Section 3

Computer Graphics

Opengl

- **Setup steps:**
- Copy files from lib folder and paste it in this path
  C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\lib
- Copy GL folder (inside include folder) and paste it in this path
  C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\include
- Copy files from dll floder and Paste it in these paths (With replacement)
  C:\Windows\System32
  C:\Windows\SysWOW64

Opengl (cont.)

- **Additional dependencies:**

- right click on project name

properties → linker→ input→ additional dependencies → edit

- **And write:**

opengl32.lib

Glu32.lib

Glut32.lib

glutInit

- **glutInit** is used to initialize the GLUT library.
- **Usage :** glutInit(&argc, argv);
- **argv** is a pointer to an array of nullterminated strings, and **argc** says how large this array is.
- **Like this**

c:\test.exe hello world

Then argc=3

argv[0]="c:\test.exe"

argv[1]="hello"

argv[2]="world"

glutInitDisplayMode

- **glutInitDisplayMode** sets the initial display mode.
- **Usage:** glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
- **GLUT_DOUBLE:** select a double buffered window.
- **GLUT_SINGLE:** select a single buffered window.

glutInitWindowPosition & glutInitWindowSize

- **glutInitWindowPosition** and **glutInitWindowSize** set the initial window position and size respectively.
- **Usage :** void glutInitWindowSize(int width, int height);

  void glutInitWindowPosition(int x, int y);

glutCreateWindow

- **glutCreateWindow** create new window with the previous window size and window position.
- **Usage:** glutCreateWindow("name");

- The name will be provided to the window system as the window's name.

glClearColor

- **glClearColor** specify clear values for the color buffers
- **Usage:** glClearColor(red, green, blue, alpha);

- The alpha is opacity.

- Values specified by glClearColor from 0 to 1.

glClear

- **glClear** clear buffers before starting drawing.
- **Usage:** glClear(GL_COLOR_BUFFER_BIT);
- **GL_COLOR_BUFFER_BIT:** Indicates the buffers currently enabled for color writing.

glBegin & glEnd

- **glBegin and glEnd** delimit the vertices that define a primitive or a group of like primitives. glBegin accepts a single argument that specifies in which of ten ways the vertices are interpreted.

glColor3f

- **glColor3f** set the current drawing color
- Usage: glColor3f(GLfloat red, GLfloat green, GLfloat blue);

- Values specified by glColor3f from 0 to 1.

glVertex3f

glVertex3f commands are used within glBegin/glEnd pairs to specify point, line, and polygon vertices.

Usage: glVertex3f(GLfloat x, GLfloat y, GLfloat z);

glFlush & glutSwapBuffers

- If we use **GLUT_SINGLE** we must write glFlush() after glEnd
- If we use **GLUT_DOUBLE** we must write glutSwapBuffers() after glEnd

glutDisplayFunc

- glutDisplayFunc sets the display callback for the current window.
- glutDisplayFunc is called whenever your window must be redrawn.
- **Usage:** glutDisplayFunc(display);

Where “display” is the function that contain the shape to be drawn.

glPointSize

- Specifies the diameter of rasterized points. The initial value is 1.
- If point size mode is disabled, use:

glEnable(GL_POINT_SIZE);

Usage: glPointSize(GLfloat *size*);

glBegin()

- The ten possible arguments for glBegin()

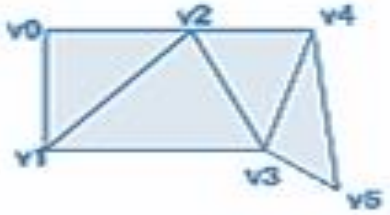
GL_POINTS

GL_LINES

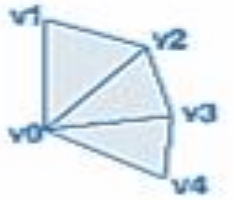
GL_LINE_STRIP

GL_LINE_LOOP

GL_TRIANGLES

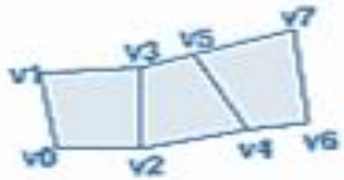


GL_TRIANGLE_STRIP

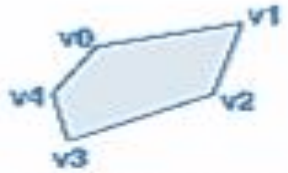


GL_TRIANGLE_FAN

GL_QUADS



GL_QUAD_STRIP



GL_POLYGON

GL_POINTS

- Draws a point at each of the n vertices.

GL_POINTS

```
#include<GL\glut.h>
void display()
{
glClear(GL_COLOR_BUFFER_BIT);
glBegin(GL_POINTS);
glColor3f(0.0, 0.0, 0.0);
glVertex3f(-0.5, -0.5,0);
glVertex3f(0.5, -0.5,0);
glVertex3f(0.0, 0.5,0);
glEnd();
glutSwapBuffers();
}
```

```
int main(int argc, char** argv)
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE |
GLUT_RGB);
glutInitWindowSize(300, 300);
glutInitWindowPosition(1, 1);
glutCreateWindow("GL_POINTS");
glClearColor(0.0, 1.0, 1.0, 1.0);
glPointSize(10);
glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```

GL_LINES

- Draws a series of unconnected line segments. Segments are drawn between v0 and v1, between v2 and v3, and so on. If n is odd, the last point is ignored.

GL_LINE_STRIP

- Draws a line segment from v0 to v1, then from v1 to v2, and so on, finally drawing the segment from vn-2 to vn-1. Nothing is drawn unless n is larger than 1. the lines can intersect arbitrarily.

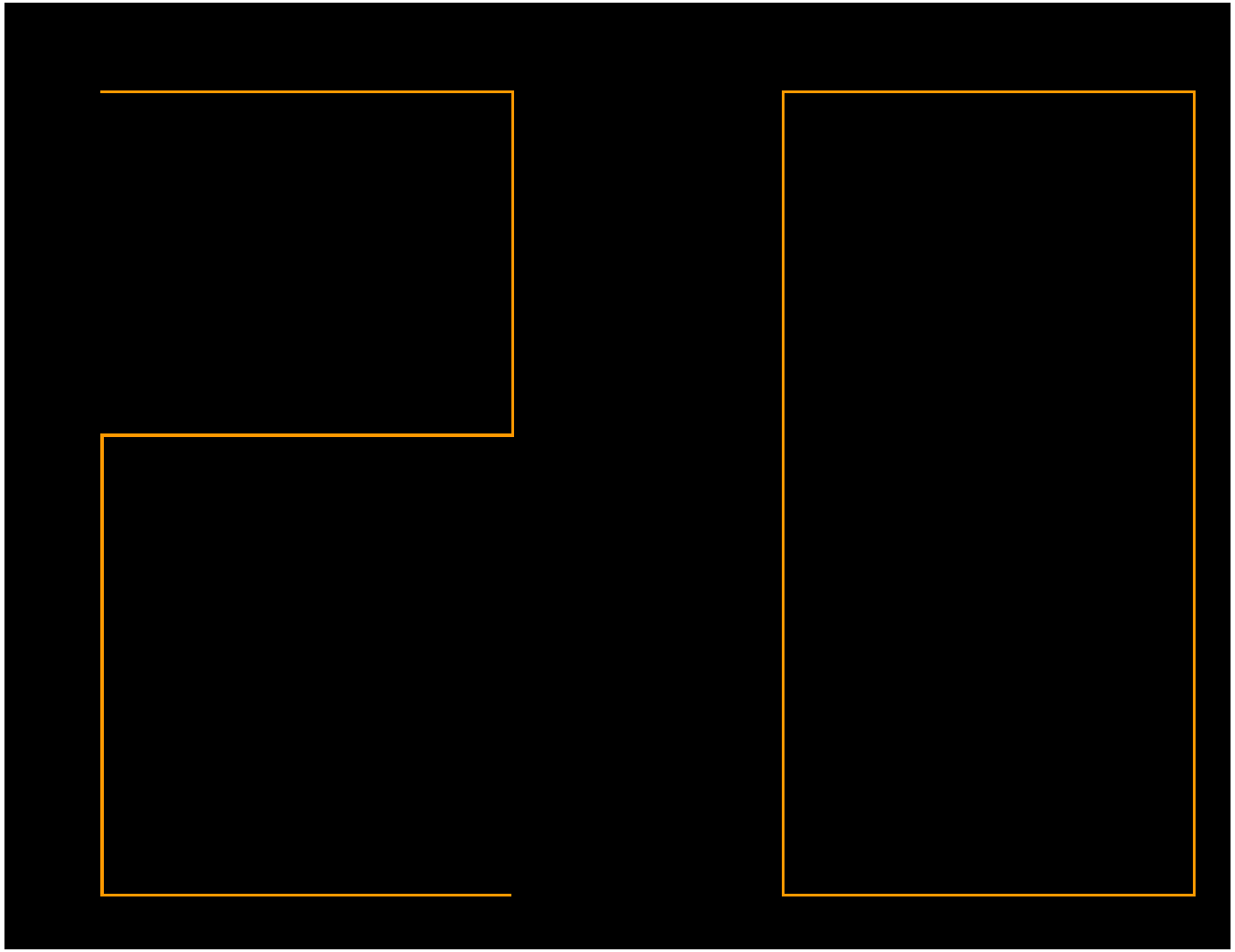
GL_LINE_LOOP

- Same as GL_LINE_STRIP, except that a final line segment is drawn from vn-1 to v0, completing a loop.

GL_LINES & GL_LINE_STRIP & GL_LINE_LOOP

```
#include<GL/glut.h>
void display2()       //line
{ glClear(GL_COLOR_BUFFER_BIT);
glColor3f(1.0, 0.6, 0.0);
glBegin(GL_LINES);
glVertex2f(-.8, -.8);
glVertex2f(.8, 0);
glVertex2f(0, .8);
glVertex2f(-.8, .8);
glEnd();
glFlush(); }
void display1()    //line Strip
{ glClear(GL_COLOR_BUFFER_BIT);
glColor3f(1.0, 0.6, 0.0);
glBegin(GL_LINE_STRIP);
glVertex2f(-.8, -.8);
glVertex2f(.8, 0);
glVertex2f(0, .8);
glVertex2f(-.8, .8);
glEnd();
glFlush(); }
void display()  //line loop
{ glClear(GL_COLOR_BUFFER_BIT);
glColor3f(1.0, 0.6, 0.0);
glBegin(GL_LINE_LOOP);
glVertex2f(-.8, -.8);
glVertex2f(.8, 0);
glVertex2f(0, .8);
glVertex2f(-.8, .8);
glEnd();
glFlush(); }
int main(int argc, char *argv[])
{ glutInit(&argc, argv);
glutInitWindowSize(250, 250);
glutInitWindowPosition(50, 100);
glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
glutCreateWindow("GL_LINES");
glutDisplayFunc(display2);
glutInitWindowPosition(350, 100);
glutCreateWindow("GL_LINE_LOOP");
glutDisplayFunc(display);
glutInitWindowPosition(650, 100);
glutCreateWindow("GL_LINE_STRIP");
glutDisplayFunc(display1);
glutMainLoop();
return 0;}
```

Sheet (Draw this using opengl)



Sheet

- Draw the first character of your name