Ejercicio 3

Aplicar el método de la secante para hallar la raíz no nula de $f(x) = \frac{x^2}{4} - sin(x)$

Lo interesante del ejercicio radica en poder implementar la función que realiza el método de la secante. ¿Qué vamos a necesitar?

1. Una función a la cual le queremos hallar la raíz
2. Dos puntos iniciales para poder calcular la recta secante. Recordamos que el nuevo punto de la iteración será la raíz de la línea secante, el cual será nuestra nueva aproximación. Las pendientes que estamos buscando igualar son:

$$\frac{f(b) - f(a)}{b - a} = \frac{0 - f(b)}{c - b}$$

3. Una tolerancia de error *eps* para hallar la solución
4. Como el método no tiene convergencia asegurada, podemos pedir un límite de iteraciones

Así, nuestro prototipo de función será $secante(f, a, b, eps, maxIter)$ y el valor que vamos a buscar en cada iteración será $c = b - f(b)\frac{b-a}{f(b)-f(a)}$

Necesitamos hacer una iteración previa a entrar en el bucle para revisar si hallamos la solución antes de comenzar a iterar, pues nuestro mecanismo de control (proximidad a cero de una función) requiere haber calculado previamente un estimado el valor

```
    function x = Secante(f,a,b,eps,maxIter)
    iter = 1
    fb = f(b)
    fa = f(a)
    actual = b - fb*(b-a)/(fb-fa);
    anterior = b;
    while (abs(f(actual)) >= eps) && (iter < maxIter)
        fant = f(anterior)
        fact = f(actual)
        siguiente = actual - fact * (actual-anterior)/(fact-fant)
        anterior = actual
        actual = siguiente
    end
    if iter == maxIter
        x = %nan
    else x = actual
    end
endfunction
```

¡Perfecto! Ya tenemos definida la función que implementa el método de la secante. Ahora nos falta resolver lo pedido en el enunciado.
Definamos en Scilab la función y, ya que estamos, un vector para graficarla:

```
--> deff("y=f(x)", "y=(x^2)/4 - sin(x)")

--> x = [-2:0.1:2];

--> plot(x,f(x))
```

```
--> a=gca();

--> a.x_location = "origin";

--> a.y_location = "origin";
```

Podemos ver que nos conviene tomar valores entre 1 y 3, un poco por arbitrariedad y otro poco porque estamos buscando la raíz no nula de la función. Vamos a elegir también una tolerancia al error de 10^{-5}, y un máximo de 10 iteraciones. Así basta ejecutar

```
--> Secante(f,1,3,0.00001,10)
 ans  =

   1.9337575
```

Y resulta que la raíz no nula de $f(x)$ es 1.9337575

Este es el código usado en el ejercicio:

```
    function y = dibujar()
    deff("y=f(x)", "y=(x^2)/4 - sin(x)")
    x = [-2:0.1:2];
    plot(x,f(x))
    a=gca();
    a.x_location = "origin";
    a.y_location = "origin";
    y = 0
endfunction

function x = Secante(f,a,b,eps,maxIter)
    iter = 1
    fb = f(b)
    fa = f(a)
    actual = b - fb*(b-a)/(fb-fa);
    anterior = b;
    while (abs(f(actual)) >= eps) && (iter < maxIter)
        fant = f(anterior)
        fact = f(actual)
        siguiente = actual - fact * (actual-anterior)/(fact-fant)
        anterior = actual
        actual = siguiente
    end
    if iter == maxIter
        x = %nan
    else x = actual
    end
endfunction
```

Ejercicio 8

Se quiere calcular la solución de la ecuación $e^x = 3x$, usando iteración simple con diferentes funciones de iteración:

- $g_1(x) = \frac{e^x}{3}$
- $g_3(x) = log(3x)$

¿Cuáles son útiles?

En este ejercicio queremos ver qué iteraciones de punto fijo podemos usar para resolver la ecuación planteada. Vamos a ver dos de las propuestas de la práctica, las otras dos quedan como ejercicio. Antes que nada, usamos un graficador para ver cómo se comporta la ecuación original.

Para saber si son útiles, tenemos que verificar si se cumple el teorema de condición suficiente de convergencia. Basta ver que existe un punto fijo y que el supremo del valor absoluto de la función es estrictamente menor a 1.

Vimos que los puntos fijos se encuentran en 0.6190613 y en 1.5121345

Función $\mathbf{g}_1(x)$

Tenemos que $0 \leq x \leq 1 \Rightarrow 0 \leq g_1(x) \leq 1$ y además $g_1'(x) = \frac{e^x}{3} < 1$ en dicho intervalo. Luego nos aseguramos que las iteraciones de punto fijo convergen y lo hacen al valor 0.6190613. Podemos probarlo de una manera muy sencilla al definir en la consola de Scilab un valor inicial de x y reescribirlo usando la función planteada. Por ejemplo:

```
--> x = 0
 x  =

   0.

--> x = exp(x)/3
 x  =

   0.3333333

--> x = exp(x)/3
 x  =

   0.4652041
   .
   .
   .
   .
--> x = exp(x)/3
 x  =

   0.6190613
```

Función $\mathbf{g}_3(x)$

Esta función tiene un comportamiento más interesante. Por un lado, si $x \leq 3$ vale que $g_3(x) \leq 3$. Ahora, ¿cuál debería ser el extremo inferior?
Queremos que en dicho intervalo la derivada $(\frac{1}{x})$ sea estrictamente menor a 1, por lo que el extremo inferior debería ser mayor a 1, tomemos 1.1 por ejemplo. Así quedaría resuelto y tomando de valor incial a 1.1 tenemos:

```
 --> x = 1.1
```

```
 x  =

   1.1

--> x = log(3*x)
 x  =

   1.1939225

--> x = log(3*x)
    .
    .
    .
    .
--> x = log(3*x)
 x  =

   1.5121345
```