Assignment 2

Assignment 2:

Implement an algorithm that is capable of rendering a cubic Bezier surface segment. Approximate the surface using the deCasteljau algorithm. For this, apply the deCasteljau algorithm. To allow for correct lighting, compute the normal vectors for each point as the cross product of the derivatives of the surface in both parameter directions. Your program should allow for an arbitrary approximation precision which can be increased or decreased based on user input (e.g. via key press).

Initially, assume that all Bezier points are located within a plane. By picking a point (you can either use OpenGL picking or gluUnproject), your program should allow the user to move the selected point within the image plane via mouse input. Your rendering of the surface should interactively adapt to the change of the Bezier point.

Assignment 2

Using gluUnProject

```
int gluUnProject(GLdouble winx,
                 GLdouble winy,
                 GLdouble winz,
                 const GLdouble modelMatrix[16],
                 const GLdouble projMatrix[16],
                 const GLint viewport[4],
                 GLdouble *objx,
                 GLdouble *objy,
                 GLdouble *objz)
```

Assignment 2

Using gluUnProject

```
Point3D unproject (int x, int y) {
  GLdouble model[16], proj[16];
  GLint viewport[4];
  GLfloat z;
  Point3D point;
  glGetDoublev (GL_MODELVIEW_MATRIX, model);
  glGetDoublev (GL_PROJECTION_MATRIX, proj);
  glGetIntegerv (GL_VIEWPORT, viewport);
  glReadPixels (x, viewport[3] - y, 1, 1,
                        GL_DEPTH_COMPONENT, GL_FLOAT, &z);
  gluUnProject ((GLdouble)x,  (GLdouble)(viewport[3] - y),  (GLdouble)z,
                        model, proj, viewport, &(point[0]), &(point[1]), &(point[2]));
 return point;
}
```

Assignment 2

Using gluProject

```
int gluProject(GLdouble objx,
               GLdouble objy,
               GLdouble objz,
               const GLdouble modelMatrix[16],
               const GLdouble projMatrix[16],
               const GLint viewport[4],
               GLdouble *winx,
               GLdouble *winy,
               GLdouble *winz)
```

Assignment 2

Using gluProject

```
void project (double &x, double &y, Point3D point) {
  GLdouble model[16], proj[16];
  GLint viewport[4];
  double z;
  glGetDoublev (GL_MODELVIEW_MATRIX, model);
  glGetDoublev (GL_PROJECTION_MATRIX, proj);
  glGetIntegerv (GL_VIEWPORT, viewport);
  gluProject (point[0], point[1], point[2],
                    model, proj, viewport,
                    &x, &y, &z);
}
```