

ANÁLISIS NUMÉRICO I/ANÁLISIS NUMÉRICO – 2023

Trabajo de Laboratorio N^o 1

1. Dadas 3 variables numéricas x , y , z notar las diferencias entre:

a) $x/y + z$ y $x/(y + z)$

b) $x/y*z$ y $x/(y*z)$.

2. Comprobar que el épsilon-máquina es $2^{-52} = 2.2204 \times 10^{-16}$, escribiendo y comparando las siguientes dos líneas de comando:

```
>>> a = 1 + 2**-53;    b = a-1
```

```
>>> a = 1 + 2**-52;    b = a-1
```

3. Obtener el mayor y menor número positivo en punto flotante (overflow y underflow). Para obtener el mayor número de overflow escribir un ciclo que vaya calculando las sucesivas potencias de 2 y que finalice cuando se produce overflow. Se recomienda utilizar el comando `isinf` (importar la librería `math`) para detectar cuando se produce el overflow (escribir `help(math.isinf)` para obtener información sobre este comando). Otra instrucción que puede resultar útil es `break` para interrumpir el ciclo cuando se produce el overflow, o utilizar un `while`. El número de underflow se puede obtener dividiendo por 2 repetidamente hasta obtener un número indistinguible del cero en punto flotante.
4. Escribir la siguiente secuencia de comandos en un archivo con extensión `.py` y ejecutarlo.

```
x = 0
while x != 10:
    x = x + 0.1
```

Para interrumpir la ejecución, pulsar <CTRL>+<c>. ¿Qué ocurre si en lugar de incrementarse la variable en 0.1 lo hace en 0.5? ¿Por qué?

5. a) Escribir un programa que calcule el factorial de 6.
b) Importar la librería `math`, ¿qué función puede utilizar para calcular el punto anterior?
c) Escribir una función que calcule el factorial de un número n dado.
6. Escribir un programa que pida dos números reales e imprima en la pantalla el mayor de ellos. El programa debe indicar si los números son iguales.
7. Escribir una función que calcule la potencia n -ésima de un número, es decir que devuelva x^n para x real y n entero. Realice un programa que utilice la función e imprima en pantalla las primeras 5 potencias naturales de un número ingresado.

8. Escribir dos funciones en Python para la resolución de ecuaciones de segundo grado $ax^2 + bx + c = 0$. Una de ellas, que llamaremos **mala**, implementando la tradicional fórmula de Baskhara y la otra, que llamaremos **buena**, usando una manera eficiente para evitar cancelación de dígitos significativos. La sintaxis de la llamada a las funciones debería ser:

```
>>> [x_1, x_2] = buena(a,b,c)
```

y análogamente para **mala**.

9. Escribir una función, llamada **horn**, que implemente el algoritmo de Horner para la evaluación de polinomios. La sintaxis de llamada a la rutina debería ser:

```
>>> p = horn(coefs, x)
```

donde **p** es el valor del polinomio, **coefs** es un vector con los coeficientes del polinomio, de mayor a menor grado y **x** es el valor de la variable independiente. Es decir que si, por ejemplo, hacemos:

```
>>> p = horn([1, -5, 6, 2],2)
```

entonces la variable **p** almacenará el valor $p(2)$ donde $p(x) = x^3 - 5x^2 + 6x + 2$.

10. Escribir una función llamada **SonReciprococ(x,y)** que tenga dos números como input y devuelva **True** si son recíprocos, es decir, si $xy = 1$.

Luego ejecutar las siguientes instrucciones:

```
import random
for _ in range(100):
    x = 1 + random.random()
    y = 1/x
    if not SonReciprococ(x,y):
        print(x)
```

Explicar lo que sucedió.

11. Escribir dos funciones **f(x)** y **g(x)** que calculen:

$$f(x) = \sqrt{x^2 + 1} - 1$$

$$g(x) = \frac{x^2}{\sqrt{x^2 + 1} + 1}$$

Compruebe a mano que ambas expresiones son iguales y luego ejecute las siguientes instrucciones:

```
for i in range(20):
    x = 8**-i
    print(f"f(x)={f(x)}, g(x)={g(x)}")
```

¿Ambas funciones devuelven los mismos resultados? ¿Cuál es más confiable?

12. Escribir una función llamada `SonOrtogonales([x1,x2],[y1,y2])` que tenga como entrada dos vectores $x, y \in \mathbb{R}^2$ y que devuelva `True` si los vectores son ortogonales.

Ejecutar las siguientes instrucciones:

```
x = [1, 1.1024074512658109]
y = [-1, 1/x[1]]
if not SonOrtogonales(x,y):
    print("Algo salió mal...")
```

Explicar lo que sucedió.