Kingdom of Saudi Arabia
المملكة العربية السعودية
Ministry of Education
وزارة التعليم
Umm AlQura University
جامعة أم القرى
Adham University College
الكلية الجامعية بأضم
Computer Science Department
قسم الحاسب الآلي

CS
Adam College
Department

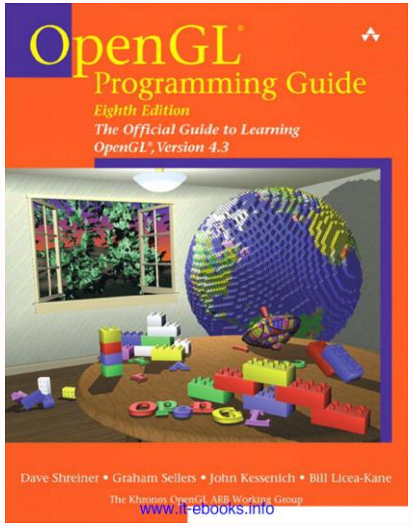
Computer Graphics Course, 3-6803430

LAB

T.Mariah Khayat

References

- *Lab Lectures, Computer Graphics, Taif University, Faculty Of Computers And Information Technology, TA. Maha Thafar &TA. Haifa Alshehri, TA.Sohair Soliman & L.Shakila Bano.*
- *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 4.3, 8th edition, Dave Shreiner, Graham Sellers, John Kessenich, Bill Licea-Kane & The Khronos OpenGL ARB Working Group, Addison-Wesley.*



OpenGL
Programming Guide
Eighth Edition
The Official Guide to Learning OpenGL, Version 4.3
Dave Shreiner • Graham Sellers • John Kessenich • Bill Licea-Kane
The Khronos OpenGL ARB Working Group
www.it-ebooks.info

Lecture Four

Drawing Triangles, Quads and Polygons using OpenGL

Computer Graphics Course, 3-6803430

T.Mariah Khayat

content

1. Attributes in OpenGL
2. Triangles in OpenGL
3. Draw Triangles using OpenGL
4. Draw Triangle Fan using OpenGL
5. Quads in OpenGL
6. Draw a Square using OpenGL
7. Polygons in OpenGL
8. Draw an Ark Using OpenGL

Attributes in OpenGL

- **Attributes: are part of the OpenGL and determine the appearance of objects.**
- **Color** deal with all objects **(points, lines, triangles, quads, polygons).**
- Each color component stored separately in the frame buffer.
- Usually 8 bits per component in buffer.
- Note: in **`glColor3f`** **the color values range from** 0.0 to 1.0 , while in **`glColor3ub`** **the** values range from 0 to 255.

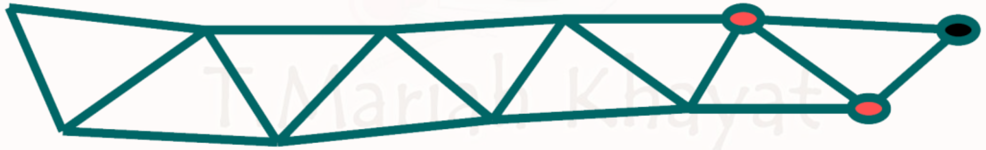
Attributes in OpenGL

- **Size** and **width** (points, lines).
- **Stipple** pattern (lines, polygons).

- Polygon mode:
  - Display as filled: (solid color) use the:
    - ❖`glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);`
  - Display edges use the functions:
    - ❖`glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);`

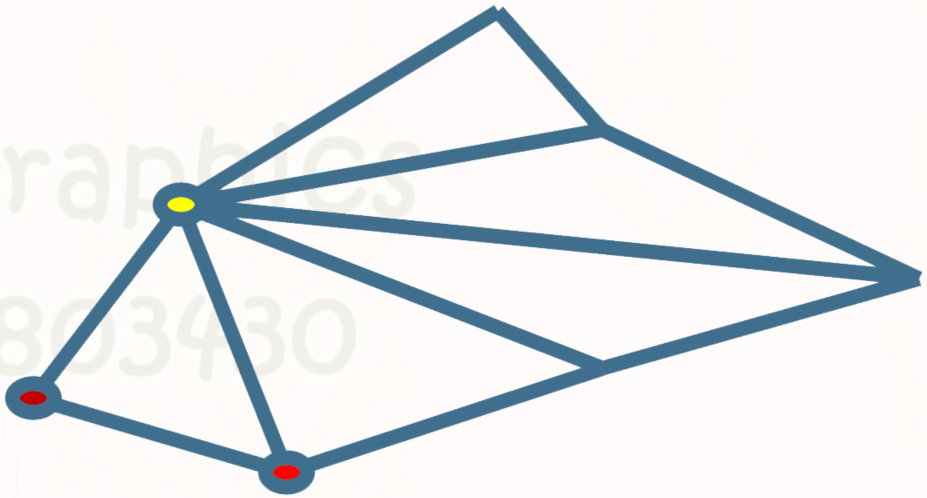
Triangles in OpenGL

- **There are 3 functions** with Triangle: {S | _STRIP | _FAN}.
- **Triangle (GL_TRIANGLES):**
  - to draw normal Triangle. & Represent triangle as 3 vertices.
- **Triangle Strip** : **(GL_TRIANGLE_STRIP):**
  - Group of triangles sharing 2 vertices from previous triangle.
  - Use triangles to represent a solid object as a mesh.
  - Triangles frequently appear in strips:



Triangles in OpenGL

- **Triangular Fans: (GL_TRIANGLE_FAN):**
  - Connected group sharing 1 common vertex, and 1 from previous triangle.

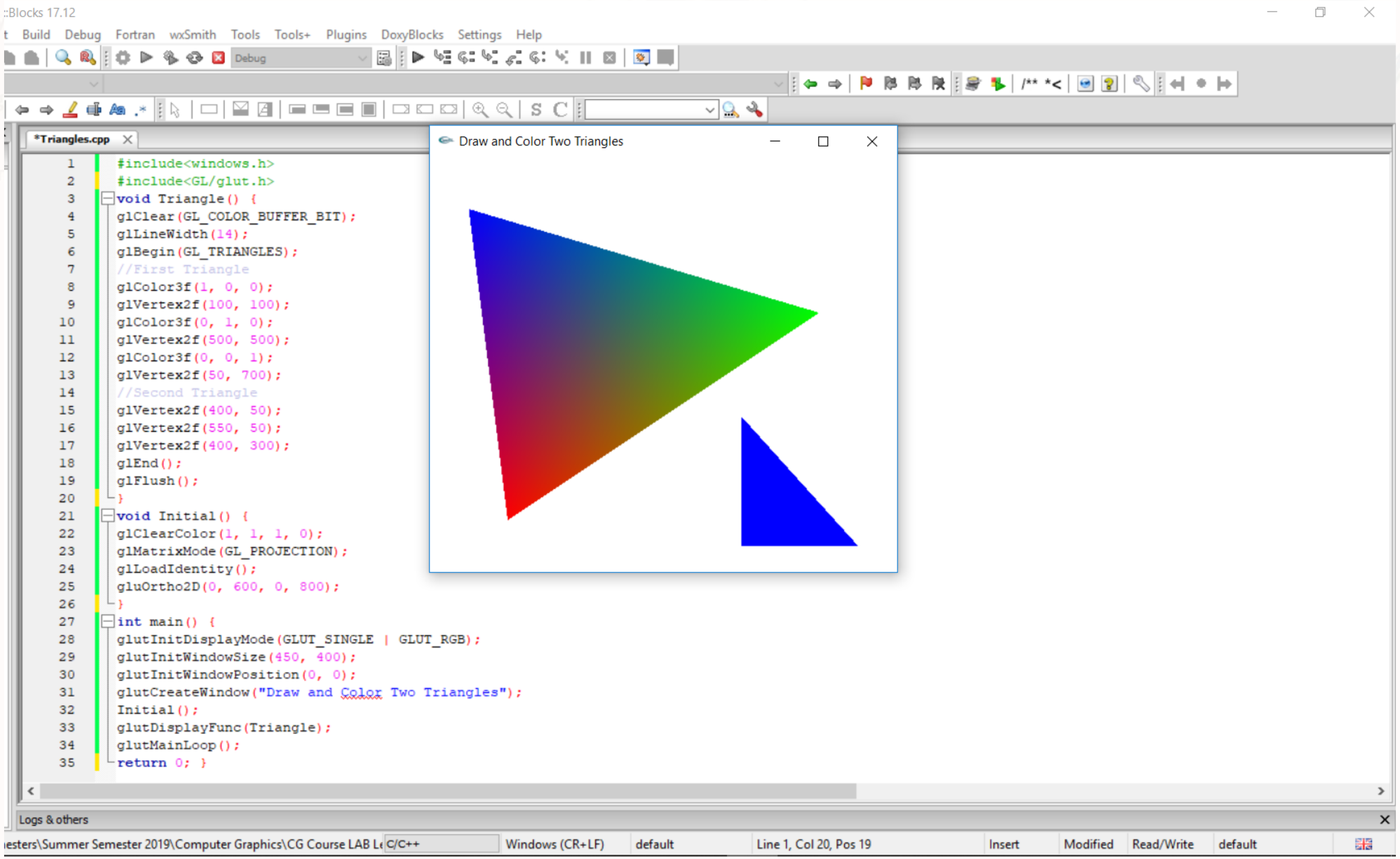


- **For tri-strips and fans**,
  - A new triangle is defined by 1 new vertex added to the strip or fans.

Draw Triangles using OpenGL

```
#include<windows.h>
#include<GL/glut.h>
void Triangle() {
glClear(GL_COLOR_BUFFER_BIT);
glLineWidth(14);
glBegin(GL_TRIANGLES);
//First Triangle
glColor3f(1, 0, 0);
glVertex2f(100, 100);
glColor3f(0, 1, 0);
glVertex2f(500, 500);
glColor3f(0, 0, 1);
glVertex2f(50, 700);
//Second Triangle
glVertex2f(400, 50);
glVertex2f(550, 50);
glVertex2f(400, 300);
glEnd();
glFlush();
}
void Initial() {
glClearColor(1, 1, 1, 0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0, 600, 0, 800);
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(450, 400);
glutInitWindowPosition(0, 0);
glutCreateWindow("Draw and Color Two Triangles");
Initial();
glutDisplayFunc(Triangle);
glutMainLoop();
return 0; }
```

Draw Triangles using OpenGL


:Blocks 17.12
t Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
*Triangles.cpp
Draw and Color Two Triangles
Logs & others

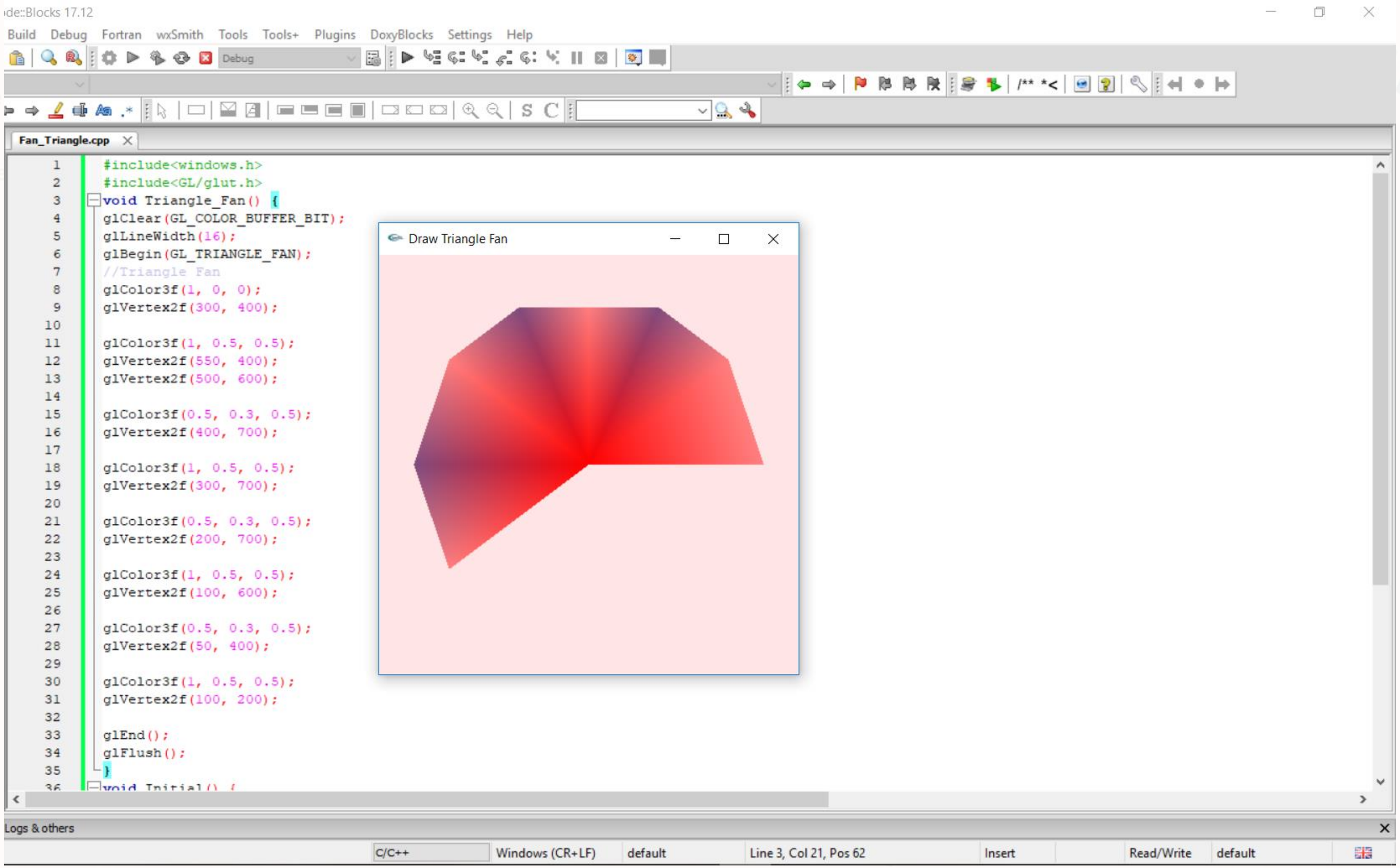
```cpp
#include<windows.h>
#include<GL/glut.h>
void Triangle() {
glClear(GL_COLOR_BUFFER_BIT);
glLineWidth(14);
glBegin(GL_TRIANGLES);
//First Triangle
glColor3f(1, 0, 0);
glVertex2f(100, 100);
glColor3f(0, 1, 0);
glVertex2f(500, 500);
glColor3f(0, 0, 1);
glVertex2f(50, 700);
//Second Triangle
glVertex2f(400, 50);
glVertex2f(550, 50);
glVertex2f(400, 300);
glEnd();
glFlush();
}
void Initial() {
glClearColor(1, 1, 1, 0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0, 600, 0, 800);
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(450, 400);
glutInitWindowPosition(0, 0);
glutCreateWindow("Draw and Color Two Triangles");
Initial();
glutDisplayFunc(Triangle);
glutMainLoop();
return 0; }
```

```
esters\Summer Semester 2019\Computer Graphics\CG Course LAB L   C/C++   Windows (CR+LF)   default   Line 1, Col 20, Pos 19   Insert   Modified   Read/Write   default
```

Draw Triangle Fan using OpenGL

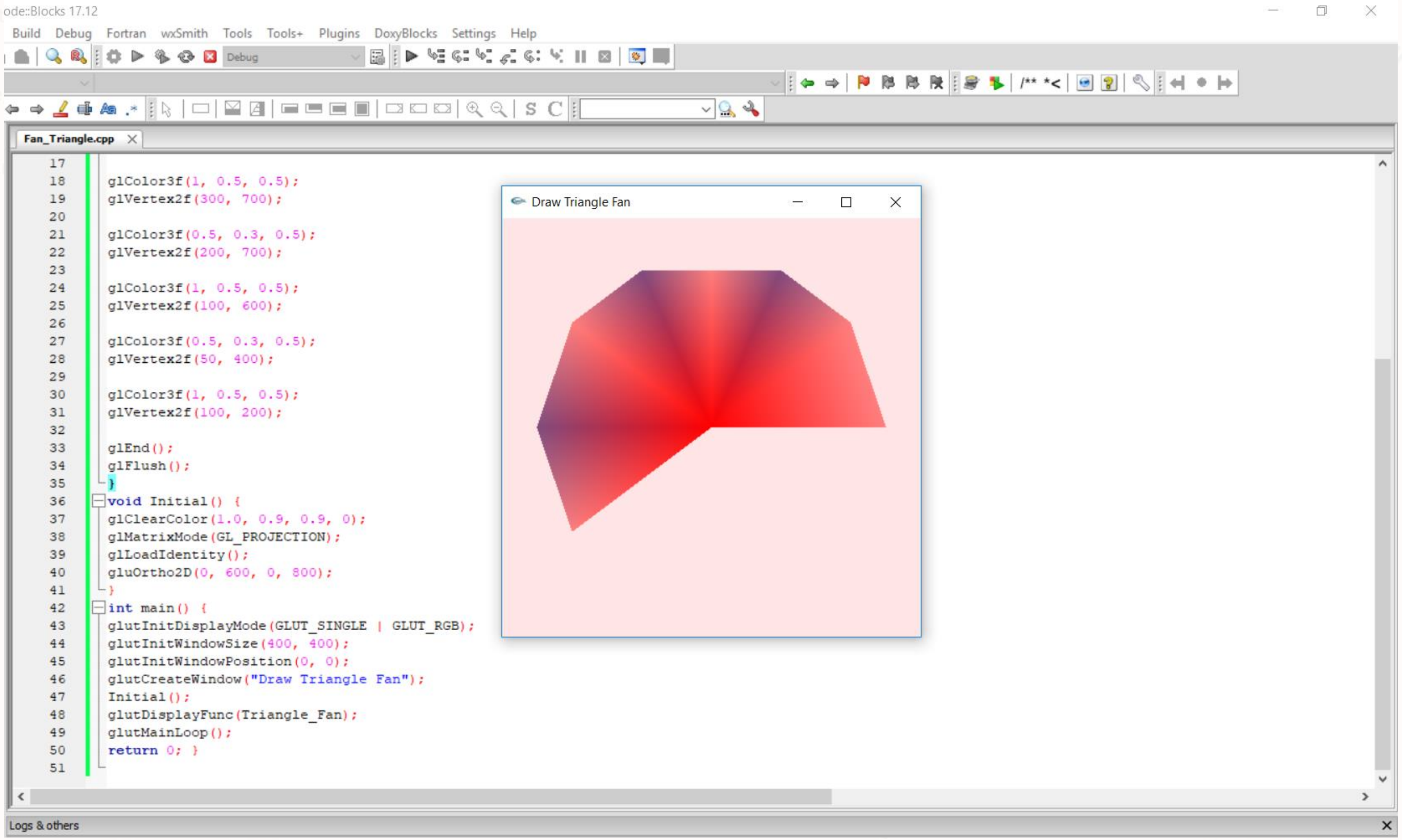
```
#include<windows.h>
#include<GL/glut.h>
void Triangle_Fan() {
glClear(GL_COLOR_BUFFER_BIT);
glLineWidth(16);
glBegin(GL_TRIANGLE_FAN);
//Triangle Fan
glColor3f(1, 0, 0);
glVertex2f(300, 400);

glColor3f(1, 0.5, 0.5);
glVertex2f(550, 400);
glVertex2f(500, 600);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(400, 700);

glColor3f(1, 0.5, 0.5);
glVertex2f(300, 700);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(200, 700);

glColor3f(1, 0.5, 0.5);
glVertex2f(100, 600);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(50, 400);

glColor3f(1, 0.5, 0.5);
glVertex2f(100, 200);

glEnd();
glFlush();
}
void Initial() {
glClearColor(1.0, 0.9, 0.9, 0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0, 600, 0, 800);
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400, 400);
glutInitWindowPosition(0, 0);
glutCreateWindow("Draw Triangle Fan");
Initial();
glutDisplayFunc(Triangle_Fan);
glutMainLoop();
return 0; }
```

Draw Triangle Fan using OpenGL



```
#include<windows.h>
#include<GL/glut.h>
void Triangle_Fan() {
glClear(GL_COLOR_BUFFER_BIT);
glLineWidth(16);
glBegin(GL_TRIANGLE_FAN);
//Triangle Fan
glColor3f(1, 0, 0);
glVertex2f(300, 400);

glColor3f(1, 0.5, 0.5);
glVertex2f(550, 400);
glVertex2f(500, 600);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(400, 700);

glColor3f(1, 0.5, 0.5);
glVertex2f(300, 700);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(200, 700);

glColor3f(1, 0.5, 0.5);
glVertex2f(100, 600);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(50, 400);

glColor3f(1, 0.5, 0.5);
glVertex2f(100, 200);

glEnd();
glFlush();
}
void Initial() {
```

Draw Triangle Fan using OpenGL


ode::Blocks 17.12
Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
Fan_Triangle.cpp
Draw Triangle Fan
Logs & others

```
glColor3f(1, 0.5, 0.5);
glVertex2f(300, 700);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(200, 700);

glColor3f(1, 0.5, 0.5);
glVertex2f(100, 600);

glColor3f(0.5, 0.3, 0.5);
glVertex2f(50, 400);

glColor3f(1, 0.5, 0.5);
glVertex2f(100, 200);

glEnd();
glFlush();
}
void Initial() {
glClearColor(1.0, 0.9, 0.9, 0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0, 600, 0, 800);
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400, 400);
glutInitWindowPosition(0, 0);
glutCreateWindow("Draw Triangle Fan");
Initial();
glutDisplayFunc(Triangle_Fan);
glutMainLoop();
return 0; }
```

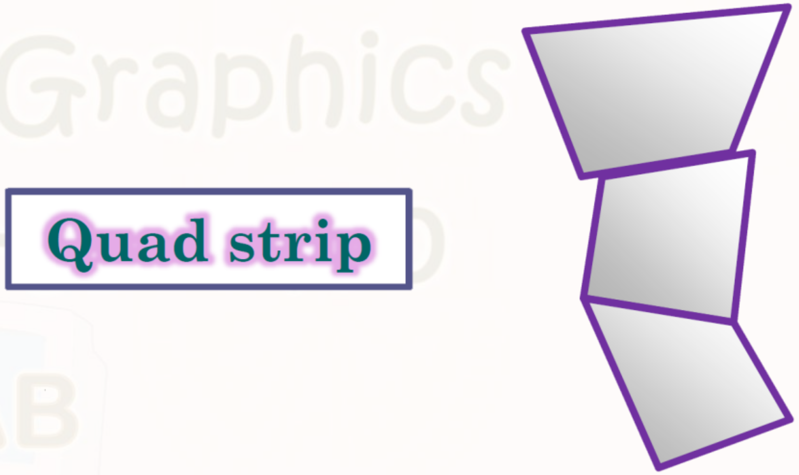
Quads using OpenGL

- **There are 2 functions** with Quad: {**S** | **_STRIP** }.

- **Quad :(GL_QUADS):**
  - To draw normal Quad & Represent as 4 vertices.
  - You can draw more than one quad by determine 4 vertex of each Quad.

Quads using OpenGL

- **Quad Strip : (GL_QUAD_STRIP):**
  - Group of Quads sharing 2 vertices from previous Quad.

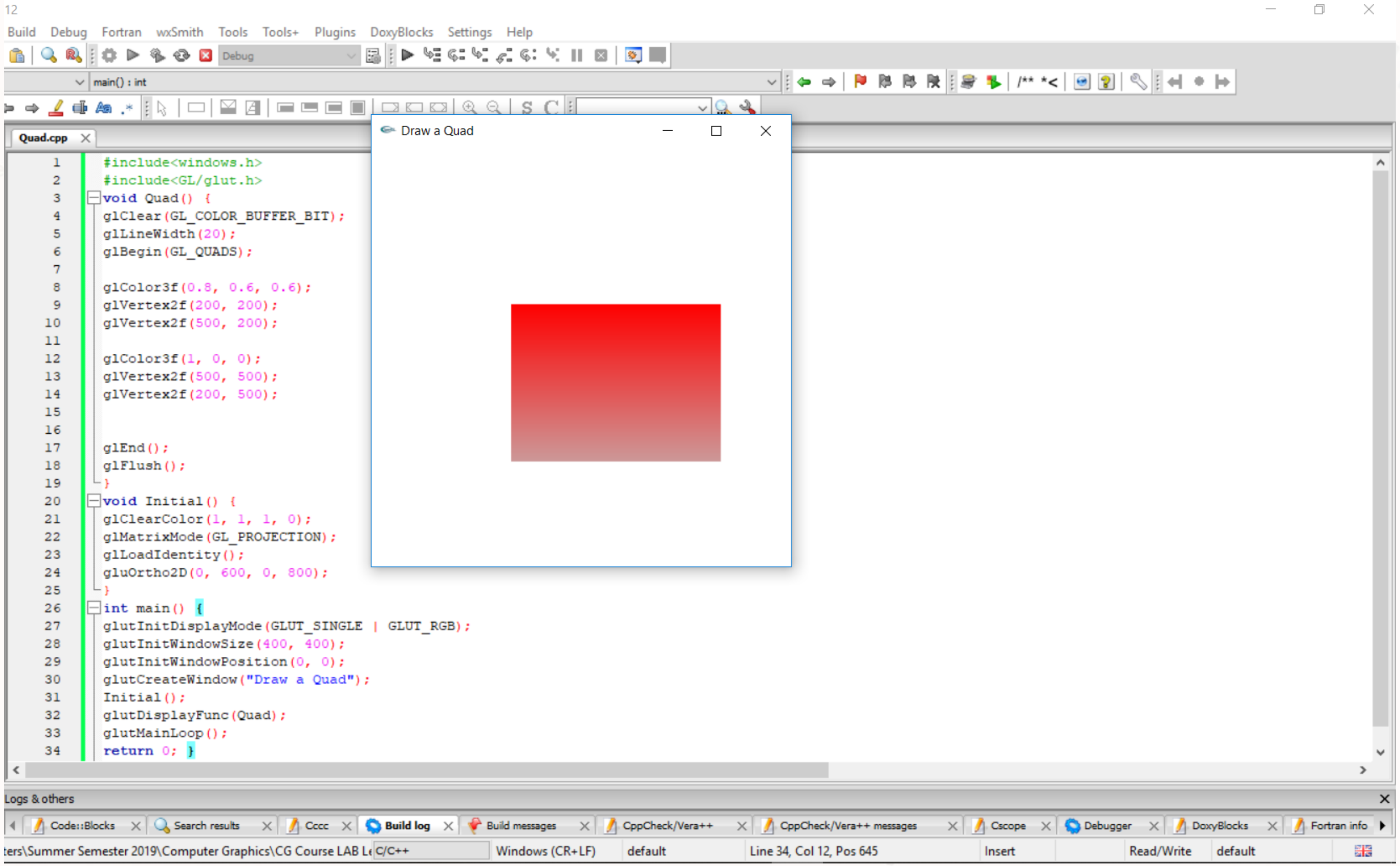


2 vertices per quad

Draw a Square using OpenGL

```
#include<windows.h>
#include<GL/glut.h>
void Quad() {
glClear(GL_COLOR_BUFFER_BIT);
glLineWidth(20);
glBegin(GL_QUADS);
glColor3f(0.8, 0.6, 0.6);
glVertex2f(200, 200);
glVertex2f(500, 200);
glColor3f(1, 0, 0);
glVertex2f(500, 500);
glVertex2f(200, 500);
glEnd();
glFlush();
}
void Initial() {
glClearColor(1, 1, 0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0, 600, 0, 800);
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400, 400);
glutInitWindowPosition(0, 0);
glutCreateWindow("Draw a Quad");
Initial();
glutDisplayFunc(Quad);
glutMainLoop();
return 0; }
```

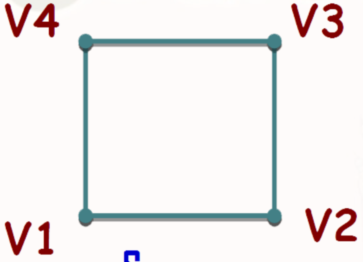
Draw a Square using OpenGL



```
#include<windows.h>
#include<GL/glut.h>
void Quad() {
glClear(GL_COLOR_BUFFER_BIT);
glLineWidth(20);
glBegin(GL_QUADS);

glColor3f(0.8, 0.6, 0.6);
glVertex2f(200, 200);
glVertex2f(500, 200);

glColor3f(1, 0, 0);
glVertex2f(500, 500);
glVertex2f(200, 500);


glEnd();
glFlush();
}
void Initial() {
glClearColor(1, 1, 1, 0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0, 600, 0, 800);
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400, 400);
glutInitWindowPosition(0, 0);
glutCreateWindow("Draw a Quad");
Initial();
glutDisplayFunc(Quad);
glutMainLoop();
return 0; }
```

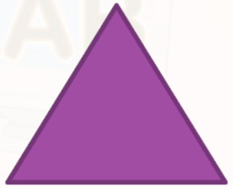
Polygons in OpenGL

- **A Polygon:** is a 2D shape that is made up of a number of vertices.
- An ordered set of vertices defines a polygon.
- You can also set other properties of the polygon, such as its color.
- **Quadrilateral** & **Triangle** is a special case of polygon.
- Use **GL_POLYGON** argument to draw polygon by determine its vertexes.

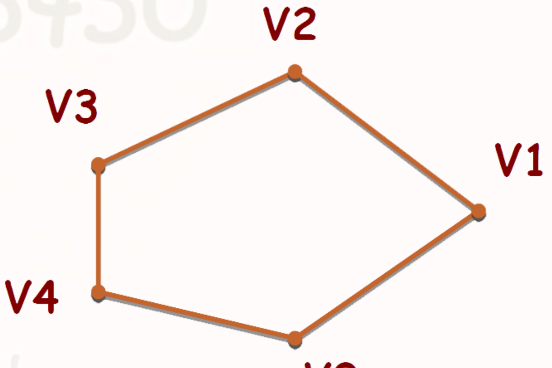


Square

(Quad)

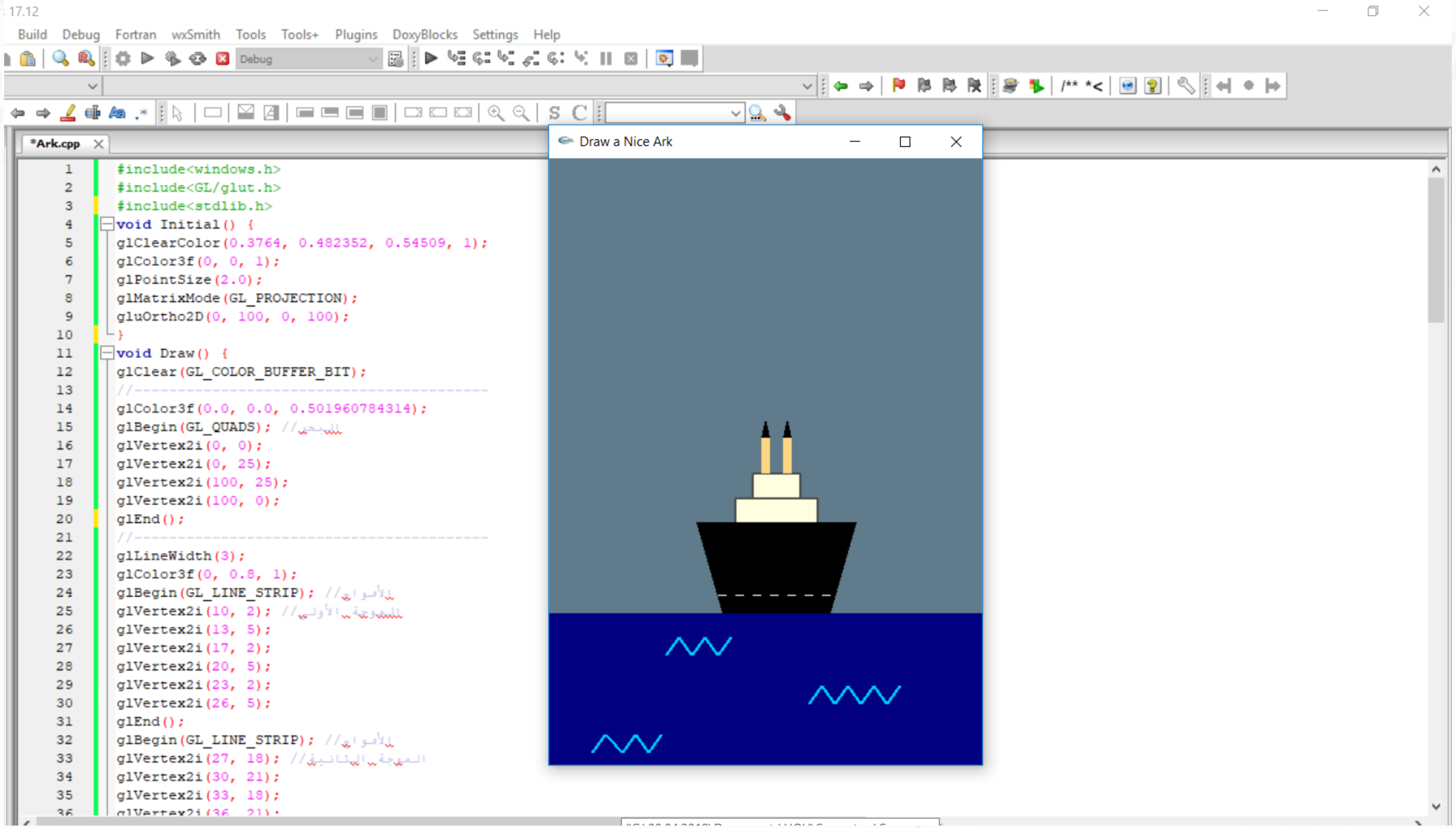


Triangle



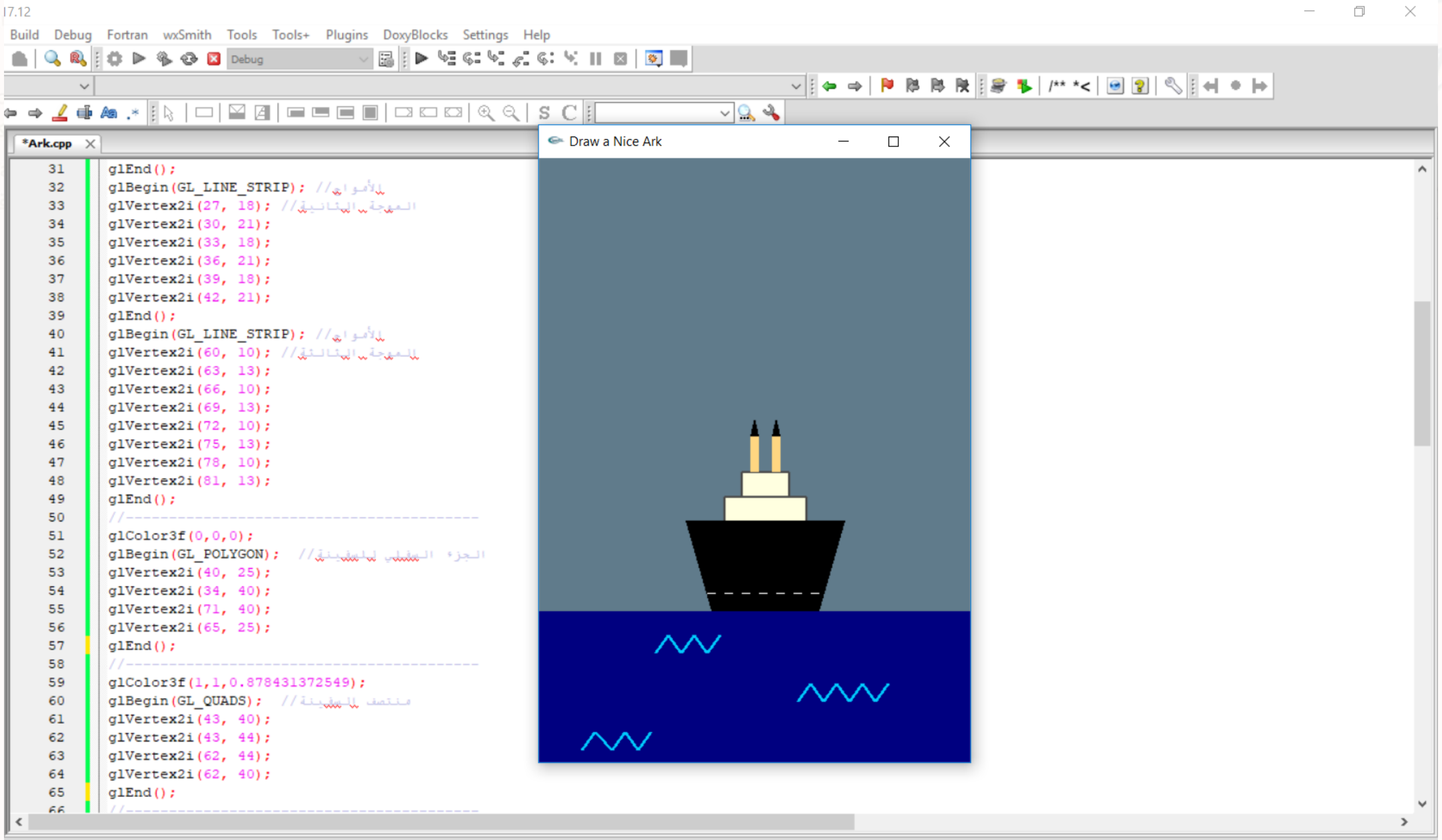
Polygon

Draw Ark using OpenGL



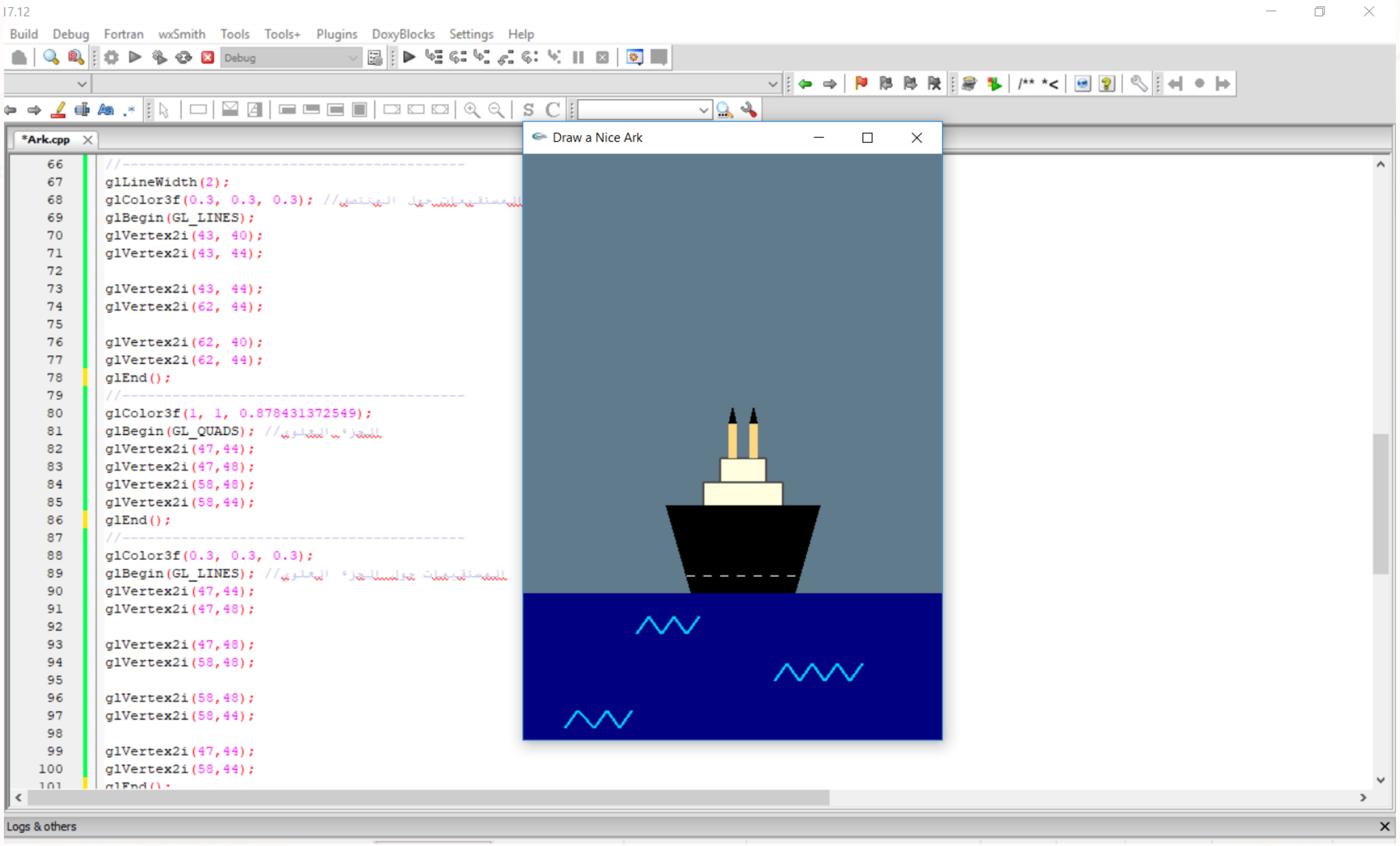
```
#include<windows.h>
#include<GL/glut.h>
#include<stdlib.h>
void Initial() {
glClearColor(0.3764, 0.482352, 0.54509, 1);
glColor3f(0, 0, 1);
glPointSize(2.0);
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0, 100, 0, 100);
}
void Draw() {
glClear(GL_COLOR_BUFFER_BIT);
//--------------------------------------
glColor3f(0.0, 0.0, 0.501960784314);
glBegin(GL_QUADS); //البحر
glVertex2i(0, 0);
glVertex2i(0, 25);
glVertex2i(100, 25);
glVertex2i(100, 0);
glEnd();
//--------------------------------------
glLineWidth(3);
glColor3f(0, 0.8, 1);
glBegin(GL_LINE_STRIP); //الأمواج
glVertex2i(10, 2); //الموجة الأولى
glVertex2i(13, 5);
glVertex2i(17, 2);
glVertex2i(20, 5);
glVertex2i(23, 2);
glVertex2i(26, 5);
glEnd();
glBegin(GL_LINE_STRIP); //الأمواج
glVertex2i(27, 18); //الموجة الثانية
glVertex2i(30, 21);
glVertex2i(33, 18);
glVertex2i(36, 21);
```

Draw Ark using OpenGL


17.12
Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
*Ark.cpp
Draw a Nice Ark

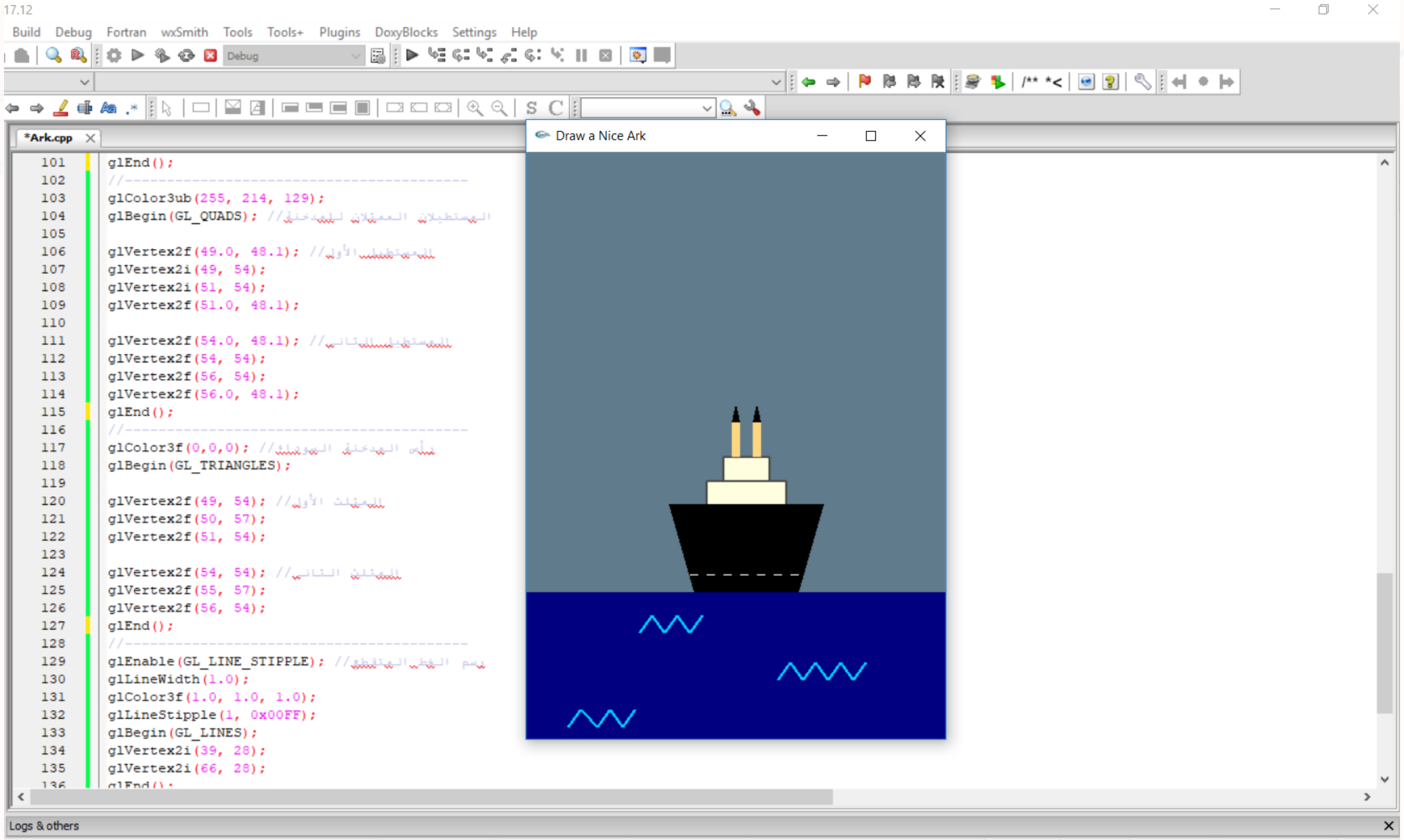
```
glEnd();
glBegin(GL_LINE_STRIP); //الأمواج
glVertex2i(27, 18); //الموجة الثانية
glVertex2i(30, 21);
glVertex2i(33, 18);
glVertex2i(36, 21);
glVertex2i(39, 18);
glVertex2i(42, 21);
glEnd();
glBegin(GL_LINE_STRIP); //الأمواج
glVertex2i(60, 10); //الموجة الثالثة
glVertex2i(63, 13);
glVertex2i(66, 10);
glVertex2i(69, 13);
glVertex2i(72, 10);
glVertex2i(75, 13);
glVertex2i(78, 10);
glVertex2i(81, 13);
glEnd();
//-----------------------------------------
glColor3f(0,0,0);
glBegin(GL_POLYGON);  //الجزء السفلي للسفينة
glVertex2i(40, 25);
glVertex2i(34, 40);
glVertex2i(71, 40);
glVertex2i(65, 25);
glEnd();
//-----------------------------------------
glColor3f(1,1,0.878431372549);
glBegin(GL_QUADS);  //منتصف السفينة
glVertex2i(43, 40);
glVertex2i(43, 44);
glVertex2i(62, 44);
glVertex2i(62, 40);
glEnd();
//-----------------------------------------
```

Draw Ark using OpenGL


17.12
Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
*Ark.cpp
Draw a Nice Ark
Logs & others

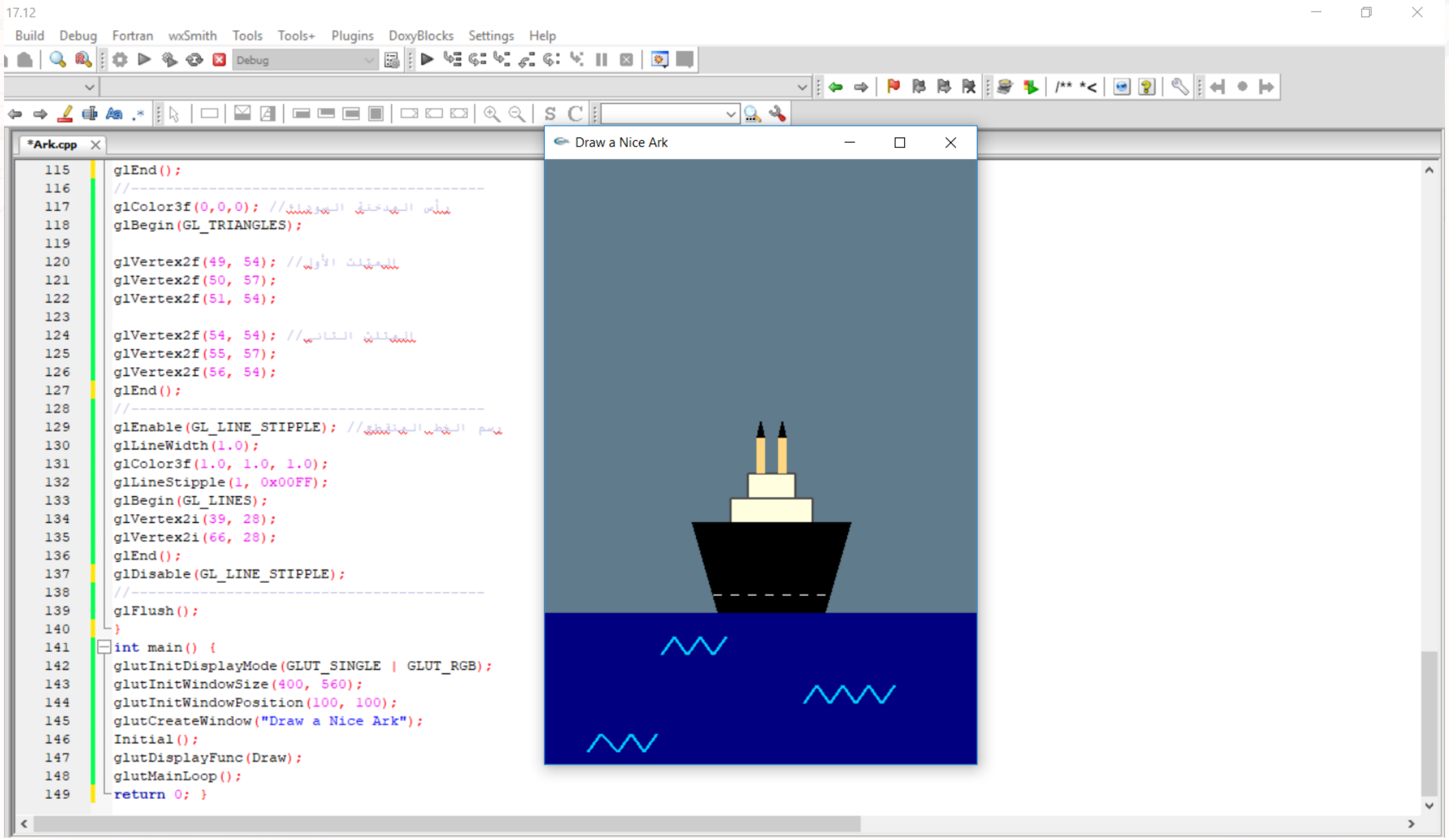
```
//----------------------------------------
glLineWidth(2);
glColor3f(0.3, 0.3, 0.3); //المستقيمات حول المنتصف
glBegin(GL_LINES);
glVertex2i(43, 40);
glVertex2i(43, 44);

glVertex2i(43, 44);
glVertex2i(62, 44);

glVertex2i(62, 40);
glVertex2i(62, 44);
glEnd();
//----------------------------------------
glColor3f(1, 1, 0.878431372549);
glBegin(GL_QUADS); //الجزء العلوي
glVertex2i(47,44);
glVertex2i(47,48);
glVertex2i(58,48);
glVertex2i(58,44);
glEnd();
//----------------------------------------
glColor3f(0.3, 0.3, 0.3);
glBegin(GL_LINES); //المستقيمات حول الجزء العلوي
glVertex2i(47,44);
glVertex2i(47,48);

glVertex2i(47,48);
glVertex2i(58,48);

glVertex2i(58,48);
glVertex2i(58,44);

glVertex2i(47,44);
glVertex2i(58,44);
glEnd();
```

Draw Ark using OpenGL



```
glEnd();
//----------------------------------------
glColor3ub(255, 214, 129);
glBegin(GL_QUADS); //المستطيلان الممثلان للمدخنة

glVertex2f(49.0, 48.1); //المستطيل الأول
glVertex2i(49, 54);
glVertex2i(51, 54);
glVertex2f(51.0, 48.1);

glVertex2f(54.0, 48.1); //المستطيل الثاني
glVertex2f(54, 54);
glVertex2f(56, 54);
glVertex2f(56.0, 48.1);
glEnd();
//----------------------------------------
glColor3f(0,0,0); //رأس المدخنة السوداء
glBegin(GL_TRIANGLES);

glVertex2f(49, 54); //المثلث الأول
glVertex2f(50, 57);
glVertex2f(51, 54);

glVertex2f(54, 54); //المثلث الثاني
glVertex2f(55, 57);
glVertex2f(56, 54);
glEnd();
//----------------------------------------
glEnable(GL_LINE_STIPPLE); //رسم الخط المتقطع
glLineWidth(1.0);
glColor3f(1.0, 1.0, 1.0);
glLineStipple(1, 0x00FF);
glBegin(GL_LINES);
glVertex2i(39, 28);
glVertex2i(66, 28);
glEnd();
```

Draw Ark using OpenGL



```
glEnd();
//------------------------------------------
glColor3f(0,0,0); //رأس المدخنة السوداء
glBegin(GL_TRIANGLES);

glVertex2f(49, 54); //المثلث الأول
glVertex2f(50, 57);
glVertex2f(51, 54);

glVertex2f(54, 54); //المثلث الثاني
glVertex2f(55, 57);
glVertex2f(56, 54);
glEnd();
//------------------------------------------
glEnable(GL_LINE_STIPPLE); //رسم الخط المتقطع
glLineWidth(1.0);
glColor3f(1.0, 1.0, 1.0);
glLineStipple(1, 0x00FF);
glBegin(GL_LINES);
glVertex2i(39, 28);
glVertex2i(66, 28);
glEnd();
glDisable(GL_LINE_STIPPLE);
//------------------------------------------
glFlush();
}
int main() {
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400, 560);
glutInitWindowPosition(100, 100);
glutCreateWindow("Draw a Nice Ark");
Initial();
glutDisplayFunc(Draw);
glutMainLoop();
return 0; }
```

Excercise

- Draw a Square that satisfies the following specifications:
  - Line Width = 10.
  - Vertex1 = (200, 200), Vertex2 = (200, 600), Vertex3 = (600, 600), Vertex4 = (200, 600).
  - Color of the Square = Orange.
  - Background Color = White.
  - Window Title Bar = “My Orange Square”.

وصلّى الله وبارك على نبيّنا محمد

The End Summary of Lecture Four

T.Mariah Khayat
الأستاذة/ مارية خياط
Adham University College
الكلية الجامعية بأضم

mskhayat@uqu.edu.sa