Controlling Access

```
classdef Poly1 < handle
    % A Poly1 object is a polygon and its centroid

    properties
        x = [];     % x-coordinates of the vertices
        y = [];     % y-coordinates of the vertices
        xc          % x-coordinate of the centroid
        yc          % y-coordinate of the centroid
    end

    methods
        function P = Poly1(x,y)
        % Constructs a Poly1 object.
        % x and y are column n-vectors
            P.x = x; P.y = y;
            n = length(x);
            P.xc = sum(x)/n; P.yc = sum(y)/n;
        end

        function MoveToOrigin(ThisPoly1)
        % Translates ThisPoly1 to the origin
            ThisPoly1.x = ThisPoly1.x - ThisPoly1.xc;
            ThisPoly1.y = ThisPoly1.y - ThisPoly1.yc;
            ThisPoly1.xc = 0;
            ThisPoly1.yc = 0;
        end

        function Show(ThisPoly1,c)
        % Displays ThisPoly1 in the current figure window with color c.
        % Assumes hold is on.
            x  = ThisPoly1.x;    y = ThisPoly1.y;
            xc = ThisPoly1.xc;  yc = ThisPoly1.yc;
            plot([x x(1)],[y y(1)],c,'linewidth',2)
            plot(x,y,'ok')
            plot(xc,yc,['.' c],'Markersize',20)
        end
    end

end

% Script ShowPoly1
% Illustrates the class Poly1.

close all
plot([-10 10],[0 0],':k',[0 0],[-10 10],':k')
axis equal
axis([-10 10 -10 10])
hold on

% Construct and display a heptagon...
x = 6 + 2*cos([0 30 100 150 200 280 320]*pi/180);
y = 5 + 3*sin([0 30 100 150 200 280 320]*pi/180);
P = Poly1(x,y);
P.Show('m')

% Remove its last vertex...
n = length(P.x);
P.x = P.x(1:n-1);
P.y = P.y(1:n-1);
P.xc = sum(P.x)/(n-1);
P.yc = sum(P.y)/(n-1);

% Translate the resulting hexagon to the origin and display...
P.MoveToOrigin();
P.Show('b')
```

```
classdef Poly2 < handle
    % Same as Poly1 except that its properties are private

    properties(Access= private)
        % These properties cannot be accessed outside the class
        x = [];     % x-coordinates of the vertices
        y = [];     % y-coordinates of the vertices
        xc          % x-coordinate of the centroid
        yc          % y-coordinate of the centroid
    end

    methods
        function P = Poly2(x,y)
        % Constructs a Poly2 object.
        % x and y are column n-vectors
            P.x = x;
            P.y = y;
            n = length(x);
            P.xc = sum(x)/n;
            P.yc = sum(y)/n;
        end

        function DeleteLast(ThisPoly2)
        % Deletes the last vertex in ThisPoly2. Assumes that the
        % represented polygon has at least 2 sides.
        n = length(ThisPoly2.x);
        ThisPoly2.x = ThisPoly2.x(1:n-1);
        ThisPoly2.y = ThisPoly2.y(1:n-1);
        ThisPoly2.xc = sum(ThisPoly2.x)/(n-1);
        ThisPoly2.yc = sum(ThisPoly2.y)/(n-1);
        end

        function MoveToOrigin(ThisPoly2)
        % Translates ThisPoly2 to the origin
            ThisPoly2.x = ThisPoly2.x - ThisPoly2.xc;
            ThisPoly2.y = ThisPoly2.y - ThisPoly2.yc;
            ThisPoly2.xc = 0; ThisPoly2.yc = 0;
        end

        function Show(ThisPoly2,c)
        % Displays ThisPoly1 in the current figure window with color c.
        % Assumes hold is on.
            x  = ThisPoly2.x;   y = ThisPoly2.y;
            xc = ThisPoly2.xc; yc = ThisPoly2.yc;
            plot([x x(1)],[y y(1)],c,'linewidth',2)
            plot(x,y,'ok')
            plot(xc,yc,['.' c],'Markersize',20)
        end
    end
end

% ShowPoly2
% Illustrates the class Poly2.

close all
plot([-10 10],[0 0],':k',[0 0],[-10 10],':k')
axis equal
axis([-10 10 -10 10])
hold on

% Construct and display a heptagon...
x = 6 + 2*cos([0 30 100 150 200 280 320]*pi/180);
y = 5 + 3*sin([0 30 100 150 200 280 320]*pi/180);
P = Poly2(x,y);
P.Show('m')
% Remove its last vertex...
P.DeleteLast();
% Translate the resulting hexagon to the origin and display...
P.MoveToOrigin();
P.Show('b')
```