CS3451-Fall 2014, P03 REPORT

The title: CS3451 Fall 2014, Project 5 Coon Patch and Neville Interpolation

Student: Bo Chen

Project description: Implementing the algorithm as known as Coons Path, interpolating between left and right curves, the top and bottom curves as guides. Also creating a animation shown left curve transfer to right curve.

In the Coons Path interpolates a point(s, t) on surface by generating three points, and computing a weight average of these three points.

Bezier:

Coons Path generates a point by interpolating top and down curves by using t, then interpolating across the line that joins them (the grey horizontal line in the picture) by s:

Lt = L(leftCurve(P, s, t), rightCurve(P, s, t), t);

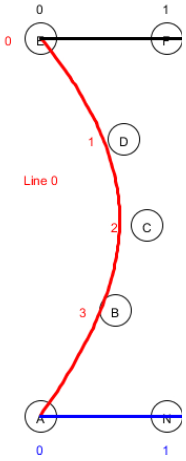
And inside the lerp method, need to create different lerp for multi points:

```
pt B(pt A, pt B, pt C, float t){
  return L(L(A, B, t), L(B, C, t), t);
}

pt B(pt A, pt B, pt C, pt D, float t){
  return L(B(A, B, C,t), B(B, C, D, t), t);
}

pt B(pt A, pt B, pt C, pt D, pt E, float t){
  return L(B(A, B, C, D, t), B(B, C, D, E, t), t);
}
```

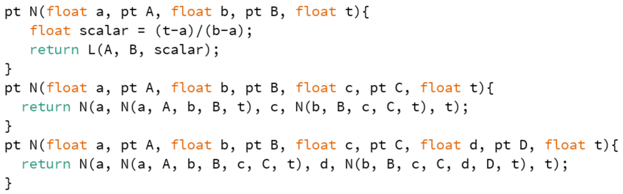
In the Bezier curve:



Neville:

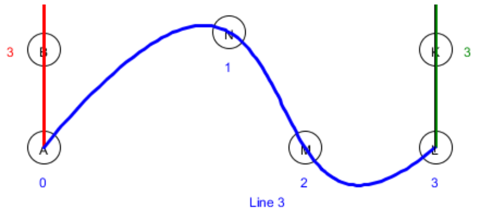
Coons Path generates one point by first interpolating across the horizontal control curves by s, then interpolating down the line that joins them (the grey vertical line) by t.

Ls = L(topCurve(P, s, t), downCurve(P, s, t), s);



```
pt N(float a, pt A, float b, pt B, float t){
   float scalar = (t-a)/(b-a);
   return L(A, B, scalar);
}
pt N(float a, pt A, float b, pt B, float c, pt C, float t){
  return N(a, N(a, A, b, B, t), c, N(b, B, c, C, t), t);
}
pt N(float a, pt A, float b, pt B, float c, pt C, float d, pt D, float t){
  return N(a, N(a, A, b, B, c, C, t), d, N(b, B, c, C, d, D, t), t);
}
```

In the Neville, the curve touch all points:



Coons path, It generates one point by the bilinear interpolation of the four outer control points (D, G, A, and J in the picture), shown near the intersection of the two lines.

pt x = P(Lt.x+Ls.x-Lst.x, Lt.y+Ls.y-Lst.y);

All curves are constructed using the LERP function, with A and B as points, and time t, whenever there are more than two points need to use lerp, the lerp function is repeated as above code shows.

Pt newPoint = A + t(AB).

But in the Neville, we don’t use t(AB), instead of it, by using (t-a)/(b-a) (AB) with a and b the time for A, B respectively.