PRIMER CODIGO

```
#include <iostream>
#include <GL/glut.h>
#include <math.h>

int W = 800,H = 600;
int P[2][3];
int Hi = 0;

void Pixel(float x, float y) {
	glPointSize(2);
	glColor3f(0, 0, 1);
	glBegin(GL_POINTS);
	glVertex2f(x, y);
	glEnd();
	glFlush();

}
void Spline(int P[][3]) {
	for (float t = 0; t <= 1; t += 0.001)
	{
		Pixel(P[0][0] * pow(1 - t, 2) + P[0][1] * 2 * t * (1 - t) + P[0][2] *
pow(t, 2),
			P[1][0] * pow(1 - t, 2) + P[1][1] * 2 * t * (1 - t) + P[1][2] *
pow(t, 2));
	}

}
void Mouse(int btn, int state, int x, int y) {
	if (btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN && Hi == 0)
	{
		P[0][0] = { x };
		P[1][0] = { y };
		Hi++;
		return;
	}
	if (btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN && Hi == 1)
	{
		P[0][1] = { x };
		P[1][1] = { y };
		Hi++;
		return;
	}
	if (btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN && Hi == 2)
	{
		P[0][2] = { x };
		P[1][2] = { y };
		Spline(P);
		Hi = 0;
		return;
	}

}
void vis() {
	glFlush();
}
```

```
int main(int argc, char** argv) {
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGBA);
	glutInitWindowSize(W, H);
	glutInitWindowPosition(0, 0);
	glutCreateWindow("Curvas Spline");
	glutDisplayFunc(vis);
	glutMouseFunc(Mouse);
	gluOrtho2D(0, W, H, 0);
	glClearColor(1, 1, 1, 1);
	glutMainLoop();
	return 0;
}
```

SEGUNDO CODIGO

```
#include <iostream>
#include <GL/glut.h>
#include <math.h>


// Contador para numero de clicks
int num = 0;

// Configuracion de los puntos x,y con clases
class punto {
public:
	float x, y;
	void configxy(float x2, float y2) { x = x2; y = y2; }
	const punto& operator=(const punto& rpunto) {
		x = rpunto.x;
		y = rpunto.y;

		return *this;
	}

};

punto abc[4];

void pintapixel(int x, int y, int R, int G, int B) {
	glBegin(GL_POINTS);
	glVertex2d(x, y);
	glEnd();
	glPointSize(4);
	glFlush();
}

void pintalinea(punto p1, punto p2) {
	glBegin(GL_LINES);
	glVertex2f(p1.x, p1.y);
```

```
	glVertex2f(p2.x, p2.y);
	glEnd();
	glFlush();
}

// CALCULO DE PUNTOS
punto spline(punto A, punto B, punto C, punto D, double t) {
	punto P;
	P.x = pow((1 - t), 3) * A.x + 3 * t * pow((1 - t), 2) * B.x + 3 * (1 - t) * pow(t,
2) * C.x + pow(t, 3) * D.x;
	P.y = pow((1 - t), 3) * A.y + 3 * t * pow((1 - t), 2) * B.y + 3 * (1 - t) * pow(t,
2) * C.y + pow(t, 3) * D.y;
	return P;
}

void Mouse(int btn, int state, int x, int y) {
	if (btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN) {
		// Se guardan los clicks
		abc[num].configxy((float)x, (float)(600 - y));
		num++;
		pintapixel(x, 600 - y, 1, 1, 1);

	}
	if (btn == GLUT_RIGHT_BUTTON && state == GLUT_DOWN) {
		if (num == 4) {
			glColor3f(0, 1, 1);
			// Se pintan las piernas del rectangulo
			pintalinea(abc[0], abc[1]);
			pintalinea(abc[1], abc[2]);
			pintalinea(abc[2], abc[3]);
			punto pini = abc[0];
			// Se dibuja la curva
			for (double t = 0; t <= 1; t += 0.1) {
				punto P = spline(abc[0], abc[1], abc[2], abc[3], t);
				pintalinea(pini, P);
				pini = P;
			}
			glColor3f(1, 0, 0);
			num = 0;
		}
	}
}


void display() {
	glClear(GL_COLOR_BUFFER_BIT);
	glFlush();
}

int main(int argc, char* argv[]) {
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(800, 600);
	glutInitWindowPosition(100, 150);
	glutCreateWindow("SPLINE O CURVA DE BREZIER");
	glClearColor(0, 0, 0, 0);
	glColor3f(1, 0, 0);
	glMatrixMode(GL_PROJECTION);
```

```
        glLoadIdentity();
        gluOrtho2D(0, 800, 0, 600);
        glutMouseFunc(Mouse);
        glutDisplayFunc(display);
        glutMainLoop();
        return 0;
}
```