OPENGL

Preparing 1/2

- environment : Microsoft Visual C++ 6.0、Microsoft Visual C++ .Net
- Also need : GLUT
  http://www.xmission.com/~nate/glut.html

Preparing 2/2

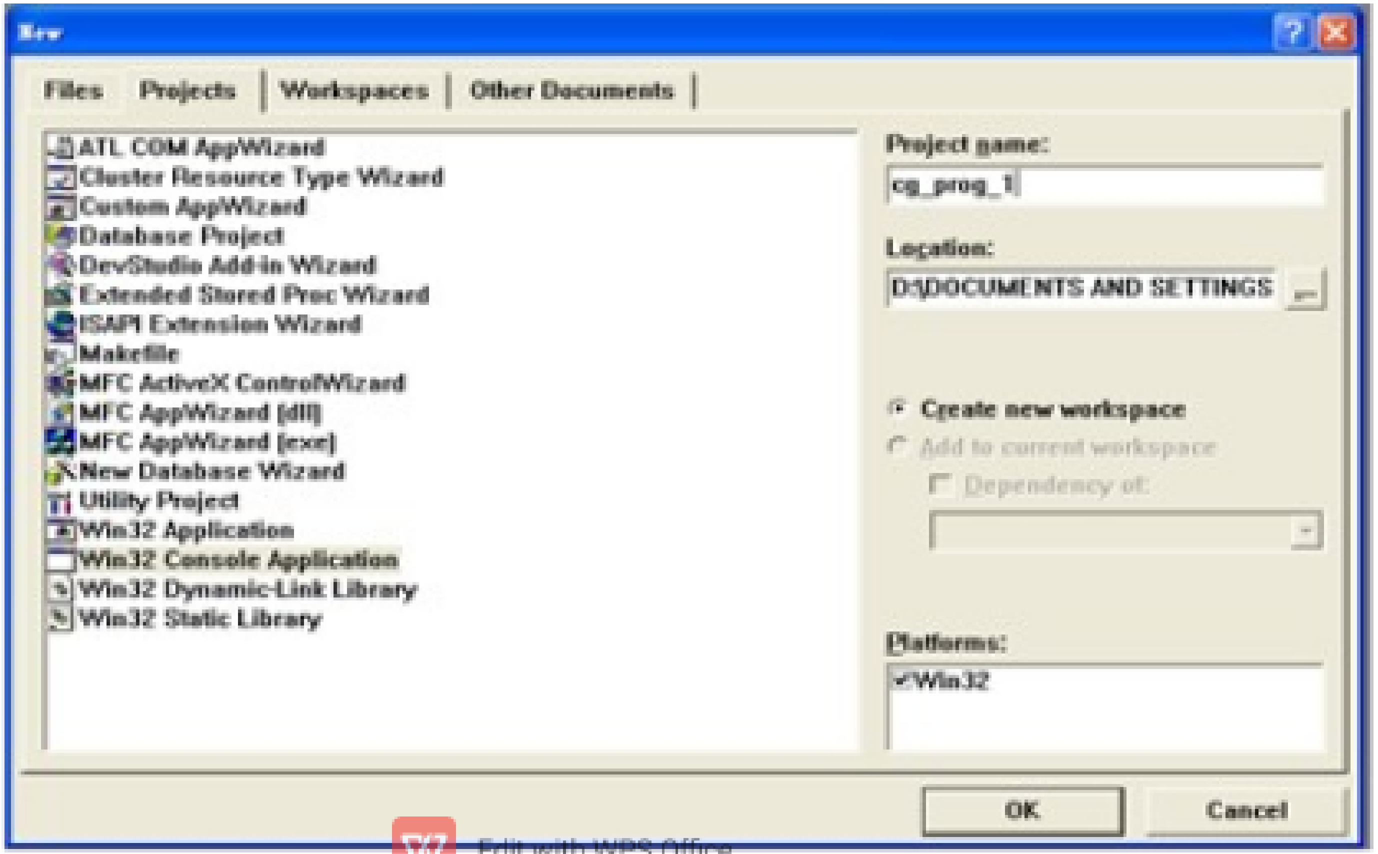
- On Microsoft Visual C++ 6.0
  - Put glut.h into <MSVC>/include/GL/
  - Put glut.lib into <MSVC>/lib/
  - Put glut32.dll into <window>/System32/
- On Microsoft Visual C++ .Net
  - Put glut.h into <MSVC>/platformSDK/include/GL/
  - Put glut.lib into <MSVC>/platformSDK/lib/
  - Put glut32.dll into <window>/System32/

OpenGL Utility Toolkit (GLUT)

- A window system-independent toolkit to hide the complexity of differing window system APIs.
- Providing following operations:
  - Initializing and creating window
  - Handling window and input events
  - Drawing basic 3D objects
  - Running the program
- Use the prefix of **glut** (ex: glutCreateWindow)

Write an OpenGL Program

- Microsoft Visual C++ 6.0
  - Step 1: create a Win32 Console Application project



Project name: cg_prog_1

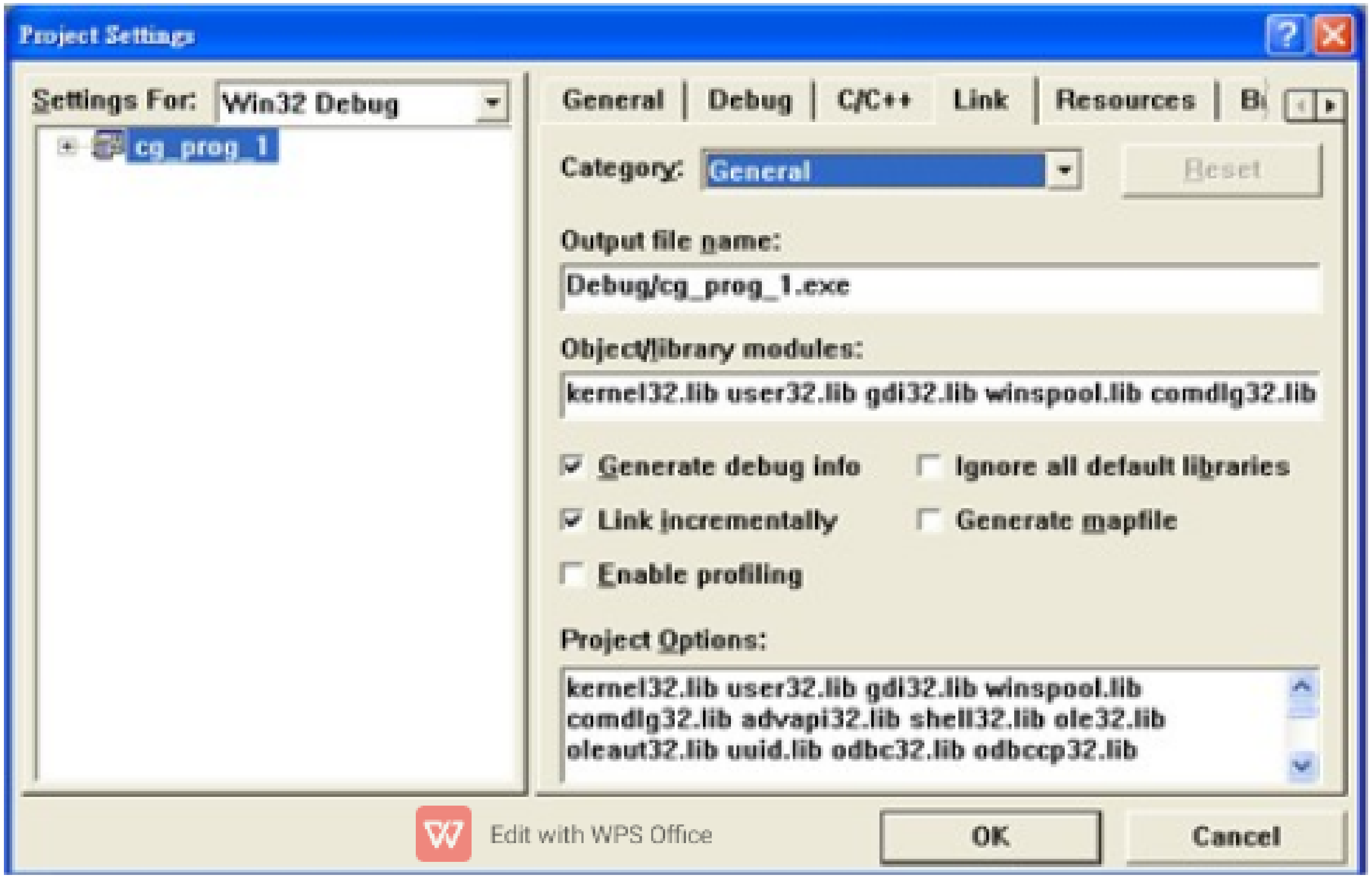
Location: D:\DOCUMENTS AND SETTINGS

[✓ Create new workspace]

Platforms: ✓Win32

Write an OpenGL Program

- Step 2:Press Alt-F7 , brings "Project Settings" , select "Link"



Category: General

Output file name:
Debug/cg_prog_1.exe

Object/library modules:
kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib

[✓] Generate debug info
[ ] Ignore all default libraries
[✓] Link incrementally
[ ] Generate mapfile
[ ] Enable profiling

Project Options:

```
kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib
oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
```

Write an OpenGL Program

- Step 3:add opengl32.lib glu32.lib glut32.lib into Object/library modules
- Step 4:write your code
- Step 5:compile

```
#include "windows.h"
#include "gl/Gl.h"
#include "gl/glut.h"
```

1

```
void mydisplay()
{           glClear(GL_COLOR_BUFFER_BIT);
            glBegin(GL_LINES);
                        glVertex2i(0,110);
                        glVertex2i(45,110);
            glEnd();
            glFlush();
}
```

2

```
void MyInit()
{glClearColor(7.86, 6.98, 8.0, 10.0);
glColor3f(1.0, 0.0, 0.0);
glPointSize(20.0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0, 200.0, 0.0,
200.0,2000);
}
```

3

```
int main(int argc, char* argv[])
{           glutInit(&argc, argv);
            glutInitDisplayMode(GLUT_SINGL
E|GLUT_RGB);
            glutInitWindowSize(500,400,100);
            glutInitWindowPosition(100, 150);
            glutCreateWindow("my Program");
            glutDisplayFunc(mydsplay);
            MyInit();
}
```

4

```
#include "windows.h"
#include "gl/Gl.h"
#include "gl/glut.h"
```

```
void mydisplay()
{     glBegin(GL_LINES);
            glVertex2i(0,110);
            glVertex2i(45,110);
            glVertex2i(0,110);
            glVertex2i(45,110);
            glEnd();
      glFlush();
}
```

```
void MyInit()
{glClearColor(1.0, 0.0,0.0, 1.0);
glColor3f(1.0, 0.0, 0.0);
glPointSize(4.0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0, 200.0, 0.0, 200.0);
}
```

```
int main(int argc, char* argv[])
{       glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_SINGLE|GLUT_R
GB);
        glutInitWindowSize(500,400);
        glutInitWindowPosition(100, 150);
        glutCreateWindow("my Program");
        glutDisplayFunc(mydisplay);
        MyInit();
        glutMainLoop();
}
```

Basic 2D square

1. INIT

```
#include <GL/glut.h>
void init()
 {
   glClearColor(1.0, 1.0, 1.0, 0.0);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   glOrtho(0.0, 500.0, 0.0, 500.0, -1.0, 1.0);
 }
```

2.Function

```
void drawSquare() {
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.0, 0.0); // Set color to black
    glBegin(GL_QUADS); // Draw a square using quads
    glVertex2i(100, 100); // Bottom-left vertex
    glVertex2i(200, 100); // Bottom-right vertex
    glVertex2i(200, 200); // Top-right vertex
    glVertex2i(100, 200); // Top-left vertex
    glEnd();
    glFlush(); // Flush OpenGL buffer }
```

3. main()

```
int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Square");
    init();
    glutDisplayFunc(drawSquare);
    glutMainLoop();
    return 0;}
```

3D Basic Square

```
#include <GL/glut.h>
void init() {
    glClearColor(0.0, 0.0, 0.0, 1.0); // Set the background color to black
    glMatrixMode(GL_PROJECTION); // Set the matrix mode to projection
    glLoadIdentity(); // Load the identity matrix
    glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0); // Set the orthographic view volume
}
```

```
void display() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the color
buffer
    glColor3f(1.0, 1.0, 1.0); // Set the drawing color to white


    glBegin(GL_QUADS); // Begin drawing quads
    glVertex3f(-0.5, -0.5, 0.0); // Bottom left corner
    glVertex3f(0.5, -0.5, 0.0); // Bottom right corner
    glVertex3f(0.5, 0.5, 0.0); // Top right corner
    glVertex3f(-0.5, 0.5, 0.0); // Top left corner
    glEnd(); // End drawing quads
    glFlush(); // Flush the buffer
}
```

```
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set the display
mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("3D Square Example"); // Create the window with
the given title
    init(); // Call the init function
    glutDisplayFunc(display); // Set the display function
    glutMainLoop(); // Enter the main loop
    return 0;
}
```

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

void init() {
    glClearColor(0.0, 0.0, 0.0, 1.0); // Set the
background color to black
    glMatrixMode(GL_PROJECTION); // Set the
matrix mode to projection
    glLoadIdentity(); // Load the identity matrix
    glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0); // Set the
orthographic view volume
}
```

```
void display() {
```

LAB TASK 1

- Write a program that shows a Hut made up of dots.

How to MAKE a Vertex

- **Void glVertex *( )** $\longrightarrow$ (*can be)
- void glVertex<u>2d</u>(GLdouble *x*, GLdouble *y*)

  void glVertex<u>2f</u>(GLfloat *x*, GLfloat *y*)

  void glVertex<u>2i</u>(GLint *x*, GLint *y*)

  void glVertex<u>2s</u>(GLshort *x*, GLshort *y*)

  void glVertex3d(GLdouble *x*, GLdouble *y*, GLdouble *z*)

  void glVertex3f(GLfloat *x*, GLfloat *y*, GLfloat *z*)

  void glVertex3i(GLint *x*, GLint *y*, GLint *z*)

  void glVertex3s(GLshort *x*, GLshort *y*, GLshort *z*)

A Simple OpenGL Program 1/4

```
#include “glut.h”
void display();
void reshape(GLsizei, GLsizei);
int main(int argc, char** argv){
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutCreateWindow("sample");
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutMainLoop();
    return 0;
}
```

A Simple OpenGL Program

```
void display(){
   glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
   glClear(GL_COLOR_BUFFER_BIT);
   glColor3f(1.0f, 1.0f, 1.0f);
   glutSolidTeapot(1.0);
   glFlush();
}
```

A Simple OpenGL Program 3/4

```
void reshape(GLsizei w, GLsizei h){
   glViewport(0, 0, w, h);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   glFrustum(-0.5, 0.5, -0.5, 0.5, 1.0, 20.0);
   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();
  gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0,
   0.0);
}
```

A Simple OpenGL Program

A simple OpenGL program

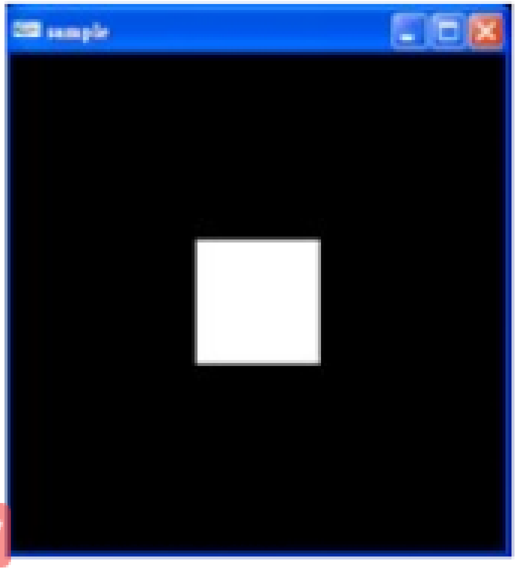
```
#include<GL/glut.h>

void GL_display(){
     glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
     glClear(GL_COLOR_BUFFER_BIT);
     glColor3f(1.0f, 1.0f, 1.0f);
     glutSolidCube(1.0);
     glFlush();
}

void GL_reshape(GLsizei w, GLsizei h){
     glViewport(0, 0, w, h);
     glMatrixMode(GL_PROJECTION);
     glLoadIdentity();
     glOrtho(-2.0f, 2.0f, -2.0f, 2.0f, -2.0f, 2.0f);
     glMatrixMode(GL_MODELVIEW);
     glLoadIdentity();
}
```

A simple OpenGL program

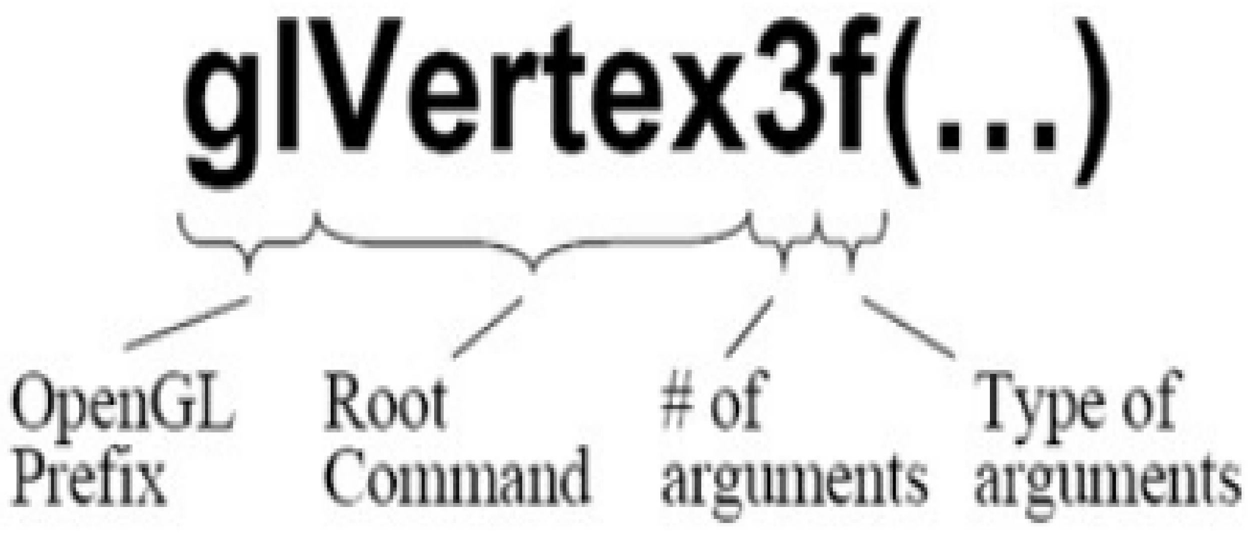
```
void main(int argc, char** argv){
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutCreateWindow("sample");
    glutDisplayFunc(GL_display);
    glutReshapeFunc(GL_reshape);
    glutMainLoop();
}
```



OpenGL command Syntax 1/2

- OpenGL commands use the prefix gl and initial capital letters for each word ex: glClearColor
- OpenGL defined constants begin with GL_ , use all capital letters and underscores to separate words ex: GL_COLOR_BUFFER_BIT

OpenGL command Syntax



GLUT Functions 1/7

- void glutInit(int *argcp, char **argv);
  - Initializing the GLUT library
  - Should be called before any other GLUT funcitons
  - http://www.opengl.org/resources/libraries/glut/spec3/node10.html
- void glutInitDisplayMode(unsigned int mode);
  - Specify a display mode for windows created.
  - GLUT_RGB / GLUT_RGBA / GLUT_INDEX
  - GLUT_SINGLE / GLUT_DOUBLE
  - GLUT_DEPTH / GLUT_STENCIL / GLUT_ACCUM
  - http://www.opengl.org/resources/libraries/glut/spec3/node12.html

GLUT Functions 2/7

- void glutInitWindowSize(int width, int height);
- void glutInitWindowPosition(int x, int y);
  - Initializing the window position and size.
  - http://www.opengl.org/resources/libraries/glut/spec3/node11.html
- int glutCreateWindow(char *name);
  - Open a window with previous settings.
  - http://www.opengl.org/resources/libraries/glut/spec3/node16.html#383

GLUT Functions 3/7

- void glutDisplayFunc(void (*func)(void));
  - Called whenever the contents of the windows need to be redrawn.
  - Put whatever you wish to draw on screen here.
  - Use glutPostRedisplay() to manually ask GLUT to recall this display function
  - http://www.opengl.org/resources/libraries/glut/spec3/node46.html

GLUT Functions 4/7

- void glutReshapeFunc(void (*func)(int width, int height));
  - Called whenever the window is resized or moved.
  - You should always call glViewport() here to resize your viewport.
  - http://www.opengl.org/resources/libraries/glut/spec3/node48.html

GLUT Functions 5/7

- void glutKeyboardFunc(void (*func)(unsigned char key, int x, int y));
  - Sets the keyboard callback for the current window.
  - http://www.opengl.org/resources/libraries/glut/spec3/node49.html
- void glutIdleFunc(void (*func)(void));
  - Sets the global idle callback.
  - http://www.opengl.org/resources/libraries/glut/spec3/node63.html

GLUT Functions 6/7

- void glutMouseFunc(void (*func)(int button, int state, int x, int y));
  - sets the mouse callback for the current window.
  - http://www.opengl.org/resources/libraries/glut/spec3/node50.html
- void glutMotionFunc(void (*func)(int x, int y));
  - set the motion callbacks respectively for the current window.
  - http://www.opengl.org/resources/libraries/glut/spec3/node51.html

GLUT Functions 7/7

- void glutMainLoop(void);
  - Enter the GLUT processing loop and never return.
  - http://www.opengl.org/resources/libraries/glut/spec3/node14.html#376
- void glutPostRedisplay(void);
  - marks the current window as needing to be redisplayed.
  - http://www.opengl.org/resources/libraries/glut/spec3/node20.html#465

- Write a program that creates a square using lines.