```cpp
#include "gwm.h"

int gWidth = 0;
int gHeight = 0;

void main (void)
{
    // declarations
    void keyboardFunc(unsigned int);
    void displayFunc(void);
    void reshapeFunc(int, int);

    // code
    gwmKeyboardCallback(keyboardFunc);
    gwmDisplayCallback(displayFunc);
    gwmReshapeCallback(reshapeFunc);

    gwmCreateWindow("Viewports", 100, 100, 800, 600);
    gwmEventLoop();
}

void keyboardFunc(unsigned int key)
{
    switch(key)
    {
        // VK_ESCAPE
        case 0x1B:
            gwmExitEventLoop();
            break;

        case '0':
        case VK_NUMPAD0:
            glViewport(0, 0, gWidth, gHeight);
            break;

        case '1':
        case VK_NUMPAD1:
            glViewport(0, 0, gWidth / 2.0f, gHeight / 2.0f);
            break;

        case '2':
        case VK_NUMPAD2:
            glViewport(gWidth / 2, 0, gWidth / 2.0f, gHeight / 2.0f);
            break;

        case '3':
        case VK_NUMPAD3:
            glViewport(gWidth / 2, gHeight / 2, gWidth / 2.0f, gHeight / 2.0f);
            break;

        case '4':
        case VK_NUMPAD4:
            glViewport(0, gHeight / 2, gWidth / 2.0f, gHeight / 2.0f);
            break;

        case '5':
        case VK_NUMPAD5:
            glViewport(0, 0, gWidth / 2.0f, gHeight);
            break;

        case '6':
        case VK_NUMPAD6:
            glViewport(gWidth / 2, 0.0f, gWidth / 2.0f, gHeight);
            break;

        case '7':
        case VK_NUMPAD7:
            glViewport(0, gHeight / 2, gWidth, gHeight / 2.0f);
            break;

        case '8':
        case VK_NUMPAD8:
            glViewport(0, 0, gWidth, gHeight / 2.0f);
            break;

        case '9':
        case VK_NUMPAD9:
            glViewport(gWidth / 4, gHeight / 4, gWidth / 2.0f, gHeight / 2.0f);
            break;
    }
}

void reshapeFunc(int width, int height)
{
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    gWidth = width;
    gHeight = height;

    glViewport(0, 0, gWidth, gHeight);
    gluPerspective(45.0, (float)width / (float)height, 0.1f, 100.0f);
}

void displayFunc(void)
{
    // code
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glTranslatef(0.0f, 0.0f, -3.0f);

    glBegin(GL_TRIANGLES);

    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex2f(0.0f, 1.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex2f(-1.0f, -1.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex2f(1.0f, -1.0f);

    glEnd();

    gwmSwapBuffers();
}


```