```
function tema4_1(arg)
clc
if nargin==0, arg='5'; end

switch(arg)

  case '1'
    s = 1.839286755214161;
    x=zeros(1,11); x(1)=1.8;
    for k=1:10,
      x(k+1) = power(1+x(k)+x(k)*x(k),1/3);
      e = abs(x(k+1)-s)/s;
      fprintf('Iter %2d :: x = %.16f -> erel %.2e\n',...
                 k,x(k+1),e)
    end
    e = abs(x-s)/s;
    figure(1); semilogy((0:10),e,'bo:','LineWidth',2)


    ratio=e(2:5)./e(1:4)
    K=e(end)/e(end-1); cif=log10(1/K);
    fprintf('K=%.3f -> %.2f cif/iter  = %.2f↙
iter/cif\n',...
        K,cif,1/cif);


    fprintf('---------------- NEWTON------------\n');
    x(1)=1.8;
    for k=1:10,
      xx=x(k);
      x(k+1) = xx - (xx^3-xx^2-xx-1)/(3*xx^2-2*xx-1);
      e = abs(x(k+1)-s)/s;
      fprintf('Iter %2d :: x = %.16f -> erel %.2e\n',...
                 k,x(k+1),e)
    end

      e = abs(x-s)/s;
```

```
      hold on; semilogy((0:10),e,'ro:','LineWidth',2); hold↙
off

      ratio=e(2:4)./e(1:3).^2


   case '2'
     %s = newton(@fun,1.5,1e-4), abs(s-sqrt(3))
     s = newton(@fun2,3,1e-12),  abs(s-pi)
     x=(3:0.001:3.3); plot(x,fun2(x));

   case '3'
     % Resolución de x^2-3
     fprintf('NEWTON **************\n');
     s1=newton(@fun,1.6,1e-10);
     fprintf('SECANTE *************\n');
     s2=secante(@fun,1.5,1.7,1e-10);
     abs(s2-sqrt(3))

   case '4a'

      R=(5:5:35)/100;
      plot(R,prestamo(R),'ro:',R([1 end]),[0 0],'k:');

      R=secante(@prestamo,0.20,0.25,1e-5);
      fprintf('Interes R=%.2f%%\n',R*100);

     case '4b'
      % Como R=0.75 la altura h no puede superar 1.5m
      % Podemos empezar entre 0.3 y 1.5
      % La precisión como mucho será ~ 1mm (1e-3 m)
      h = secante(@deposito,0.3,1.5,1e-3);
      fprintf('h final = %.3f m\n',h);

     case '5'

      N=70; x=zeros(1,N);
```

```
    x(1)=1; for k=1:N-1, x(k+1)=exp(-x(k)); end
    s=x(N), e=abs(x-s);
    semilogy((1:N),e,'bo:')

    K = mean(e(20:50)./e(19:49))
    cif_iter=log10(1/K)
    % K = 0.56, método más lento que bisección.

    % Newton
    x2(1)=1;
    for i=2:N,
      xx=x2(i-1);
      xx = xx - (xx-exp(-xx))/(1+exp(-xx));
      x2(i)=xx;
    end
    x=x2;

    e = abs(x-s);
    hold on; semilogy((1:N),e,'ro:'); hold off

    fprintf('K en Newton %.3f %.3f %.3f\n',...
        e(2:4)./e(1:3).^2)


end


% Funciones de los métodos
function s=newton(fx,x0,tol)
% Entrada  fx  puntero a función (devolver f(x) y f'(x))
%          x0  punto inicial
for iter=1:10,
  [f,fp]=fx(x0); % Pido valores de f(x) y f'(x)
  if f==0, break; end  % He tenido suerte (me salgo)
  x1 = x0-(f/fp);      % Iteracion de Newton.
  fprintf('%2d -> %18.16f\n',iter,x1);  % Vuelco valores
  if abs(x1-x0)<tol, break; end
```

```
  x0=x1;    % Actualizo x0 <- x1 para volver a iterar.
end
s = x1;  % Devuelvo último término de la sucesión.
return


function s=secante(fx,x0,x1,tol)
% Entrada fx, funcion cuyo cero deseamos.
%           I = [x0 x1] = puntos de partida
%           tol_user = error RELATIVO máximo (por defecto↙
1e-8)
% Salida   s, estimación de raíz tras N iteraciones

if nargin==3, tol=1e-8; end  % Si no hay tercer argumento↙
tol=1e-8

% Evaluación en 1er y 2do punto.
f0=fx(x0); f1=fx(x1);

iter=1;
while (abs(x1-x0)>tol) && (iter<=10)

  aprox_fp = (f1-f0)/(x1-x0);
  x2 = x1 - f1/aprox_fp;

  x0=x1; f0=f1;
  x1=x2; f1=fx(x1);      % actualizo x0 y x1.

  % Vuelco valores
  fprintf('%2d -> %18.16f  dx %4.2e\n',iter,x1,abs(x1-x0));

  iter=iter+1;
end

s = x1;  % solucion = último punto
return
```

```
% Funciones f(x) a resolver
function [f fp]=fun(x)
  f = x*x-3;
  fp = 2*x;
return

function [f fp]=fun2(x)
 f= 1 + cos(x);
 fp = -sin(x);
return

function f=fun3(x)
 f = x-exp(-x);
return

function f=prestamo(R)
 M=60; P=1540; N=36;
 r=R/12;
 f = M-P*r./(1-(1+r).^-N);
return

function f = deposito(h)
 R = 0.75;
 h0=0.3; V0=pi*h0^2*(R-h0/3); % Volumen inicial
 Vf = V0+1;  % Vfinal = V0 + 1000 litros
 f = Vf - pi*h*h*(3*R-h)/3;
return
```