Since you already have an explanation for quadratic splines from discussion, I figured I would cover cubic splines for the ones of you that haven't been able to make it to office hours. Keep in mind that for this derivation I assume we are using the following form of the cubic equation.

$$y = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

Since we have four unknown coefficients, we need four equations and these come from the following conditions.

1. $y_i = a_i + b_i(x_i - x_i) + c_i(x_i - x_i)^2 + d_i(x_i - x_i)^3 = a_i$
2. $y_{i+1} = a_i + b_i(x_{i+1} - x_i) + c_i(x_{i+1} - x_i)^2 + d_i(x_{i+1} - x_i)^3$ (Continuity of y)
3. $y'_{i+1} = b_i + 2c_i(x_{i+1} - x_i) + 3d_i(x_{i+1} - x_i)^2 = b_{i+1}$ (Continuity of y')
4. $y''_{i+1} = 2c_i + 6d_i(x_{i+1} - x_i) = 2c_{i+1}$ (Continuity of y'')

You will notice that these equations are not directly solveable at a single point as there are 6 unknowns since they depend on both b_{i+1} and c_{i+1}. So we need to do a little algebra and a little thinking before going on... Now I don't want to type all the steps, not to mention it is good practice for you to do them yourself, so I will merely state the result. From the above system, we can derive the following equation:

$$\frac{1}{6}b_{i-1} + \frac{4}{6}b_i + \frac{1}{6}b_{i+1} = \frac{y_{i+1} - y_{i-1}}{2\Delta x} \tag{1}$$

This gives you an equation for the value b everywhere, and it has the additional benefit of looking suspiciously tri-diagonal. The only problem is that it requires that you know $b(1)$ and $b(n)$. These can be approximated using a quadratic spline. Below is the system of equations you would have to solve to find $b(1)$. Note that this is the same system you should have solved to find b(1) in the quadratic spline case...

1. $y(1) = a(1)$
2. $y(2) = a(1) + b(1)\Delta x + c(1)\Delta x^2$
3. $y(3) = a(1) + 2b(1)\Delta x + 4c(1)\Delta x^2$

You know delta x, you know all of y and so you can solve this system for $b(1)$. This looks suspiciously like a 2x2 matrix so you could copy and paste your code from project 1... Now you just need to re-write this system at $i = n$. As a check, keep in mind that b is actually the derivative of the cubic spline, so your value should be similar to the exact derivative. Once you have $b(1)$ and $b(n)$ you can find all other values of b using the tri-diagonal solver and the equation I listed previously. *Be careful not to mix up a,b,c and d from your tri-diagonal coefficients with the coefficients of your cubic functions, they are different!!!*. Below is a rather unsubtle hint as to how your tri-diagonal system should look...

and Settings/Michael/Desktop/ENG-180/tricubic.png

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & & & \\ \frac{1}{6} & \frac{4}{6} & \frac{1}{6} & 0 & \cdots & & \\ & \ddots & \ddots & \ddots & & & \\ \cdots & 0 & \frac{1}{6} & \frac{4}{6} & \frac{1}{6} & 0 & \cdots \\ & & & \ddots & \ddots & \ddots & \\ & & & \cdots & \ddots & \ddots & \ddots \\ & & & \cdots & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ \vdots \\ b_i \\ \vdots \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ \vdots \\ d_i \\ \vdots \\ \vdots \\ d_n \end{bmatrix}$$

Where $d(1)$ and $d(n)$ correspond to the the slopes you calculated from fitting a quadratic spline to the first and last point respectively. Now you have all of a and b so the only thing missing is c and d and these can be found from the continuity and the continuity of the derivatives listed in the orignal system of equation we were trying to solve. You will notice that this yields another 2x2 so whip out project 1 and copy paste. Having said all this, let's have a look at a qausi-psuedocoding of the problem...

```
%Cubic spline algorithm
clc, clear all, close all
%Setup x and y
xmin =
xmax =
Nx = %number of points in x
Nxx = %number of sup-points on spline
dx =
for i = 1:Nx
x(i) =
y(i) =
end
%Solve for first and last b by fitting a quadratic
....
%Setup tridiagonal system
aa(1) = 0; aa(n) = 0;
bb(1) = 1; bb(n) = 1;
cc(1) = 0; cc(n) = 0;
dd(1) = b(1); dd(n) = b(n);
for i = 2:n-1
aa(i) =
bb(i) =
cc(i) =
dd(i) =
end

%Solve for c and d and plot spline
figure(1),hold on
for i = 1:n-1
c(i) = %from 2x2 system
```

```
d(i) = %from 2x2 system
%Setup and plot spline
dxx = (x(i)-x(i+1))/(Nxx-1);
for j=1:Nxx
xx(j) = x(i) + (j-1)*dxx;
yy(j) = a(i) + b(i)*(xx(j)-x(i)) + ...
end
plot(xx,yy)
end
%Label plot and make it pretty
```

I would reccomend copying and pasting the above code into MatLab, hiting ctrl-a and then right-click and choose smart indent. This will make it a little easier to visualize. GOOD LUCK!!!