> restart;

bezier := proc(pt::list, rg::list, lg::list, b::name)
    local N, X, Y, XPL, YPL, I, XMI, YMI, A0, B0, A1, B1, A2, B2, A3, B3, B;
    N := nops(pt) - 1;
    if nops(lg) <> N then
        ERROR("The left guidepost list is the wrong size.");
    fi;
    if nops(rg) <> N then
        ERROR("The right guidepost list is the wrong size.");
    fi;
    for I from 0 to N do
        X[I] := op(1, pt[I + 1]);
        Y[I] := op(2, pt[I + 1]);
        if I <> N then XPL[I] := op(1, rg[I + 1]) fi;
        if I <> N then YPL[I] := op(2, rg[I + 1]) fi;
        if I <> 0 then XMI[I] := op(1, lg[I]) fi;
        if I <> 0 then YMI[I] := op(2, lg[I]) fi;
    od;

    STEP 1
    for I from 0 to N - 1 do

    STEP 2
    A0[I] := X[I];
    B0[I] := Y[I];
    A1[I] := 3*(XPL[I] - X[I]);
    B1[I] := 3*(YPL[I] - Y[I]);
    A2[I] := 3*(X[I] + XMI[I + 1] - 2*XPL[I]);
    A3[I] := X[I + 1] - X[I] + 3*XPL[I] - 3*XMI[I + 1];
    B3[I] := Y[I + 1] - Y[I] + 3*YPL[I] - 3*YMI[I + 1];
    od;

    STEP 3
    for I from 0 to N - 1 do
        B[I] := [X[I], Y[I]], [X[I + 1], Y[I + 1]], x[I](t) = A3[I]*t^3 + A2[I]*t^2 + A1[I]*t + A0[I], y[I](t) = B3[I]*t^3 + B2[I]*t^2 + B1[I]*t + B0[I];
        od;
        b := [seq(B[I], I = 0 .. N - 1)];

    STEP 4
    end;

Warning, imaginary unit `I` used as a local variable in procedure bezier
b := seq(B[I], I = 0 .. N - 1))
end proc

> bezier_dir:=proc()
>
> printf(`bezier returns a list where each component is a list\n`);
> printf(`containing a starting point, an ending point, and the two\n`);
> printf(`cubic parametric equations between the points.\n`);
> printf(`The arguments for bezier are:\n`);
> printf(`(1) the list of points [x[i],y[i]] for i=0..n\n`);
> printf(`(2) the list of left (or exiting) guideposts [x[i]+,y[i]+] for i=0..n-1\n`);
> printf(`(3) the list of right (or entering) guideposts [x[i]-,y[i]-] for i=1..n\n`);
> printf(`(4) the variable for returning the bezier curve\n`);
> printf(`If assigning the result to a variable, have the\n`);
> printf(`and has already been given a value,\n`);
end;

bezier_dir := proc()
print( 'bezier returns a list where each component is a list' );
print( 'containing a starting point, an ending point, and the two' );
print( 'cubic parametric equations between the points. ' );
print( 'The arguments for bezier are: ' );
print( '(1) the list of points [x[i],y[i]] for i=0..n' );
print( '(2) the list of left (or exiting) guideposts [x[i]+,y[i]+] for i=0..n-1' );
print( '(3) the list of right (or entering) guideposts [x[i]-,y[i]-] for i=1..n' );
print( '(4) the variable for returning the bezier curve' );
print( 'If assigning the result to a variable, have the' );
print( 'and has already been given a value, ' );
end;
\begin{verbatim}
printf("the procedure should be preceded by the statement: ");
printf("b := 'b'")
end proc
\end{verbatim}