CMT107 Visual Computing

Assignment Project Exam Help

1.4 Introduction to OpenGL

https://tutorcs.com

WeChat: cstutorcs

Xianfang Sun

School of Computer Science & Informatics
Cardiff University

Overview

- Introduction to OpenGL
  - What is OpenGL
  - OpenGL History
  - OpenGL Pipeline
  - OpenGL Components
  - Java OpenGL (Jogl)
    - Installation of Jogl on Eclipse
- OpenGL Programming
  - Basic OpenGL Coding Framework
  - OpenGL Geometric Primitives
  - A Simple OpenGL Program
    - In C
    - In Java

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

What is OpenGL?

- OpenGL: Open Graphics Library
  - Originally IRIS GL (Integrated Raster Imaging System Graphics Library) from Silicon Graphics
- OpenGL is NOT a language, it is
  - a software interface to graphics hardware
  - a graphics programming library
  - a standard for 3D graphics
- At the lowest possible level it still allows device independence
  - OpenGL is partly implemented in software and partly in hardware depending on the device
  - No high-level modelling operations, etc.

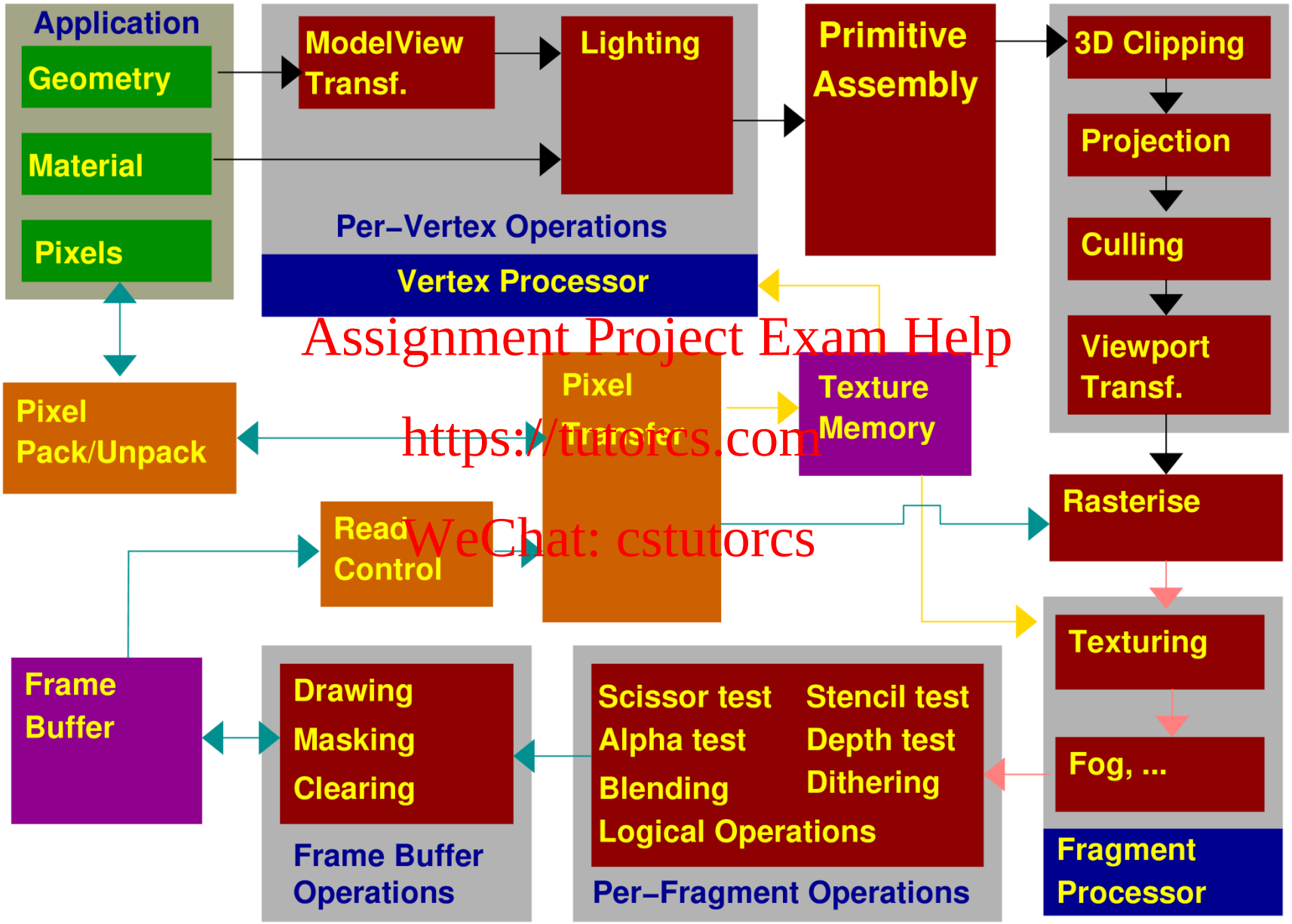
Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

OpenGL History

OpenGL Release	GLSL Release	Year	Features
1.0	---	1992	Fixed-function Pipeline
1.1 ~ 1.5	---	1997 ~ 2003	
2.0	1.10	2004	vertex shaders and fragment shaders
2.1	1.20	2006	
3.0 ~ 3.2	1.30 ~ 1.50	2008 ~ 2009	Deprecated features; Geometry shaders from 3.2.
3.3	3.30	2010	
4.0, 4.1	4.00, 4.10	2010	Tessellation shaders
4.2	4.20	2011	
4.3	4.30	2012	Compute shaders
4.4	4.40	2013	
4.5	4.50	2014	
4.6	4.60	2017	

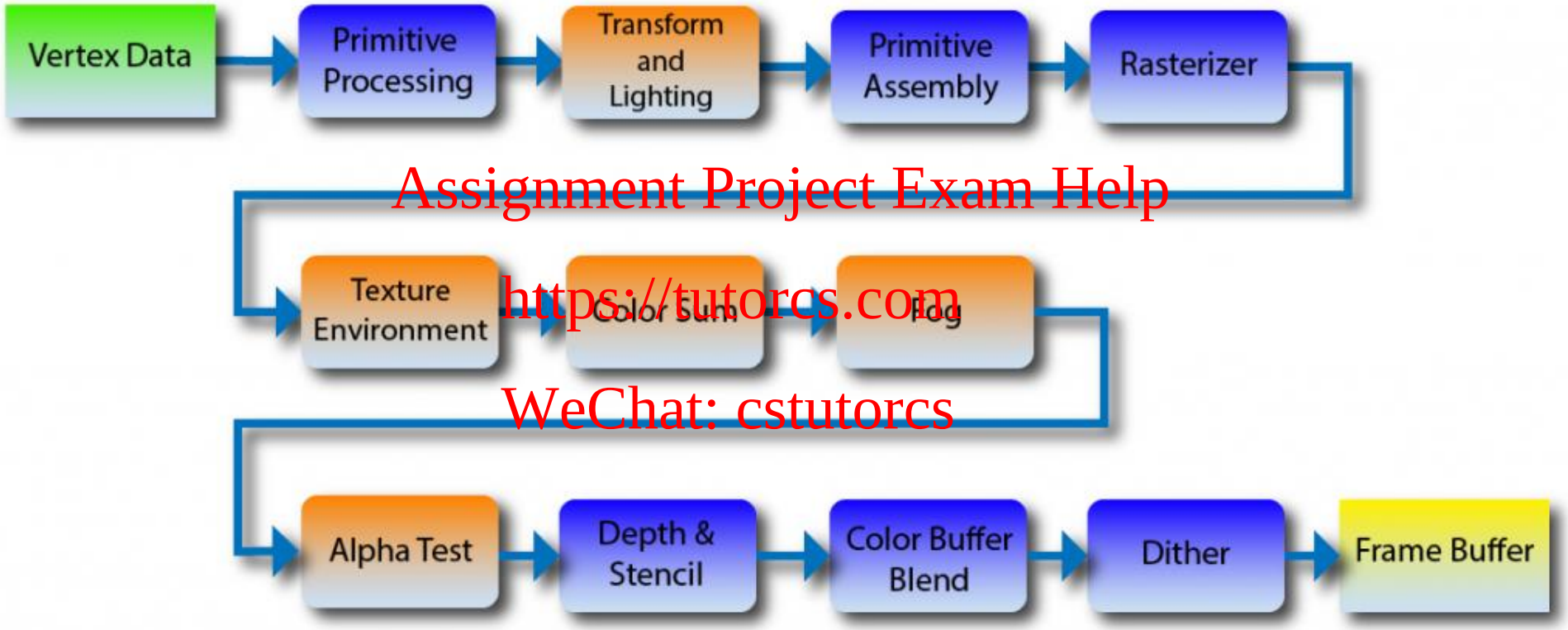
Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

The OpenGL Pipeline



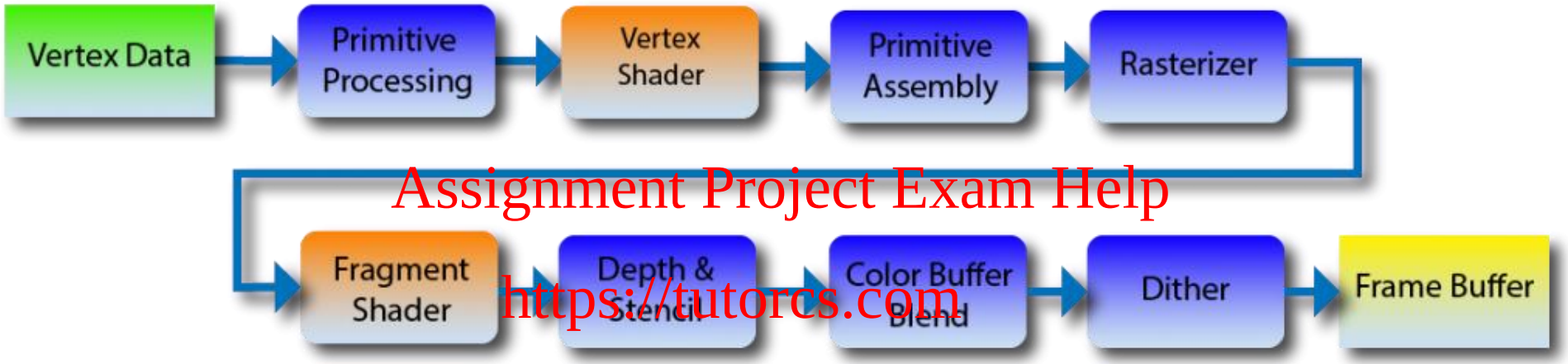
Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

The OpenGL Pipeline (Ver < 2.0)



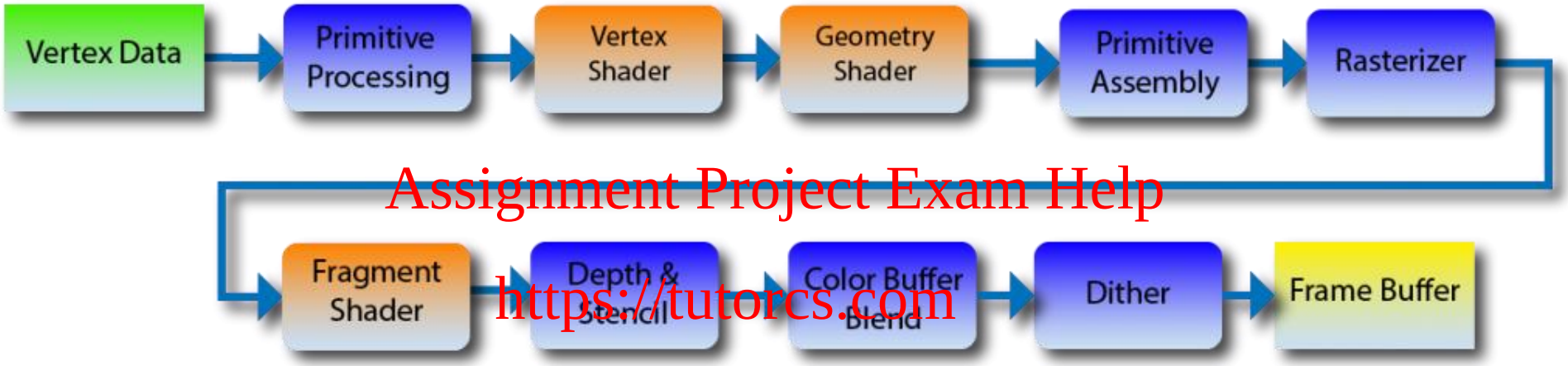
Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

The OpenGL Pipeline (Ver = 2.0)



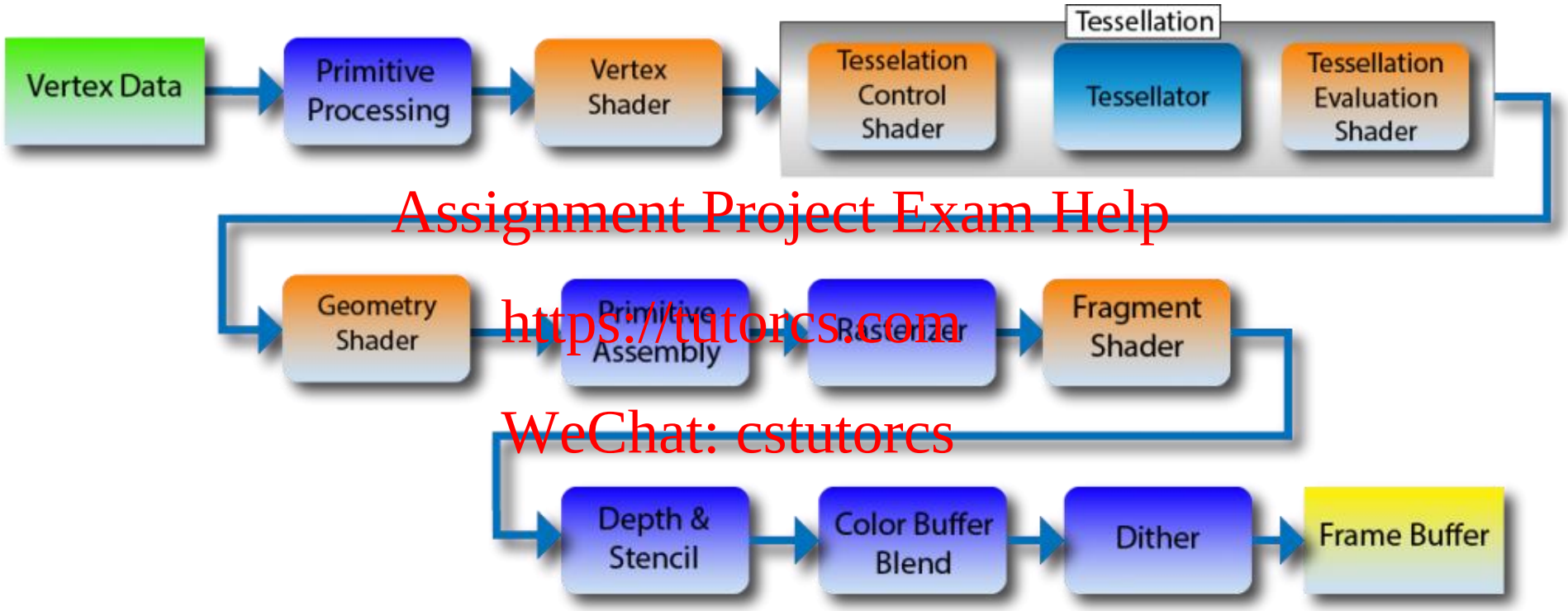
Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

The OpenGL Pipeline (Ver = 3.2)



Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

The OpenGL Pipeline (Ver = 4.0)

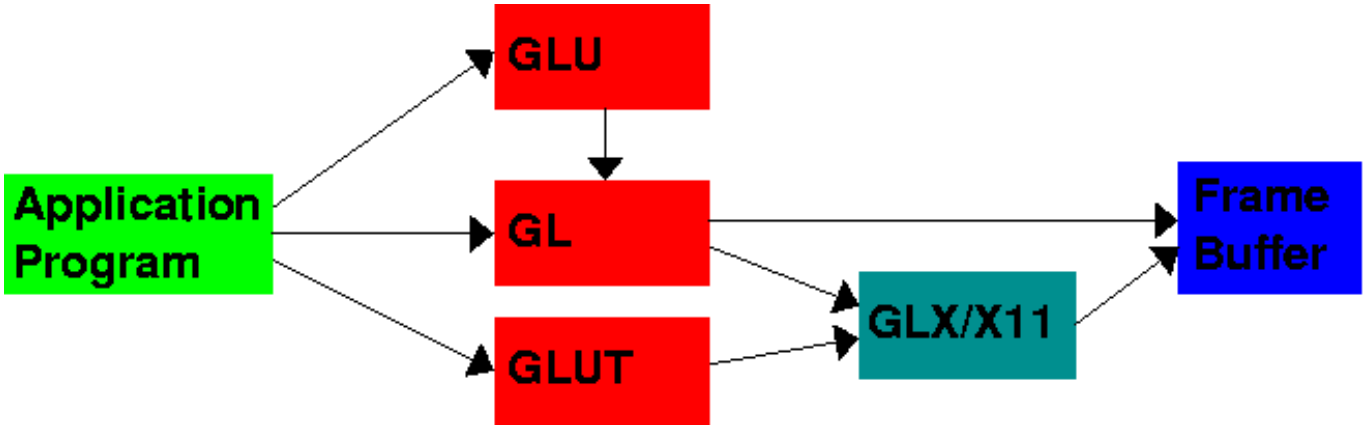


Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

OpenGL Components

- Components of the OpenGL interface:
  - GL: core OpenGL functions
  - GLU: graphics utility library
    (a variety of graphics accessory functions, e.g. gluLookAt)
  - GLUT: OpenGL Utility Toolkit
    (interface to windowing system via xlib; alternatives: glib+GTK, QT; helpers for creating common objects, e.g. spheres, the teapot)
  - GLX: low-level interface to X11
    (different interfaces for other platforms: glw for windows)

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs



Java OpenGL (JOGL)

- Java OpenGL (JOGL) is a wrapper library that allows OpenGL to be used in the Java programming.
- JOGL 1.1.1 gives full access to the APIs in the OpenGL 2.0 specification and limited access to GLU NURBS, providing rendering of curved lines and surfaces via the traditional GLU APIs.
- JOGL 2.0 provides full access to the APIs in the OpenGL 1.3 - 3.0, 3.1 - 3.3 ≥ 4.0 ES 1.x and ES 2.x specification as well as nearly all vendor extensions.
- Newest version (2.3.2) of JOGL can be downloaded from http://jogamp.org/deployment/jogamp-current/archive/

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

Installation of Jogl on Intellij

- Download and install Intellij.
- Download and install the latest Jogl api.
- Set up Jogl as a user library.
- Configure Jogl library in each OpenGL (Jogl) project.

Assignment Project Exam Help

- All downloads and install instruction are free available from related official sites.

https://tutorcs.com

- More detail about installation can be found in the file available from learningcentral.

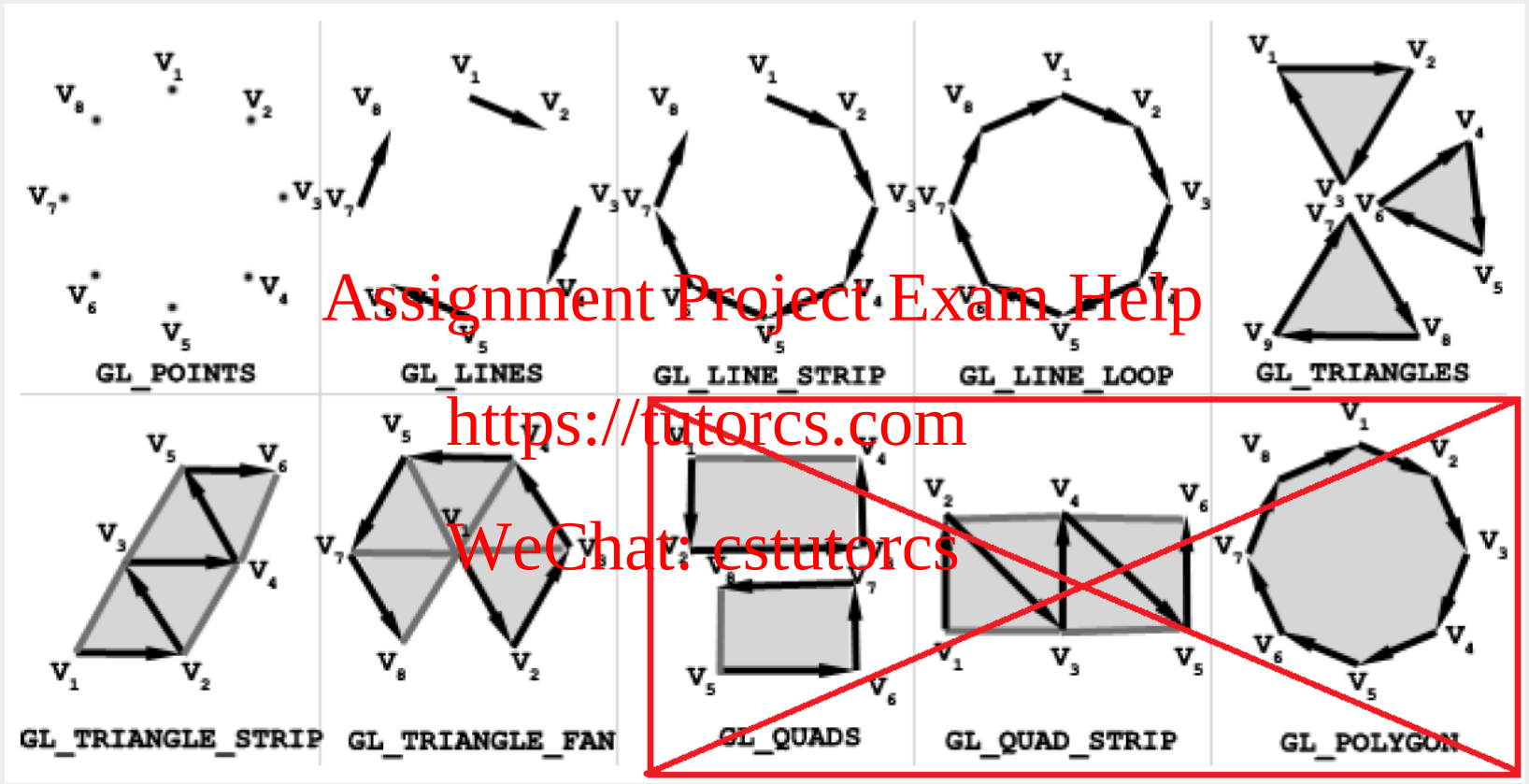
WeChat: cstutorcs

Basic OpenGL Coding Framework

- Configure OpenGL
  - Create window, Display mode
- OpenGL state initialisation
  - Set background colour, View positions, ……
  - Compile and link shader programs
- Set up Display Function
  - Render the scene
- Set up Reshape Function
  - resize the view window and recompute projection matrices
- Process Event loop

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

OpenGL Geometric Primitives



Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL C Program (1)

```
#include <GL/glew.h>
#include <GL/freeglut.h>
// Define: number of Vertex Array Objects,
// number of Vertex Buffer Objects,
// number of Vertices

const GLuint numVAOs = 1, numVBOs = 1;
const GLuint numVertices = 1;

// Specify the ids of points, buffers,
// and the vertex attribute position
// in the vertex shader program
GLuint idPoint = 0, idBuffer = 0;
GLuint vPosition = 0;

// Declare VAOs and VBOs
GLuint VAOs[numVAOs];
GLuint VBOs[numVBOs];
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL C Program (2)

```
// Define: Vertex shader program, and Fragment shader program
const GLchar* srcVShader =
        "#version 330 core\n"
        "layout(location = 0) in vec4 vPosition;"
        "void main()"
        "{"
        "       gl_Position = vPosition;"
        "};";

const GLchar* srcFShader =
        "#version 330 core\n"
        "out vec4 fColor;"
        "void main()"
        "{"
        "       fColor = vec4(1.0, 0.0, 0.0, 1.0);"
        "}";
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL C Program (3)

```
void init(void) // initialisation
{
   //Define vertices coordinates
  GLfloat vertices[numVertices][2] = {
          {0.0f, 0.0f}
   };

   //Generate vertex array objects (VAOs), and
   //Bind a VAO, i.e., initialise this VAO
   // A second binding is needed later to use it
   glGenVertexArrays(numVAOs, VAOs);
   glBindVertexArray(VAOs[idPoints]);

   //Generate vertex buffer objects (VBOs), and
   //Bind a VBO, i.e., initialise this VBO.
   glGenBuffers(numVBOs, VBOs);
   glBindBuffer(GL_ARRAY_BUFFER, VBOs[idBuffer]);
   //The Data is then pooled into the buffer
   glBufferData(GL_ARRAY_BUFFER, sizeof(vertices),
                vertices, GL_STATIC_DRAW);
  //Specify the location and data format of the
  //array of vertex attributes for rendering
   glVertexAttribPointer(vPosition, 2, GL_FLOAT,
                GL_FALSE, 0, (void*)(0));
   glEnableVertexAttribArray(vPosition);
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL C Program (4)

```
    //Create a shader program
    GLuint program = glCreateProgram();

    //Compile and attach vertex shader
    //into the program
    GLuint shader = glCreateShader(GL_VERTEX_SHADER);
    glShaderSource(shader, 1, &srcVShader, NULL);
    glCompileShader(shader);
    glAttachShader(program, shader);
    glDeleteShader(shader);

    //Compile and attach fragment shader
    //into the program
    shader = glCreateShader(GL_FRAGMENT_SHADER);
    glShaderSource(shader, 1, &srcFShader, NULL);
    glCompileShader(shader);
    glAttachShader(program, shader);
    glDeleteShader(shader);

    //Link and use the shader program
    glLinkProgram(program);
    glUseProgram(program);
}
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL C Program (5)

```
// display the scene
void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);

    glPointSize(5);

    //Bind VAO again to use it
    glBindVertexArray(VAOs[idPoint]);
    glDrawArrays(GL_POINTS, 0, numVertices);

    glutSwapBuffers();
}

// resize the view window,
// and recompute projection matrices
void reshape(int width, int height){};
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL C Program (6)

```
int main(int argc, char** argv) {
    // Initialise GLUT
    glutInit(&argc, argv);
    glutInitDisplayMode( GLUT_RGBA | GLUT_DOUBLE );
    glutInitWindowSize(512, 512);
    // Create display window
    glutCreateWindow(argv[0]);
    // OpenGL Version and profile
    glutInitContextVersion(3, 3);
    glutInitContextProfile(GLUT_CORE_PROFILE);

    // Deal with OpenGL extensions issues
    glewExperimental = GL_TRUE;
    if( GLEW_OK != glewInit() )
        exit(EXIT_FAILURE);

    init();

    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutMainLoop(); // Start GLUT event loop
    return 0;
}
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (1)

```
// Import some packages
import java.nio.FloatBuffer;

import com.jogamp.nativewindow.WindowClosingProtocol;
import com.jogamp.newt.opengl.GLWindow;
import com.jogamp.opengl.GL3;
import com.jogamp.opengl.GLAutoDrawable;
import com.jogamp.opengl.GLCapabilities;
import com.jogamp.opengl.GLEventListener;
import com.jogamp.opengl.GLProfile;
import com.jogamp.opengl.util.FPSAnimator;

// Import GL constant
import static com.jogamp.opengl.GL3.*;
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (2)

```
public class Simple implements GLEventListener {
        private GLWindow window; // Declare a canvas
        final FPSAnimator animator=new FPSAnimator(60, true);
        // Define: number of Vertex Array Objects,
        // number of Vertex Buffer Objects,
        // number of Vertices
        // Specify the ids of points, buffers,
        // and the vertex attribute position
        // in the vertex shader program
        private int idPoint = 0, numVAOs = 1;
        private int idBuffer = 0, numVBOs = 1;
        private int vPosition = 0;

        private final int numVertices = 1;

       // Declare VAOs and VBOs
        private int[] VAOs = new int[numVAOs];
        private int[] VBOs = new int[numVBOs];
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (3)

```
private String[] srcVShader =
        { "#version 330 core \n"
        + "layout(location = 0) in vec4 vPosition;"
        + "void main()"
        + "{"
        + "     gl_Position = vPosition;"
        + "}" };

private String[] srcFShader =
        { "#version 330 core\n"
        + "out vec4 fColor;"
        + "void main()"
        + "{"
        + "     fColor = vec4(1.0, 0.0, 0.0, 1.0);"
        + "}" };
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (4)

```
public Simple() {
        GLProfile glp = GLProfile.get(GLProfile.GL3);
        GLCapabilities caps = new GLCapabilities(glp);
        window = GLWindow.create(caps);
        animator.add(window);

        // Listen for openGL events
        window.addGLEventListener(this);
        window.setDefaultCloseOperation(
                WindowClosingProtocol.WindowClosingMode.
                DISPOSE_ON_CLOSE); //Exit when click close
        window.setSize(500, 500); // set the window size
        window.setTitle("Simple Graphics"); // window title
        window.setVisible(true); // Display the frame
        animator.start();
}
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (5)

```
public void init(GLAutoDrawable drawable) {
        // Get the GL pipeline object
        GL3 gl = drawable.getGL().getGL3();

        //Define the vertex coordinates
        float[] vertexArray = { 0.0f, 0.0f };
        //wrap the vertex array into a FloatBuffer.
        FloatBuffer vertices = FloatBuffer.wrap(vertexArray);

        // Generate vertex array objects (VAOs), and
        // Bind a VAO, i.e., initialise this VAO.
        // A second binding is needed later to use it
        gl.glGenVertexArrays(numVAOs, VAOs, 0);
        gl.glBindVertexArray(VAOs[idPoint]);

        // Generate vertex buffer objects (VBOs), and
        // Bind a VBO, i.e., initialise this VBO.
        // The Data is then pooled into the buffer
        gl.glGenBuffers(numVBOs, VBOs, 0);
        gl.glBindBuffer(GL_ARRAY_BUFFER, VBOs[idBuffer]);
        gl.glBufferData(GL_ARRAY_BUFFER, vertexArray.length *
                (Float.SIZE / 8),vertices, GL_STATIC_DRAW);

        gl.glVertexAttribPointer(vPosition,2,GL_FLOAT,false,0,0L);
        gl.glEnableVertexAttribArray(vPosition);
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (6)

```
        // Create a shader program
        int program = gl.glCreateProgram();

        // Compile and attach vertex shader into the program
        int shader = gl.glCreateShader(GL_VERTEX_SHADER);
        gl.glShaderSource(shader, 1, srcVShader, null);
        gl.glCompileShader(shader);
        gl.glAttachShader(program, shader);
        gl.glDeleteShader(shader);

        // Compile and attach fragment shader into the program
        shader = gl.glCreateShader(GL_FRAGMENT_SHADER);
        gl.glShaderSource(shader, 1, srcFShader, null);
        gl.glCompileShader(shader);
        gl.glAttachShader(program, shader);
        gl.glDeleteShader(shader);

        // Link and use the shader program
        gl.glLinkProgram(program);
        gl.glUseProgram(program);
    }
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

A Simple OpenGL Java Program (7)

```
public void display(GLAutoDrawable drawable) {
        GL3 gl = drawable.getGL().getGL3();

        gl.glClear(GL_COLOR_BUFFER_BIT);
        gl.glPointSize(5);

        gl.glDrawArrays(GL_POINTS, 0, numVertices);

        }

public void reshape(GLAutoDrawable drawable, int x, int y,
                int width, int height) {
        }

public void dispose(GLAutoDrawable drawable) {
        System.exit(0);
        }

public static void main(String[] args) {
                new Simple();
        }
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

Summary

- What is the underlying model for the OpenGL library?
  - What are the components of OpenGL?
- Basic OpenGL programming with C++ or Java.
  - Describe the OpenGL coding framework.

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs