Para entregar el 14 de Sept. antes de las 17:00 horas; reporte en formato PDF debe ser mandado a Jaime y Luis:
jaime.programacion.astronomica@gmail.com, gonzalez.29la@gmail.com
Atención: el nombre de su archivo PDF ha de seguir el siguiente formato: apellido_nombre_numerodetarea.pdf

Ejercicio 1: El método secante para encontrar raíces. El método Newton requiere conocimiento de la derivativa de f(x), f'(x), y que esta no sea igual a 0. Hay instancias en que estas condiciones no se cumplen. Por ejemplo, es posible que tengamos una función en forma de una tabla discreta. O a veces, la derivativa f'(x) puede ser una expresión muy complicada, y puede ser poco práctico calcularla de forma analítica.En estos casos, podemos usar el método del secante, un método que es 3000 años más antiguo que el método de Newton. Con este método, no usamos la derivativa f′(x), pero vamos a aproximarlo con la diferencia entre dos puntos. Suponga que tenemos dos supuestos x_n y x_n−1 del cero de la función. Entonces, podemos aproximar la derivativa como

$$f'(x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}.$$

Con esta aproximación, un supuesto mejor es

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

a) Implemente este método para determinar la raíz de las siguientes funciones:

- $f_1(x) = x^3 - 5$
- $f_2(x) = x^{3/5} - 16$
- $f_3(x) = \cos(x) - 6x$

Noten que $f_3(x)$ tiene más de un cero. Use el concepto de funciones en Python para implementar $f_1(x)$, $f_2(x)$ y $f_3(x)$.

b) Compare con el método de Newton. ¿Cuantas iteraciones necesita cada método con los ejemplos de arriba?

c) Use el concepto de funciones en Python para escribir una función que calcula la tercera raíz de un número del tipo float x con una precisión ε. x y ε sean parámetros formales de la función. Use el método de Newton o del secante dentro de la función. Repite el concepto de variables locales con este programa.

Ejercicio 2: Factoriales. Escriban dos funciones para calcular el factorial de un numero X entero y positivo, una función debe ser recursiva y la otra iterativa. Midan qual es la función mas rápida de ejecutar con un programa que compare los dos métodos.

Ejercicio 3: Fibonacci. Escriban 4 programas que calculen un numero N de la serie Fibonacci. Los programas deberán ser cada uno:

(a) iterativo usando for loops.

(b) iterativo usando while loops.

(c) recursivo,

y

(d) los programas a, b, y c calculan todos los números Fibonacci que vienen antes de N. Escriban un programa que devuelve el valor del numero N de la serie Fibonacci sin calcular los números previos.

Ejercicio 4: Midan qual es la función mas rápida de ejecutar con un programa que compare los diferentes métodos Fibonacci en el ejercicio 3 partes a, b, y c.

Ejercicio 5: Hagan un plot (un gráfico) del numero de iteraciones con el tiempo que se tomaron para los cuatro métodos Fibonacci (idealmente producirían un plot con 4 curvas).

Ejercicio 6: Escriban un código que imprima las primeras N filas (donde N ~ 10) del triángulo de Pascal. Den el código y el output en el PDF de la tarea.

Extra credit: This exercise does not necessarily involve writing code. Factorials: Remember the definition of factorials:

The factorial function is formally defined by the product

$$n! = \prod_{k=1}^{n} k$$

initially for integer $n \geq 1$, and resulting in this fundamental recurrence relation:

$$(n+1)! = (n+1) \cdot n!.$$

Given that (1/2)! = 1/2*Pi^0.5, give the exact expression of (-1/2)! [This part is analytical work, not coding work]. Give the exact value for Pi [=3.141592653589793238…] in terms of factorials of a negative fraction. What is the value of (-1/2)! to 10 decimal places. Hint: google can be used as a calculator. Which function allows us to calculate factorials of negative numbers?