```
// PRESENTADO POR JOAN SEBASTIAN TIBAQUIRA COD 1202060
#include <iostream>
#include <GL\glut.h>//INCLUYE LA LIBRERIA OPEN GRAPHICS LIBRARY

using namespace std;
void Display() {
	glClear(GL_COLOR_BUFFER_BIT);//BORRAR BUFFER
	glFlush();//VACIA EL BUFFER
}
void Pixel(int x, int y) {//RECIBE LAS COORDENADAS DEL USUARIO
	glBegin(GL_POINTS);// VERTICES COMO PUNTOS  5
	glVertex2f(x, y);//PERMITE VISUALIZAR EL PIXEL
	glEnd();//FINALIZA EL PROGRAMA
	glFlush();
	glPointSize(10);//ADECUA EL TAMAÑO DEL PIXEL
}
void DDisplay() {
	glClearColor(255, 255, 255, 0);
	glColor3f(255, 255, 0);
	gluOrtho2D(-400, 400, -400, 400);
}
void Grafica() {
	int x, y, op, R1, R2, R3;
	cout << "Digite los rangos de las graficas de barras en su respectivo orden " <<
endl;
	cin >> R1;
	cin >> R2;
	cin >> R3;
	if (R1 > 0 && R2 > 0 && R3 > 0)
	{
		for (int x = -300; x < R1; x++)
		{
			glColor3f(255, 0, 0);//Se asigna color a punto
			Pixel(-250, x);
		}
		for (int x = -300; x < R2; x++)
		{
			glColor3f(255,255, 0);//Se asigna color a punto
			Pixel(-200, x);
		}
		for (int x = -300; x < R3; x++)
		{
			glColor3f(0, 0, 255);//Se asigna color a punto
			Pixel(-150, x);
		}
	}
	else
	{
		cout << "Solamente se pueden digitar numeros positivos " << endl;
	}
}
void Plano() {
	for (int x = -300; x < 300; x++) {
		glColor3f(0, 0, 0);
		Pixel(-300, x);
	}
	for (int y = -300; y < 300; y++) {
		glColor3f(0, 0, 0);
```

```
			Pixel(y, -300);
		}

}
void Mouse(int b, int e, int x, int y) {
		if ((e == GLUT_DOWN) && (b == GLUT_RIGHT_BUTTON)) {
			Plano();
			Grafica();
			Pixel(0, 200);
		}
		if ((e == GLUT_DOWN) && (b == GLUT_LEFT_BUTTON)) {
			exit(0);
		}
}
int main(int argc, char** argv) {
		glutInit(&argc, argv);//INICIALIZACION DE PROGRAMAS GLUT
		glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);//INDICA LA UTILIZACION DE DOBLE
BUFFER Y RGB
		glutInitWindowPosition(50, 50);//UBICACION DE LA VENTANA
		glutInitWindowSize(800, 800);//ASIGNA EL TAMAÑO DE LA VENTANA
		glutCreateWindow("PIXEL");//CREACION DE VENTANA Y ASIGNACION DE NOMBRE
		glutMouseFunc(Mouse);
		glutDisplayFunc(Display);//REGISTRA LA FUNCION DEL REDIBUJADO
		DDisplay();
		glutMainLoop();//REALIZA UN BUCLE INFINITO
		system("pause");
}
/*
#include <iostream>
#include <GL\glut.h>//INCLUYE LA LIBRERIA OPEN GRAPHICS LIBRARY
using namespace std;
void dis() {
		glClear(GL_COLOR_BUFFER_BIT);// Se borran el buffer de pantalla
		glFlush();
}
void pintar_pix(int x, int y) {
		glBegin(GL_POINTS);//Se iniciliza el punto
		glVertex2f(x, y);// Se daran las coordenadas y posicion inicial del punto
		glEnd();//Fin al punto
		glFlush();
		glPointSize(10);//Tamaño de Punto
}
void pintar_pantalla() {
		glClearColor(1, 1, 1, 1);//Color Pantalla
		glColor3f(0.f, 0, 1);//Se asigna color a punto
		gluOrtho2D(-400, 400, -400, 400);//Tamaño Pantalla
}
void Pintafuncion() {
		int x, y, op, rango1, rango2, rango3;
		float Apertura = 0, Amplitud = 0, Periodo = 0, radio = 0;
		float a;
		cout << "Digite los rangos de las graficas de barras en su respectivo orden " <<
endl;
		cin >> rango1;
		cin >> rango2;
		cin >> rango3;
		if (rango1 > 0 && rango2 > 0 && rango3 > 0)
		{
```

```
		for (int x = 10; x < rango1; x++)
		{
			glColor3f(0.5, 0, 1);//Se asigna color a punto
			pintar_pix(20, x);
		}
		for (int x = 10; x < rango2; x++)
		{
			glColor3f(1, 0, 0.5);//Se asigna color a punto
			pintar_pix(40, x);
		}
		for (int x = 10; x < rango3; x++)
		{
			glColor3f(0.f, 0, 1);//Se asigna color a punto
			pintar_pix(60, x);
		}
	}
	else
	{
		cout << "Solamente se pueden digitar numeros positivos " << endl;
	}
}
void cartesiano() {
	for (int x = -400; x < 400; x++)
	{
		pintar_pix(0, x);
	}
	for (int y = -400; y < 400; y++)
	{
		pintar_pix(y, 0);
	}


}
void mov_raton(int boton, int estado, int x, int y) {
	if ((estado == GLUT_DOWN) && (boton == GLUT_RIGHT_BUTTON)) {
		cartesiano();
		Pintafuncion();
		pintar_pix(0, 200);

	}
	if ((estado == GLUT_DOWN) && (boton == GLUT_LEFT_BUTTON)) {
		exit(0);
	}
}


int main(int argc, char* args[]) {
	glutInit(&argc, args);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowPosition(50, 50);
	glutInitWindowSize(800, 800);
	glutCreateWindow("NICK PIXEL");
	glutMouseFunc(mov_raton);
	glutDisplayFunc(dis);
	pintar_pantalla();
	glutMainLoop();
	system("pause");
}*/
```