resolver los dos ejercicios últimos utilice los módulos, math y random.

NOTA2: Buscar y tener una prueba de lateralidad, para implementarlo en Python.