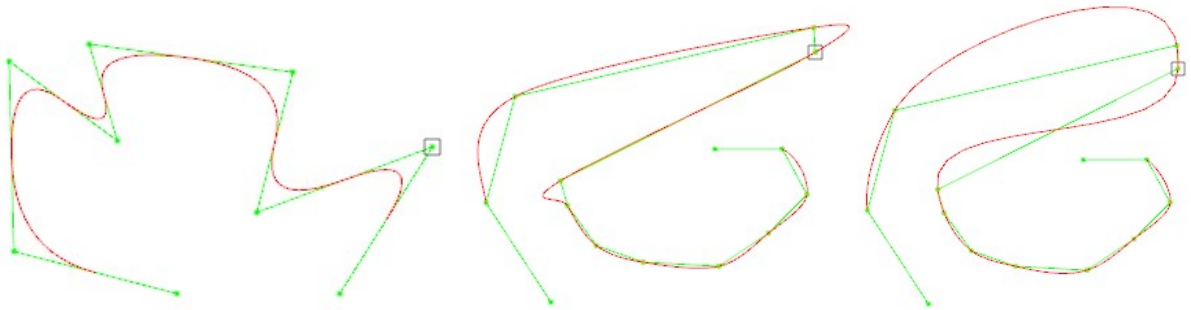
**Programing Assignment #8 - Splines**

In this assignment you will implement a **quadratic B-Spline** using the Cox-de Boor formulation with uniform knot vectors, and two Spline Interpolators: **Catmull-Rom, and Bessel-Overhauser**. Their formulations are available in the lecture slides.



**Quadratic BSpline** **Catmull-Rom** **Bessel-Overhauser**

In this assignment you will be extending your implementation of the previous assignment with these new curves.

1) Implement the following functions to evaluate points in each spline. These functions will not make calls to OpenGL. Please implement them in a dedicated file:

```
/*! B-Spline order k, n=pnts.size()-1;
    For order k=3, (degree 2, quadratic case): t in [2,n+1]  */
GsVec eval_bspline ( float t, int k, const GsArray<GsPnt2>& pnts );

/*! Evalutes a Catmull-Rom cubic spline, n=pnts.size()-1, t in [0,n-2] */
GsVec crspline ( float t, const GsArray<GsVec>& pnts );

/*! Evalutes a Bessel-Overhauser spline, n=pnts.size()-1, t in [0,n-2] */
GsVec bospline ( float t, const GsArray<GsVec>& pnts );
```

(small variations in the function prototypes are fine, for example, you can choose to use GsArray or std::vector; GsVec or GsPnt2, etc.)

2) You will then evaluate points in your curves as in the previous assignment, but now use the following keys to choose which curve to draw:

**0** - no curve
**3** - Quadratic BSpline
**4** - Catmull Rom Spline
**5** - Bessel Overhauser Spline

3) As before, you will also need to be able to change the number of segments used to approximate each curve; use the following keys:

**q** - increase the number of segments
**a** - decrease the number of segments

4) Integrate this assignment with the previous one, so that keys 1,2,3,4, 5 will each show a different type of curve.

5) You will also generate a simple 3D surface from the current curve being edited in the 2D XY plane, in the exact same way as described in the previous assignment.

6) As for additional controls, just maintain what you already have from the previous assignment.

**Grading**

50% - Correct curves are generated.
15% - Resolution is correctly controlled.
15% - 3D Surface is correctly generated with correct smooth shading.
10% - real-time curve generation is correct, and z and x keys work as expected.
10% - all controls are correct, the results look good, and the application is well developed.