Método de la bisección

https://es.wikipedia.org/wiki/M%C3%A9todo_de_bisecci%C3%B3n

```
function [y]=funcion(x)
    y=exp(-x)-abs(cos(x));
end

% Método de la bisección
function buscaceros()

% Datos iniciales
    a=input('Dame a: ');
    b=input('Dame b: ');

% Gráfica
    t=0:0.1:3;
    plot(t,funcion(t))
    grid on
    hold on
%   plot(t,zeros(length(t)),'g') % línea del cero
    plot([0 3],[0 0],'g') % línea del cero

% Inicializamos
    x=(a+b)/2;
    fa=funcion(a);
    fb=funcion(b);
    fx=funcion(x);
    pasos=0;
    if sign(fa)==sign(fb)
        fprintf('Quizás no hay un cruce por el cero\n')
    end

    % Búsqueda
    % En vez de hacer un número fijo de iteraciones con un ciclo for
    % for k:1NumIteraciones % por ejemplo
    % hacemos un ciclo mientras el error sea más de una milésima

    while abs(fx)>0.001 % distancia al y=0
        if fa>0 && fx > 0 % x y a son del mismo signo
            a=x;
        elseif fa<0 && fx < 0 %  x y a son del mismo signo
            a=x;
        else % x y b son del mismo signo
            b=x;
        end
        % Alternativa con sign
        % if sign(fa)==sign(fx)
        x=(a+b)/2;
        fa=funcion(a);
        fb=funcion(b);
```

```
            fx=funcion(x);
            pasos=pasos+1;
            punto=plot(x,fx,'.r', 'MarkerSize',10);
            % para hacer paso a paso espero tecla con pause
            % pause
            pause (0.3) % paramos 0,3 segundos
            delete(punto) % Borro el punto anterior
        end
        fprintf('El cero está en %f\n', x)
        % uiwait(msgbox(c1)) % entorno gráfico
        fprintf('Encontrado en %d pasos\n', pasos)
        plot(x,fx,'*r', 'MarkerSize',10)
    end
```

EJERCICIO

Escribe un *script* que produzca la siguiente salida, iterando de 1 a 9 para producir las expresiones de la izquierda y realizando la operación para imprimir exactamente en el siguiente formato:

```
>> beautyofmath
1 x 8 + 1 = 9
12 x 8 + 2 = 98
123 x 8 + 3 = 987
1234 x 8 + 4 = 9876
12345 x 8 + 5 = 98765
123456 x 8 + 6 = 987654
1234567 x 8 + 7 = 9876543
12345678 x 8 + 8 = 98765432
123456789 x 8 + 9 = 987654321
```

EJERCICIO

Escribe un *script* que calcule el siguiente valor para los números del 1 al 100, y lo represente en una gráfica.

Se debe calcular el número de veces que se aplica esta norma hasta que el número llega a 1. Si el número es par, se divide entre 2 y si es impar se multiplica por 3 y se le suma 1. Por ejemplo, para n=5, se hace cinco veces:

```
1. 5*3+1->16
2. 16/2->8
3. 8/2->4
4. 4/2->2
5. 2/2->1 FIN

1. 3*3+1->10
2. 10/2->5
3. 5*3+1->16
4. 16/2->8
5. 8/2->4
6. 4/2->2
7. 2/2->1 FIN
```

EJERCICIO Z

"No debía haberme ofrecido voluntaria", pensó Jenny.
"Claro, ahora es más fácil decir eso, con el tobillo torcido, y ese zombi tan rápido que quiere saborear mi cerebro.
Tengo que pensar rápido: como mucho puedo correr a 7km/h y si decido disparar a 5km/h pero el bicho calculo que se mueve a 3m/s. Menos mal que lo tengo a 10m.
Cada vez que le acierto un tiro baja un 20% de su velocidad máxima.
Calculo que tengo 30 segundos hasta que me recojan en ese coche que han ido a buscar con mi maniobra de despiste.
Espero que no aparezca ninguno más ni que se me acaben las balas. Me quedan 10.
No me voy a guardar ninguna para mí."

Haz un programa que calcule las distancias que van recorriendo Jenny y el zombi y que diga si el zombi cae o Jenny es alcanzada.
Como ayuda usa de ejemplo este programa de movimiento:

```
clc
clear all
hold on

d=0;  % distancia
v=3;  % velocidad
s=0;    % segundo
maxseg=30; % tiempo max
dt=0.5;   % actualizamos cada dt segundos
vd=[d];  % vector con distancias
vt=[0];  % vector con tiempos
while s<=maxseg
    plot([s],[d],'r*');
    vt=[vt s]; % Añado un nuevo elemento al vector de tiempos
    vd=[vd d];
    d=d+v*dt;  % Actualizo la distancia
    s=s+dt;    % Actualizo el tiempo
    v=v*0.9; % rozamiento
end
plot(vt, vd,'r') % tiempo contra distancia
plot([0 30],[0 0],'k'); % eje cero en negro
```