Homework 8 - Bezier Curve

Basic:

1. 用户能通过左键点击添加Bezier曲线的控制点，右键点击则对当前添加的最后一个控制点进行消除
2. 工具根据鼠标绘制的控制点实时更新Bezier曲线。

Hint: 大家可查询捕捉mouse移动和点击的函数方法

Bonus:

1. 可以动态地呈现Bezier曲线的生成过程。

首先实现鼠标点击的功能，在这里需要捕获鼠标的点击事件和鼠标的位置

一开始先将鼠标的位置设置成全局变量，这样一来可以在函数中更改鼠标的位置

```
16      loat mousePosX = 0;
17      loat mousePosY = 0;
```

```
174  void mouse_button_callback(GLFWwindow* window, int button, int action, int mods) {
175      if (action == GLFW_RELEASE) {
176          if (button == GLFW_MOUSE_BUTTON_LEFT) {
177              points.push_back(mousePosX);
178              points.push_back(mousePosY);
179          }
180          if (button == GLFW_MOUSE_BUTTON_RIGHT) {
181              if (points.size() > 0) {
182                  points.pop_back();
183                  points.pop_back();
184              }
185          }
186      }
187  }
188
189  void cursor_position_callback(GLFWwindow* window, double x, double y) {
190      mousePosX = float((x - SCR_WIDTH / 2) / SCR_WIDTH) * 2;
191      mousePosY = float(0 - (y - SCR_HEIGHT / 2) / SCR_HEIGHT) * 2;
192  }
```

鼠标移动事件是一直触发的，当点击鼠标左键释放后会将当前鼠标位置放到一个vector容器中，这个位置会被用来生成新的点，点击鼠标右键释放后会删掉最后一个控制点，此外还需要注册这两个函数

```
49      glfwSetMouseButtonCallback(window, mouse_button_callback);
50      glfwSetCursorPosCallback(window, cursor_position_callback);
```

Beizier曲线的实现只需要根据老师课件给的方法来实现就行了

formula is as follows:

$$B_{i,n}(t)=\frac{n!}{i!(n-i)!}t^i(1-t)^{n-i},\ i=0,\ 1\ldots n$$

$$Q(t)=\sum_i^3 P_iB_{i,3}(t)=P_0B_{0,3}(t)+P_1B_{1,3}(t)+P_2B_{2,3}(t)+P_3B_{3,3}(t),\quad t\in[0,1]$$
$$=(1-t)^3P_0+3t(1-t)^2P_1+3t^2(1-t)P_2+t^3P_3$$

```
200 void getBezierVertices() {
201     if (points.size() > 2) {
202         linepoints.clear();
203         for (float t = 0.0f; t <= 1.0f; t = t + 0.01f) {
204             float xsum = 0;
205             float ysum = 0;
206             int n = points.size() / 2 - 1;
207             for (int j = 0; j < points.size() / 2; j++) {
208                 float coefficient = (factorial(n) / (factorial(j)*factorial(n - j)))
209                     * pow(t, j) * pow(1 - t, n - j);
210                 xsum += coefficient * points[j * 2];
211                 ysum += coefficient * points[j * 2 + 1];
212             }
213             linepoints.push_back(xsum);
214             linepoints.push_back(ysum);
215         }
216     }
217 }
```

为了让点显示得更清楚可以设置点的大小：

```
83          // 设置点的大小
84          glad_glPointSize(10);
```

最后的效果如下所示：