# OpenGL 예제

## 키 입력하여 카메라 위치 변환 실습

```
#define GLUT_DISABLE_ATEXIT_HACK

#include <Windows.h>
#include <gl/GL.h>
#include <gl/glut.h>
#include <math.h>
#include <conio.h> // getch(); 함수를 사용

float eyey = 0, eyex = 6.5, eyez = 10, tr = 0.01;

void drawSphere() {
	glPushMatrix();
	glRotatef(90, 1, 0, 0);
	glutWireSphere(0.1, 10, 10);
	glPopMatrix();
}

void drawTriangle(float size) {
	glBegin(GL_POLYGON);
	glColor3f(1, 1, 0);
	glVertex3f(1 * size, 1 * size, 1 * size);
	glColor3f(0, 1, 0);
	glVertex3f(1 * size, 1 * size, 0);
	glColor3f(0, 0, 1);
	glVertex3f(0, 1 * size, 1 * size);
	glColor3f(0, 0, 1);
	glVertex3f(1 * size, 2 * size, 1 * size);
	glColor3f(0, 0, 1);
	glVertex3f(1 * size, 1 * size, 0);
	glEnd();
}

void drawBox(float w, float h) {
	glPushMatrix();
	glScalef(w, h, w);
	glutWireCube(1);
	glPopMatrix();
}

void drawAxes() {
	glBegin(GL_LINES);

	glColor3f(1, 0, 0);
	glVertex3f(0, 0, 0);
	glVertex3f(1, 0, 0); // x
	glColor3f(0, 1, 0);
	glVertex3f(0, 0, 0);
	glVertex3f(0, 1, 0); // y
	glColor3f(0, 0, 1);
	glVertex3f(0, 0, 0);
	glVertex3f(0, 0, 1); // z
	glEnd();
}

void drawPlane(void) {
	glColor3f(0.7, 0.7, 0.7);
	glBegin(GL_LINES);
	for (int i = 0; i<20; i++) {
		glVertex3f(-10, 0, i - 10);
		glVertex3f(10, 0, i - 10);
	}
	for (int i = 0; i<20; i++) {
		glVertex3f(i - 10, 0, -10);
		glVertex3f(i - 10, 0, 10);
	}
	glEnd();

	glColor3f(0, 0, 0);
	glLineWidth(3);
	glBegin(GL_LINES);
	glVertex3f(-20, 0, 0);
	glVertex3f(20, 0, 0);
	glVertex3f(0, 0, -20);
	glVertex3f(0, 0, 20);
	glEnd();
}

// 키 입력
void  special(int key, int x, int y)
{
        switch (key) {
                //  spin key for image rotation
        case GLUT_KEY_UP:
                eyey += 0.3;
                break;
        case GLUT_KEY_DOWN:
                eyey -= 0.3;
                break;
        case GLUT_KEY_LEFT:
                eyex -= 0.3;
                break;
        case GLUT_KEY_RIGHT:
                eyex += 0.3;
                break;
        default:
                break;
        }
        glutPostRedisplay();
}

void myDisplay() {
        char info[128];

        glClear(GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT);

        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluPerspective(60, 1, 0.1, 100); //-2.0, 2.0, -2.0, 2.0, -1.0, 1.0);

        static float angle = 0.0;
        angle += 0.01;
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        //gluLookAt(3.0*cos(angle), eyey, 3.0*sin(angle), 0, 1.5, 0, 0, 1, 0);
        gluLookAt(eyex, eyey, eyez, 0, 1.5, 0, 0, 1, 0);
        drawPlane();
        glLineWidth(1);
        glColor3f(0.0, 1.0, 1.0);


        for (int i = 0; i < 30; i++){
                tr += 0.002;
                glTranslatef(0.1, 0.1, 0.1);
                glRotatef(tr, tr, tr, 0);
        }

        drawTriangle(1.0);

        static float tAngle;
        tAngle += 0.1;
        float hAngle = sin(tAngle);
        hAngle *= hAngle;
        /*
        static float tAngle;
        tAngle += 0.1;
        float hAngle = sin(tAngle);
        hAngle *= hAngle;
        */

        glutSwapBuffers();
}

int main(int argc, char **argv) {
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH | GLUT_RGBA);
        glutInitWindowPosition(0, 0);
        glutInitWindowSize(512, 512);
        glutCreateWindow("12510096 조광민");

        glEnable(GL_DEPTH_TEST);

        glClearColor(0.0, 0.0, 0.0, 1.0);
        glutSpecialFunc(special);
        glutDisplayFunc(myDisplay);
        glutIdleFunc(myDisplay);
        glutMainLoop();
        return 0;
}
```

**성 만들기**

```
// 성 만들기  //
#define GLUT_DISABLE_ATEXIT_HACK

#include <Windows.h>
#include <gl/GL.h>
#include <gl/glut.h>
#include <math.h>
#include <conio.h> // getch(); 함수를 사용

float eyey = 2, eyex = 0, eyez = 0, tr = 0.01; // 전역 변수
double delay = -1;

void drawTriangle(float size) { // 삼각형 그리기 (사면체 - 면하나 제외)
        glBegin(GL_POLYGON);

        glColor3f(0.5, 0, 1);
        glVertex3f(1 * size, 1 * size, 1 * size);
        glColor3f(0, 0, 1);
        glVertex3f(1 * size, 1 * size, 0);
        glColor3f(0, 0, 1);
        glVertex3f(0, 1 * size, 1 * size);
        glColor3f(0, 0, 1);
        glVertex3f(1 * size, 2 * size, 1 * size);
        glColor3f(0, 0, 1);
        glVertex3f(1 * size, 1 * size, 0);

        glEnd();
}

void drawCircle(float radius, float size){
        glBegin(GL_POLYGON);

        int nPoints=20;
        float angle = 0.0;
        float step=(3.14159*2.0)/nPoints;
        // 반복문 내에서 여러 개의 정점 좌표를 계산한 뒤에 지정하는 방식
        // 여기서는 원을 이루는 정점들을 계산
        while (angle <3.14159*2.0) {
                glVertex3f(radius*cos(angle), size ,radius*sin(angle));
                glVertex3f(radius*cos(angle), -size ,radius*sin(angle));
                glVertex3f(radius*cos(angle+step), -size ,radius*sin(angle+step));
                glVertex3f(radius*cos(angle), size ,radius*sin(angle));
                angle += step;
        }
        glEnd();

        glBegin(GL_QUAD_STRIP);



        glEnd();
}

void drawRactangle(float xscale, float yscale, float zscale){
        glBegin(GL_QUADS);

        // 앞부분
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        // 뒷부분
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(xscale,yscale,-zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 윗부분
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(xscale,yscale,-zscale);

        // 아래 부분
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 왼쪽
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 오른쪽
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(xscale,yscale,-zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glEnd();

        /*
        glPushMatrix();
        glScalef(0.3, 0.6, 0.3);
        glutWireCube(1);
        glPopMatrix();
        */
}

void drawPlane(void) { // 바닥 타일 생성
        glColor4f(1, 1, 1, 0.1);
        glBegin(GL_LINES);
        for (int i = 0; i<=20; i++) {
                glVertex3f(-10, 0, i - 10);
                glVertex3f(10, 0, i - 10);
        }
        for (int i = 0; i<=20; i++) {
                glVertex3f(i - 10, 0, -10);
                glVertex3f(i - 10, 0, 10);
        }
        glEnd();
}

// 키 입력
void  keyboard(unsigned char key, int x, int y)
{
        int    mod;

        switch (key) {
        case 'z':
                delay *= -1;
                break;
        }
        glutPostRedisplay();
}

void  special(int key, int x, int y)
{
        switch (key) {
        case GLUT_KEY_UP:
                eyey += 0.3;
                break;
        case GLUT_KEY_DOWN:
                eyey -= 0.3;
                break;
        case GLUT_KEY_LEFT:
                eyex += 0.05;
                break;
        case GLUT_KEY_RIGHT:
                eyex -= 0.05;
                break;
        default:
                break;
        }
        glutPostRedisplay();
}

void drawWall(){
        for (int i= -5.0; i<15; i+=10){
                for (int j= -5.0; j<15; j+=10){
                        glPushMatrix();
                                glColor4f(1, 1, 1, 1);
                                //glRotatef(45, 0, 0, 1); // 로테이션 각도, x, y, z축 지정
                                glTranslatef(i, 0.5, j);
                                // -5   -2.5   0   2.5   5
                                drawRactangle(0.5, 0.5, 0.5);
                        glPopMatrix();

                        glPushMatrix();
                                glColor4f(0, 1, 0, 1);
                                glTranslatef(i, 1.5, j);
                                // -5   -2.5   0   2.5   5
                                drawCircle(0.4, 1.5);
                        glPopMatrix();
                }
        }
}

void myDisplay() {
        glClear(GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT);

        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glOrtho(-10, 10, -10, 10, -10, 10);
        gluPerspective(0, 1, 0.1, 2000);
        // 시야각, 종횡비, 전방절단면, 후방절단면
         // 카메라의 상을 맺는 최소 거리와 최대 거리를 정해 입체감 있게 만듬
        glMatrixMode(GL_MODELVIEW); //
        glLoadIdentity();

        gluLookAt(-3.0*cos(eyex), eyey, -3.0*sin(eyex), 0, 1.5, 0, 0, 1, 0); //카메라 회전

        glPushMatrix();
        drawPlane();
        glLineWidth(1);
        glPopMatrix();

        glPushMatrix(); // Begin~End와 달리 push~pop은 한 단락으로 적용시킨다.

        //glRotatef(tr * 2, 0, tr * 2, 0);
        if (delay == 1){
                // 바닥 타일
                glPushMatrix();
                glColor4f(1 ,0, 1, 1);
                glTranslatef(0, -0.5, 0);
                drawRactangle(5, 0.1, 5); // x, y, z 넓이
```

```
			glPopMatrix();

			// 첫번째 칸 정사각형 기둥
			drawWall();

			glPushMatrix();
			glTranslatef(0, 3, 0);
			drawWall();
			glPopMatrix();

			/*
			// 두번째 칸 원기둥
			glPushMatrix();
			glColor4f(1, 0, 0, 1);
			glTranslatef(0, 2, 3);
			drawCircle(0.3, 2);
			glPopMatrix();

			glPushMatrix();
			glColor4f(1, 0, 0, 1);
			glTranslatef(0, 2, -3);
			drawCircle(0.3, 2);
			glPopMatrix();
			*/
		}else {
			drawTriangle(0.5);
		}

		glPopMatrix();


		for (int i = 0; i < 30; i++){
			tr += 0.05;
		}

		glutSwapBuffers();
}

int main(int argc, char **argv) {
		glutInit(&argc, argv);
		glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH | GLUT_RGBA);
		glutInitWindowPosition(0, 0);
		glutInitWindowSize(512, 512);
		glutCreateWindow("12510096 조광민");

		glEnable(GL_DEPTH_TEST);

		glClearColor(0.0, 0.0, 0.0, 1.0);
		glutKeyboardFunc(keyboard);
		glutSpecialFunc(special);
		glutDisplayFunc(myDisplay);
		glutIdleFunc(myDisplay);
		glutMainLoop();

		return 0;
}
```

## 3D 핑퐁

```
#define GLUT_DISABLE_ATEXIT_HACK


#include <Windows.h>
#include <gl/GL.h>
#include <gl/glut.h>
#include <math.h>
```

```
#include <conio.h> // getch(); 함O수ùo를ㄱ 사íc용첩

//GLfloat ax, ay, az;
GLdouble angle = 0.0; // 회¸전u 각퉤Ë
GLfloat cx, cy, cz; // 클¬릭¬? 좌A표¥
GLfloat ca; // 클¬릭¬? 앵奚-글펢
// 마БÒ우칮스翠¬ 이I동醫¯량ㄲç 배öe율²

GLfloat red[] = { 0.8, 0.2, 0.2, 1.0 };
GLfloat pos[] = { 3.0, 4.0, 5.0, 1.0 };
GLdouble ex = 0.0, ey = 0.0, ez = 10.0;
GLdouble tx = 0.0, ty = 0.0, tz = 0.0;
GLdouble ax = 0.0, ay = 1.0, az; // 회¸전u 축a

double sx, sy;
#define SCALE 360.0

////////////////////////////////////
float mvx= 0, mvz = 0;
float ballx = 0, bally = 0, ballz = 0;
float xc = 1, yc = 1, zc = 1;
float viewx = 0, viewy = 0, viewz = 0;

void drawRactangle(float xscale, float yscale, float zscale){
        /// 선品¾ 생íy성彼¬

        glColor3f(1, 0, 0);
        // 앞úO부첼분Ж¢
        glBegin(GL_LINE_LOOP);
        glVertex3f(-xscale, -yscale, zscale);
        glVertex3f(xscale, -yscale, zscale);
        glVertex3f(xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, zscale);
        glEnd();
        // 뒷夷-부첼분Ж¢
        glBegin(GL_LINE_LOOP);
        glVertex3f(-xscale, yscale, -zscale);
        glVertex3f(xscale, yscale, -zscale);
        glVertex3f(xscale, -yscale, -zscale);
        glVertex3f(-xscale, -yscale, -zscale);
        glEnd();
        // 윗-부첼분Ж¢
        glBegin(GL_LINE_LOOP);
        glVertex3f(xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, -zscale);
        glVertex3f(xscale, yscale, -zscale);
        glEnd();
        // 아奚¡래ㄲ® 부첼분Ж¢
        glBegin(GL_LINE_LOOP);
        glVertex3f(-xscale, -yscale, zscale);
        glVertex3f(xscale, -yscale, zscale);
        glVertex3f(xscale, -yscale, -zscale);
        glVertex3f(-xscale, -yscale, -zscale);
        glEnd();
        // 왼¯Þ쪽E
        glBegin(GL_LINE_LOOP);
        glVertex3f(-xscale, -yscale, zscale);
        glVertex3f(-xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, -zscale);
        glVertex3f(-xscale, -yscale, -zscale);
        glEnd();
        // 오츎른AÍ쪽E
        glBegin(GL_LINE_LOOP);
        glVertex3f(xscale, -yscale, zscale);
        glVertex3f(xscale, yscale, zscale);
        glVertex3f(xscale, yscale, -zscale);
        glVertex3f(xscale, -yscale, -zscale);

        glEnd();
}

void drawBall(){
        glBegin(GL_LINE_STRIP);

        /*
        int nPoints = 20;
        float angle = 0.0;
        float step = (3.14159*2.0) / nPoints;
        glColor3f(1, 1, 0);
        while (angle <3.14159*2.0) {
                glVertex3f(0.05*cos(angle), 0.05*sin(angle), 0);
                angle += step;
        }*/

        glutWireSphere(0.3, 20, 20);
        glEnd();
}

void drawPlayer(float xscale, float yscale, float zscale){
        /*glBegin(GL_QUADS);

        glColor3f(0, 0, 1);
        glVertex3f(0.1, 0.01, 0);
        glVertex3f(-0.1, 0.01, 0);
        glVertex3f(-0.1, -0.01, 0);
        glVertex3f(0.1, -0.01, 0);

        glEnd();*/
        glBegin(GL_QUADS);

        // 앞úO부쳏분Ж¢
        glVertex3f(-xscale, -yscale, zscale);
        glVertex3f(xscale, -yscale, zscale);
        glVertex3f(xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, zscale);

        // 뒷夷-부췗분Ӂ¢
        glVertex3f(-xscale, yscale, -zscale);
        glVertex3f(xscale, yscale, -zscale);
        glVertex3f(xscale, -yscale, -zscale);
        glVertex3f(-xscale, -yscale, -zscale);

        // 윗-부췗분Ӂ¢
        glVertex3f(xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, -zscale);
        glVertex3f(xscale, yscale, -zscale);

        // 아奚¡래ㄲ® 부췗분Ӂ¢
        glVertex3f(-xscale, -yscale, zscale);
        glVertex3f(xscale, -yscale, zscale);
        glVertex3f(xscale, -yscale, -zscale);
        glVertex3f(-xscale, -yscale, -zscale);

        // 윈¯Þ쪽E
        glVertex3f(-xscale, -yscale, zscale);
        glVertex3f(-xscale, yscale, zscale);
        glVertex3f(-xscale, yscale, -zscale);
        glVertex3f(-xscale, -yscale, -zscale);

        // 오츐른AÍ쪽E
        glVertex3f(xscale, -yscale, zscale);
        glVertex3f(xscale, yscale, zscale);
        glVertex3f(xscale, yscale, -zscale);
        glVertex3f(xscale, -yscale, -zscale);
        glEnd();
}

// 마БÒ우칮스쿓¬ 입O력짝
/*
void mouse(int x, int y){

}*/

void resize(int w, int h){
        // 마БÒ우칮스쿓¬ 포  À인I터I 위§치¡ 윈©도伊ì우칮의C 상íο대쩬적u인I 위§치¡에¯¡ 환?율² 용칤
        sx = 1.0 / (double)w;
        sy = 1.0 / (double)h;
}

void motion(int x, int y){
        double dx, dy, a;
        // 마БÒ우칮스쿓ㄱ 포  À인I터I 위§치¡의C 끌炭ª기짏 시öA작U 위§치¡에¯¡서品©의C 변¬?위§
        dx = (x - cx) * sx;
        dy = (y - cy) * sy;

        // 마БÒ우칮스쿓ㄱ 포  À인I터I 위§치¡의C 끌炭ª기짏 시öA작U 위§치¡에¯¡서品©의C 거힍리Бç
        a = sqrt(dx * dx + dy * dy);

        if (a != 0.0){
                // 거힍리Бç를ㄱ 각퉤Ë도伊ì로쫣 환?산íe하I여¯© 드ìa래ㄲ®그씔¿ 시öA작U시öA의C 회¸
전u 각퉤Ë에¯¡ 가튀®산íe
                angle = fmod(ca + SCALE * a, 360.0);

                // 마БÒ우칮스쿓ㄱ 포  À인I터I의C 변¬?위§에¯¡서品© 회¸전u축a 벡Б´터I를ㄱ 요칣청≫
                ax = dy / a;
                ay = dx / a;
                az = 0.0;

                // 도伊ì형u의C 재c 묘ö이화-
                glutPostRedisplay();
        }
}

void mouse(int button, int direction, int x, int y)
{
        switch (direction){
        case GLUT_DOWN :
                // 마БÒ우칮스쿓ㄱ 버öo튼  Æ을≫ 누vïïÏ른AÍ 위§치¡를ㄱ 기짏록쫣
                cx = x;
                cy = y;
                // 표¥시öA하I고Æi 있O는쨏 물贅Æ체¼의C 회¸전u 각퉤Ë을≫ 기짏록쫣
                ca = angle;
                break;
        default :
                break;
        }
}
// 키° 입O력쫚
void keyboard(unsigned char key, int x, int y)
{
        switch (key) {
        case 'a':
                mvx -= 0.05;
                if (mvx <= -0.8){
                        mvx = -0.8;
                }
                break;
        case 'd':
                mvx += 0.05;
                if (mvx >= 0.8){
                        mvx = 0.8;
                }
                break;
        case 'w':
                mvz -= 0.05;
                if (mvz <= -0.8){
                        mvz = -0.8;
                }
                break;
        case 's':
                mvz += 0.05;
                if (mvz >= 0.8){
                        mvz = 0.8;
                }
                break;
        }
        glutPostRedisplay();
}

void specialkey(int key, int x, int y)
{
        switch (key) {
        case GLUT_KEY_LEFT:
                viewx -= 0.1;
                break;
        case GLUT_KEY_RIGHT:
                viewx += 0.1;
                break;
        case GLUT_KEY_UP:
                viewy += 0.1;
                break;
        case GLUT_KEY_DOWN:
                viewy -= 0.1;
                break;
        }
        glutPostRedisplay();
}

void myDisplay() {
        glClear(GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glOrtho(-1, 1, -1, 1, -1, 1);
        gluPerspective(30, 1, 0.1, 100);

        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        //gluLookAt(-3.0*cos(viewx), viewy, -3.0*sin(viewx), 0, 0, 0, 0, 1, 0);
        gluLookAt(ex, ey, ez, tx, ty, tz, 0.0, 1.0, 0.0);
        glRotated(angle, ax, ay, 0.0);

        glPushMatrix();
        //\\glTranslatef(ballx, bally, ballz);
        glColor3f(0, 1, 0);
        glutWireSphere(0.1, 15, 15);
        //drawBall();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(mvx, -0.8, mvz);
        glColor3f(0, 0, 1);
        drawPlayer(0.3, 0.01, 0.3);
        glPopMatrix();

        // 벽Бç 충æ돌姨ö 이I벤AÍ트、ç
        if (ballx >= 0.8 || ballx <= -0.8){ xc *= -1; }
        if (bally >= 0.8 || bally <= -0.8){ yc *= -1; }
        if (ballz >= 0.8 || ballz <= -0.8){ zc *= -1; }
        if (ballx > mvx-0.3 && ballx < mvx+0.3 && ballz > mvz-0.3 && ballz < mvz+0.3 && bally
<= -0.6){ yc *= -1; }

        ballx += 0.01 * xc;
        bally += 0.008 * yc;
        ballz += 0.003 * zc;

        drawRactangle(0.9, 0.9, 0.9);
        glPushMatrix();
        glColor3f(1,1,0);
        glPopMatrix();

        glutSwapBuffers();
}

int main(int argc, char **argv) {
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH | GLUT_RGBA);
        glutInitWindowPosition(0, 0);
        glutInitWindowSize(512, 512);
        glutCreateWindow("12510096 조¶광쐽´민öI");

        glEnable(GL_DEPTH_TEST);

        glClearColor(0.0, 0.0, 0.0, 1.0);
        glutKeyboardFunc(keyboard);
        glutSpecialFunc(specialkey);

        glutDisplayFunc(myDisplay);
        glutIdleFunc(myDisplay);
        glutReshapeFunc(resize);

        glutMouseFunc(mouse); // 마БÒ우칮스쿯¬ 클¬릭¬? 시öA 발聚¬생íy되iC는쨏 이I벤AÍ트、ç
        glutMotionFunc(motion); // 마БÒ우칮스쿯¬ 클¬릭¬? 후A 이I동醫¯ 시öA 발聚¬생íy되iC는쨏
이I벤AÍ트、ç
```

```
        //glutPassiveMotionFunc(); // 마БÒ우칮스翠¬ 클¬릭¬? 안úE하I고Æi 이I동醫¯ 시öA 발聚¬생
íy되iC는쨏 이I벤AÍ트、ç

        glutMainLoop();

        return 0;
}
```

**자동차 줌인 / 아웃**

```
// 자동차 줌인 / 아웃, 마우스 아직 안함 //

#define GLUT_DISABLE_ATEXIT_HACK

#include <Windows.h>
#include <gl/GL.h>
#include <gl/glut.h>
#include <math.h>
#include <conio.h> // getch(); 함O수ùo를¬ 사íc용칇

float eyey = 2, eyex = 0, eyez = 0, tr = 0.01; // 전u역¯ª 변¬?수ùo
double delay = -1;
float range = 1.0;
float aspRatio = 1.0;

void drawTriangle(float size) { // 삼íi각퉤Ë형u 그씰¿리Бç기퐀 (사íc면촦체¼ - 면촦하I나敗£ 제|외칄)
        glBegin(GL_POLYGON);

        glColor3f(0.5, 0, 1);
        glVertex3f(1 * size, 1 * size, 1 * size);
        glColor3f(0, 0, 1);
        glVertex3f(1 * size, 1 * size, 0);
        glColor3f(0, 0, 1);
        glVertex3f(0, 1 * size, 1 * size);
        glColor3f(0, 0, 1);
        glVertex3f(1 * size, 2 * size, 1 * size);
        glColor3f(0, 0, 1);
        glVertex3f(1 * size, 1 * size, 0);

        glEnd();
}
// 원¯ø기퐀둥iO
void drawCircle(float radius, float size){
        glBegin(GL_POLYGON);

        int nPoints=20;
        float angle = 0.0;
        float step=(3.14159*2.0)/nPoints;
        // 반öY복3ö문取ç 내阪í에¯¡서品© 여¯©러¤? 개튜ø의C 정¤점¡ 좌A표￥를¬ 계Æe산íe한N 뒤iU
에¯¡ 지o정¤하I는쨏 방聚¡식öA
        // 여¯©기퐀서品©는쨏 원¯ø을≫ 이I루�는쨏 정¤점¡들ìe을≫ 계Æe산íe
        while (angle <3.14159*2.0) {
                glVertex3f(radius*cos(angle), size ,radius*sin(angle));
                glVertex3f(radius*cos(angle), -size ,radius*sin(angle));
                glVertex3f(radius*cos(angle+step), -size ,radius*sin(angle+step));
                glVertex3f(radius*cos(angle), size ,radius*sin(angle));
                angle += step;
        }
        glEnd();
}

void drawCirclel(float setradius, float x, float y, float z){
        glBegin(GL_POLYGON);
        int Points = 40;
        float radius = setradius;
        glColor3f(0, 0, 1);
        float angle = 0.0;
        float step = (3.14159*2.0) / Points;
        while (angle <3.14159*2.0) {
                glVertex3f(x, radius*sin(angle) + y, radius*cos(angle) + z);
                // cos, sin에¯¡ 크ⓒ기끯 비촱율²을≫ 곱Æo해Ø줌U
                angle += step;
        }
        glEnd();
}

// 바öU퀴u
void drawtire(float radius, float size, float msize){
        //glBegin(GL_POLYGON);
        glBegin(GL_LINE_STRIP);
        int nPoints=20;
        float angle = 0.0;
        float step=(3.14159*2.0)/nPoints;
        // 반öY복3ö문取ç 내阪í에¯¡서品© 여¯ⓒ러¤? 개튜ø의C 정¤점¡ 좌A표￥를ㄱ 계Æe산íe한N 뒤iU
에¯¡ 지o정¤하I는쨏 방聚¡식öA
        // 여¯ⓒ기끯서品©는쨏 원¯ø을≫ 이I루쨻는쨏 정¤점¡들ìe을≫ 계Æe산íe
        while (angle <3.14159*2.0) {
                glVertex3f(size, radius*cos(angle), radius*sin(angle));
                glVertex3f(msize, radius*cos(angle), radius*sin(angle));
                glVertex3f(msize, radius*cos(angle+step), radius*sin(angle+step));
                glVertex3f(size, radius*cos(angle), radius*sin(angle));
                angle += step;
        }
        glEnd();

        glBegin(GL_LINE_STRIP);
        glColor3f(0, 1, 0);
        nPoints=20;
        angle = 0.0;
        step=(3.14159*2.0)/nPoints;
        while (angle <3.14159*2.0) {
                glVertex3f(size, radius*cos(angle), radius*sin(angle));
                angle += step;
        }
        glEnd();

        glBegin(GL_LINE_STRIP);
        glColor3f(0, 0, 1);
        nPoints=20;
        angle = 0.0;
        step=(3.14159*2.0)/nPoints;
        while (angle <3.14159*2.0) {
                glVertex3f(msize, radius*cos(angle+step), radius*sin(angle+step));
                angle += step;
        }
        glEnd();
}

// 지o붕Ꮶª
void drawtop(float radius, float size, float msize){
        glBegin(GL_POLYGON);

        int nPoints=20;
        float angle = 0.0;
        float zangle = 0.0;
        float step=(3.14159*2.0)/nPoints;
        while (zangle <3.14159*2.0) {
                glVertex3f(size, radius*sin(zangle), radius*cos(zangle));
                while (angle < 3.14159*2.0) {
                        glVertex3f(size, radius*cos(angle), radius*sin(angle));
                }
                angle += step;
        }
        glEnd();
}

void drawRactangle(float xscale, float yscale, float zscale){
        glBegin(GL_QUADS);
        // 앒úO부첼분Ꮶ¢
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);

        // 뒷夷-부첼분Ꮶ¢
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(xscale,yscale,-zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 윗-부첼분Ꮶ¢
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(xscale,yscale,-zscale);

        // 아奚¡래ㄲ® 부첼분Ꮶ¢
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 왼¯Þ쪽E
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 오흁른AÍ쪽E
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(xscale,yscale,-zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glEnd();

        /// 선品¾ 생íy성彼¬
        glBegin(GL_LINE_STRIP);
        glColor3f(1,0,0);
        // 앞úO부쳏분Ж¢
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);

        // 뒷夷-부쳏분Ж¢
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(xscale,yscale,-zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 윗-부쳏분Ж¢
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(xscale,yscale,-zscale);

        // 아奚¡래ㄲ® 부쳏분Ж¢
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 왼¯Þ쪽E
        glVertex3f(-xscale,-yscale,zscale);
        glVertex3f(-xscale,yscale,zscale);
        glVertex3f(-xscale,yscale,-zscale);
        glVertex3f(-xscale,-yscale,-zscale);

        // 오흁른AÍ쪽E
        glVertex3f(xscale,-yscale,zscale);
        glVertex3f(xscale,yscale,zscale);
        glVertex3f(xscale,yscale,-zscale);
        glVertex3f(xscale,-yscale,-zscale);
        glEnd();
}

void drawPlane(void) { // 바öU닥쩍 타¸일I 생íy성彼¬
```

```
        glColor4f(1, 1, 1, 0.1);
        glBegin(GL_LINES);
        for (int i = 0; i<=20; i++) {
                glVertex3f(-10, 0, i - 10);
                glVertex3f(10, 0, i - 10);
        }
        for (int i = 0; i<=20; i++) {
                glVertex3f(i - 10, 0, -10);
                glVertex3f(i - 10, 0, 10);
        }
        glEnd();
}

void SetCamera(){
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glOrtho(-aspRatio*range,aspRatio*range,-range,range,-10,10);
        gluPerspective(0, 1, 0.1, 2000);
}

void reshape(int w, int h){
        aspRatio = float(w)/h;
        SetCamera();
        glViewport(0, 0, w, h);
}

// 마БÒ우칯스쿐¬ 입O력짞
void  mouse(unsigned char mb, int x, int y)
{
        switch (mb) {
        case 'z':
                delay *= -1;
                break;
        case 'w':
                range *= 0.9;
                break;
        case 's':
                range *= 1.1;
                break;
        }
        SetCamera();
        glutPostRedisplay();
}


// 키° 입O력짞
void  keyboard(unsigned char key, int x, int y)
{
        int    mod;

        switch (key) {
        case 'z':
                delay *= -1;
                break;
        case 'w':
                range *= 0.9;
                break;
        case 's':
                range *= 1.1;
                break;
        }
        SetCamera();
        glutPostRedisplay();
}

void  special(int key, int x, int y)
{
        switch (key) {
        case GLUT_KEY_UP:
                eyey += 0.3;
                break;
        case GLUT_KEY_DOWN:
                eyey -= 0.3;
                break;
        case GLUT_KEY_LEFT:
                eyex += 0.05;
                break;
        case GLUT_KEY_RIGHT:
                eyex -= 0.05;
                break;
        default:
                break;
        }
        glutPostRedisplay();
}

void drawWall(){
        for (int i= -5.0; i<15; i+=10){
                for (int j= -5.0; j<15; j+=10){
                        glPushMatrix();
                                glColor4f(1, 1, 1, 1);
                                //glRotatef(45, 0, 0, 1); // 로짧테×이I션ùC 각퉤Ë도伊ì, x, y, z축a
지ㅇ정ㅁ
                                glTranslatef(i, 0.5, j);
                                // -5   -2.5   0   2.5   5
                                drawRactangle(0.5, 0.5, 0.5);
                        glPopMatrix();

                        glPushMatrix();
                                glColor4f(0, 1, 0, 1);
                                glTranslatef(i, 1.5, j);
                                // -5   -2.5   0   2.5   5
                                drawCircle(0.4, 1.5);
                        glPopMatrix();
                }
        }
}
```

```
void myDisplay() {
	glClear(GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT);
	/*
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	glOrtho(-10, 10, -10, 10, -10, 10);
	gluPerspective(0, 1, 0.1, 2000);*/
	// 시öA야孩¬각퉤Ë, 종¾횡¾비촱, 전u방聚¡절y단쩍면촦, 후A방聚¡절y단쩍면촦
	 // 카≪메Ж-라Òo의C 상ío을≫ 맺¼는쨏 최O소ùO 거힍리Бç와측 최O대쩬 거힍리Бç를ㄱ 정¤해Ø
입O체¼감퉤§ 있O게힉 만Б¬듬ìe

	glMatrixMode(GL_MODELVIEW); //
	glLoadIdentity();

	gluLookAt(-3.0*cos(eyex), eyey, -3.0*sin(eyex), 0, 1.5, 0, 0, 1, 0); //카≪메Ж-라Òo 회¸전u

	glPushMatrix();
	drawPlane();
	glLineWidth(1);
	glPopMatrix();

	glPushMatrix(); // Begin~End와측 달ⅷ-리Бç push~pop은º 한N 단쩍락Òo으¸로짧 적u용칤시öA
킨²다쩍.

	//glRotatef(tr * 2, 0, tr * 2, 0);
	if (delay == 1){
		// 자U동醫¯차÷ 몸촮통e
		glPushMatrix();
		//drawWall();
		glColor3f(1,1,1);
		glTranslatef(0, 0.8, 0.5);
		drawRactangle(2, 0.6, 3.5);
		glColor3f(1,1,1);
		glTranslatef(0, 1.41, 0.5);
		drawRactangle(2, 0.8, 2);
		glPopMatrix();

		// 바öU퀴u
		glPushMatrix();
		glColor3f(1,0,0);
		glTranslatef(1.7, 0 , -1);
		drawtire(1, 0.4, -0.5);
		glPopMatrix();

		glPushMatrix();
		glColor3f(1,0,0);
		glTranslatef(1.7, 0 , 2.3);
		drawtire(1, 0.4, -0.5);
		glPopMatrix();

		glPushMatrix();
		glColor3f(1,0,0);
		glTranslatef(-1.7, 0 , -1);
		drawtire(1,-0.4, 0.5);
```

```
                glPopMatrix();

                glPushMatrix();
                glColor3f(1,0,0);
                glTranslatef(-1.7, 0 , 2.3);
                drawtire(1, -0.4, 0.5);
                glPopMatrix();



                // 지ㅇ붕Жª 원¯ø

                glPushMatrix();
                glColor3f(0, 0, 1);
                glRotatef(45, 0, 0, 1);
                glTranslatef(1, 0.5, 0);
                drawCirclel(2, 2, 1.7, 1);
                glPopMatrix();
                /*
                glPushMatrix();
                glColor3f(0, 0, 1);
                glTranslatef(0, 3, 1);
                glutSolidSphere(2, 10, 10);
                glPopMatrix();
                */

        }else {
                drawTriangle(0.5);
        }

        glPopMatrix();


        for (int i = 0; i < 30; i++){
                tr += 0.05;
        }

        glutSwapBuffers();
}

int main(int argc, char **argv) {
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH | GLUT_RGBA);
        glutInitWindowPosition(0, 0);
        glutInitWindowSize(512, 512);
        glutCreateWindow("12510096 조¶광씸´민öI");

        glEnable(GL_DEPTH_TEST);

        glClearColor(0.0, 0.0, 0.0, 1.0);
        glutKeyboardFunc(keyboard);
        glutSpecialFunc(special);
        glutDisplayFunc(myDisplay);
        //glutIdleFunc(myDisplay);
```

```
            glutReshapeFunc(reshape);
            glutMainLoop();

            return 0;
}
```