Física Computacional
Tarea 1

Pedro Porras Flores
Efraín Ossmar Díaz Pérez

27 de agosto de 2025

Instrucciones

Resuelva los siguientes ejercicios implementando soluciones en Python. Utilice variables adecuadas, estructuras de control de flujo y funciones. Incluya comentarios en su código y muestre los resultados obtenidos.

1. La dinámica de un cometa está gobernada por la fuerza gravitacional entre el cometa y el Sol, $\mathbf{f} = -G\frac{Mm}{r^3}\mathbf{r}$, donde $G = 6{,}67 \times 10^{-11}\text{N}\cdot\text{m}^2/\text{kg}^2$ es la constante gravitacional, $M = 1{,}99 \times 10^{30}\text{kg}$ es la masa del Sol, m es la masa del cometa, $\mathbf{r}$ es el vector de posición del cometa medido desde el Sol, y r es la magnitud de $\mathbf{r}$. Escriba un programa para estudiar el movimiento del cometa Halley, que tiene una distancia de afelio (punto más alejado del Sol) de $5{,}28 \times 10^{12}\text{m}$ y una velocidad en el afelio de $9{,}12 \times 10^{2}\text{m/s}$.

a) ¿Cuáles son las elecciones apropiadas de unidades de tiempo y longitud para este problema?

b) Escriba un programa en Python que convierta unidades del Sistema Internacional a unidades astronómicas apropiadas para simulaciones orbitales, realice lo siguiente:

- Solicite al usuario una distancia en metros usando la función `input()`.
- Convierta el valor ingresado a un número de punto flotante usando `float()`.
- Convierta esta distancia a Unidades Astronómicas (UA) usando la relación: `UA = metros / 1.496e11` (donde 1 UA = 1.496×10^{11} m).
- Para velocidades, solicite al usuario un valor en m/s y conviértalo a UA/año usando: `UA_año = m_s * (365.25 * 24 * 3600) / 1.496e11`.
- Implemente esta conversión en una función.
- Verifique que los valores convertidos sean adecuados para la simulación del cometa Halley.

c) Discuta el error generado por el programa en cada período del cometa Halley.

2. Use la función `isPrime()` importándola desde `misFunciones.py` para realizar lo siguiente:

a) Una función llamada `nPrimes()` que reciba un entero positivo n y que genere los primeros n primos.

b) Una función `twinPrimes` Recibe un entero positivo n y devuelve una lista de tuplas con todos los pares de primos gemelos menores o iguales a n. Se consideran primos gemelos aquellos pares de números primos que difieren en 2 unidades (como 3 y 5, o 11 y 13).

c) Función `descomposition()` que recibe un entero positivo n y devuelve una cadena con la descomposición en factores primos con potencias. Por ejemplo, para $n = 60$ devolvería `"`$2^2 * 3^1 * 5^1$`"`.

3. Escriba una función `nameFile()` debe recibir como primer parámetro una cadena que represente el nombre de una simulación (por ejemplo, `"Pendulum"`). Adicionalmente, podrá aceptar argumentos posicionales (`*args`) o argumentos de palabra clave (`**kwargs`) que representen condiciones iniciales o parámetros del sistema.

- Los valores numéricos deben ser convertidos reemplazando el punto decimal por un guion bajo (por ejemplo, `1.2 → "1_2"`)
- El nombre del archivo resultante debe seguir el formato: `"Nombre-Parametro1-Parametro2-...-ParametroN.dat"`

Ejemplos de uso:

```
nameFile("Pendulum", [1.2, 10.5])
# Retorna: "Pendulum-conditions-1_2-10_5.dat"

nameFile("Oscillator", [1.2, 10.5] ,amplitude=2.5, frequency=1.8)
# Retorna: "Oscillator-conditions-1_2-10_5-amplitude-2_5-frequency-1_8
   .dat"
```

La función debe ser capaz de manejar cualquier combinación de argumentos posicionales y nombrados, formateando consistentemente los valores numéricos según la convención establecida.

4. Escriba funciones que calcule las funciones $\sin(x)$, $\cos(x)$, $\tan(x)$, $log(x)$ y utilizando sus series de Taylor. El cálculo debe detenerse cuando el último término de la serie sea menor que la precisión de la máquina. Además los nombres de las funciones deben ser `mysin()`, `mycos()`, etc.

5. *a*) Escriba un programa para calcular la media $\bar{x}$ y la desviación estándar σ de una secuencia finita x_i. Su programa debe aceptar un vector $\mathbf{x}$ de dimensión n como entrada y producir la media y la desviación estándar de la secuencia como salida. Para la desviación estándar, pruebe tanto la fórmula de dos pasos

$$\sigma = \left[\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \right]^{1/2}$$

como la fórmula de un paso

$$\sigma = \left[\frac{1}{n-1} \left(\sum_{i=1}^{n} x_i^2 - n\bar{x}^2 \right) \right]^{1/2}$$

y compare los resultados para una secuencia de entrada de su elección.

b) ¿Puede diseñar una secuencia de datos de entrada que ilustre dramáticamente la diferencia numérica entre estas dos fórmulas matemáticamente equivalentes? (Precaución: Tenga cuidado de no tomar la raíz cuadrada de un número negativo.)

```python
# Importar librerias
from myLibraries import myFunction


def funcionEjemplo(parametro):
    '''
    Docstring
    '''
    # Implementar solucion aqui
    resultado = parametro**2
    return resultado

# Programa principal
if __name__ == "__main__":
    datos = [1, 2, 3, 4]
    resultado = funcion_ejemplo(datos)
    print("Resultado:", resultado)
```

Listing 1: Ejemplo de estructura en Python