```
function [xMin,fMin,flag,output] = triSection(fun,a,b,opt)
% Minimize a function via trisection method
if (nargin < 4), opt = setDefaultOptimizationOptions(); end % default parameters
iter = 0;
flag = 0;
fEvals = 0;
output.message = '#FunEvals exceeded'; % default
fprintf('-------------------------------------------------\n');
fprintf('#    x1      x2      f(x1)     f(x2)    b - a\n');
fprintf('-------------------------------------------------\n');
while(fEvals < opt.MaxFunEvals)
    if ((b-a) < opt.TolX)
        output.message = 'X tol met';
        flag = 1;
        break;
    end

    x1 = a + (b-a)/3;
    x2 = a + (b-a)*2/3;
    f1 = fun(x1);
    f2 = fun(x2);
    fEvals = fEvals + 2; % note 2 function calls every iteration
    fprintf('%d %.2e %.2e %.2e %.2e %.2e\n', iter,x1, x2, f1, f2, b-a);
    % We don't test against opt.TolFun since it is possible that f1 = f2, and this
    % will lead to premature exit
    if (f1 > f2)
        a = x1;
    else % f1 = f2, or f1 < f2
        b = x2;
    end
    iter = iter + 1;
end
xMin = (x1+x2)/2;
fMin = fun(xMin);
output.iterations = iter;
output.funcCount = fEvals;
```