CADCG practical – programming part 4:
B-Spline curves

Generate a CodeBlocks project with the files from the zip file by using CMake. Select "curve-static" as build target before compiling and running.

B-Spline curves and the Cox-de Boor algorithm

For this part of the work, only the file `api/nbspline.cc` needs to be edited.

1. Implement the interpolating `P` function from the `nbspline` class with the Cox-de Boor algorithm (Theory: ppt 4 pp.80-93).

   Tip 1: Use the function `findspan` already implemented

   ```
   int findspan(const std::vector<double> &knots,
                int nknots, int deg, double u) const
   ```

   returns an index `i` such that `u` lies in the interval $[u_i, u_{i+1}[$ for the nodal sequence `knots` of length `nknots` for the degree `deg`.

   Tip 2: Points do not have the same indexes as in de Casteljau's algorithm.

2. Compile and test your program, the B-Spline curve must appear in the window in white.

Closed B-spline curves

We wish to draw two closed B-spline curves with control points located on a full circle (but the curves themselves will not be circles). The first curve will use a non-uniform nodal sequence with interpolating first and last control point. The second curve will use a periodic nodal sequence (thus uniform sequence). These functions to implement are in the file `test/curves/main.cc` under the main function (Theory: ppt 4 pp. 26, 57-62).

1. Define the first closed curve in the function `init_bspline_nonuniform`. The curve generated by this function should use a non-uniform nodal sequence and interpolate its first and last control point.
2. Define the second closed curve in the function `init_bspline_periodic`. Firstly, implement a uniform nodal sequence and look at the curve. Then, modify the location of the control points to assure periodicity.

For these functions, the curves must remain closed for any number of control points and any degree. Tip: get inspiration from the function `init_bspline`.