JOGLによる3次元グラフィクスのプログラム (第 7 回)

氏名　越智 秀次
クラス　理 科 一 類 11 組
学生証番号 J4-170235

□課題7.1 - 8.3節 例 1: JOGLによる直線群の描画(2次元)-Lines.java

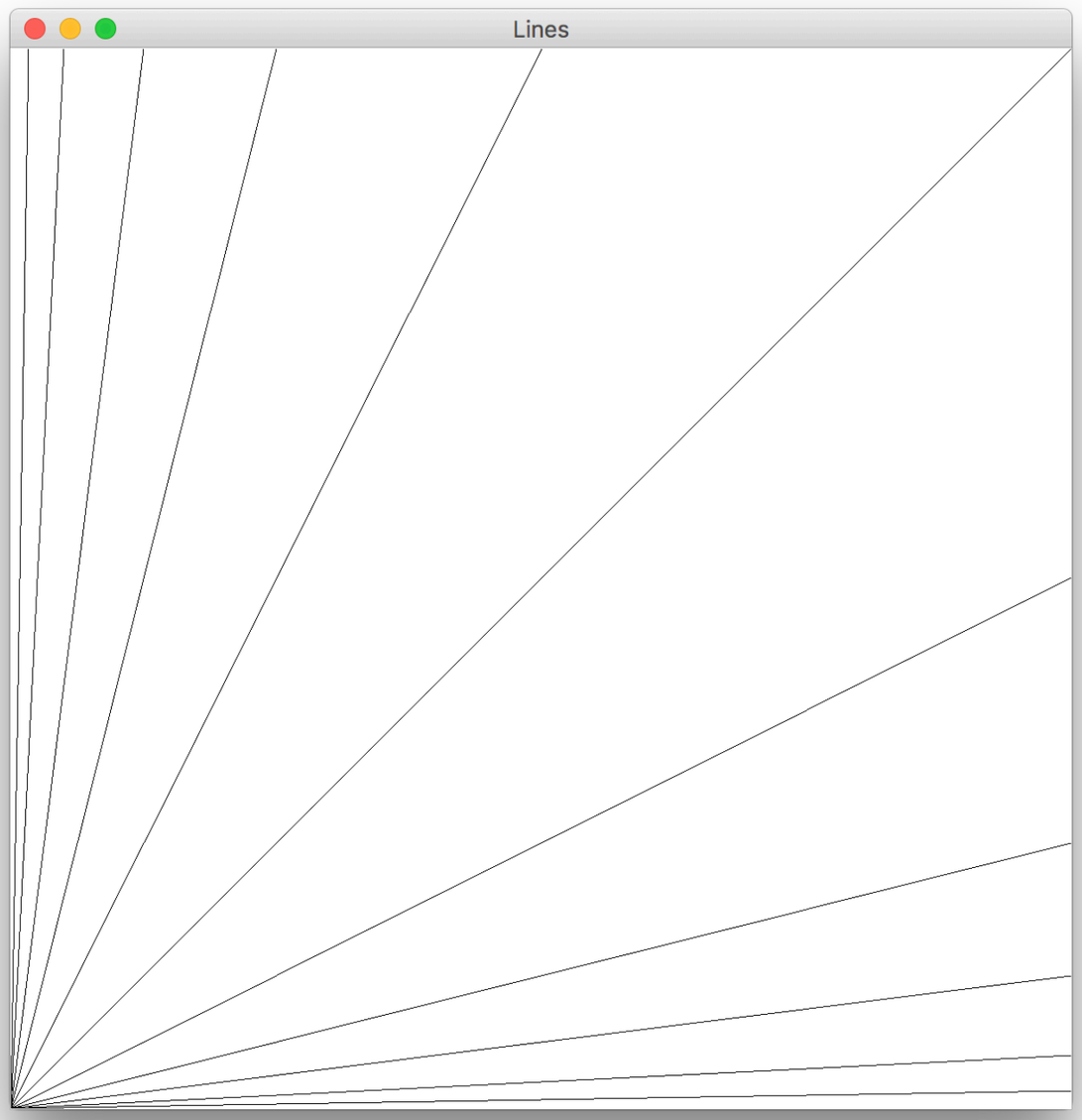
○プログラムリスト

略

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java Lines
```

○実行結果



○考察

これまでウィンドウ座標系ではウィンドウの左上が原点であったが、このJOGLによるグラフィクスではウィンドウの左下が原点となっていることに注意する。

□課題7.2 - 10.3節 例 1: 透視投影による立方体の描画-CubePosition.java

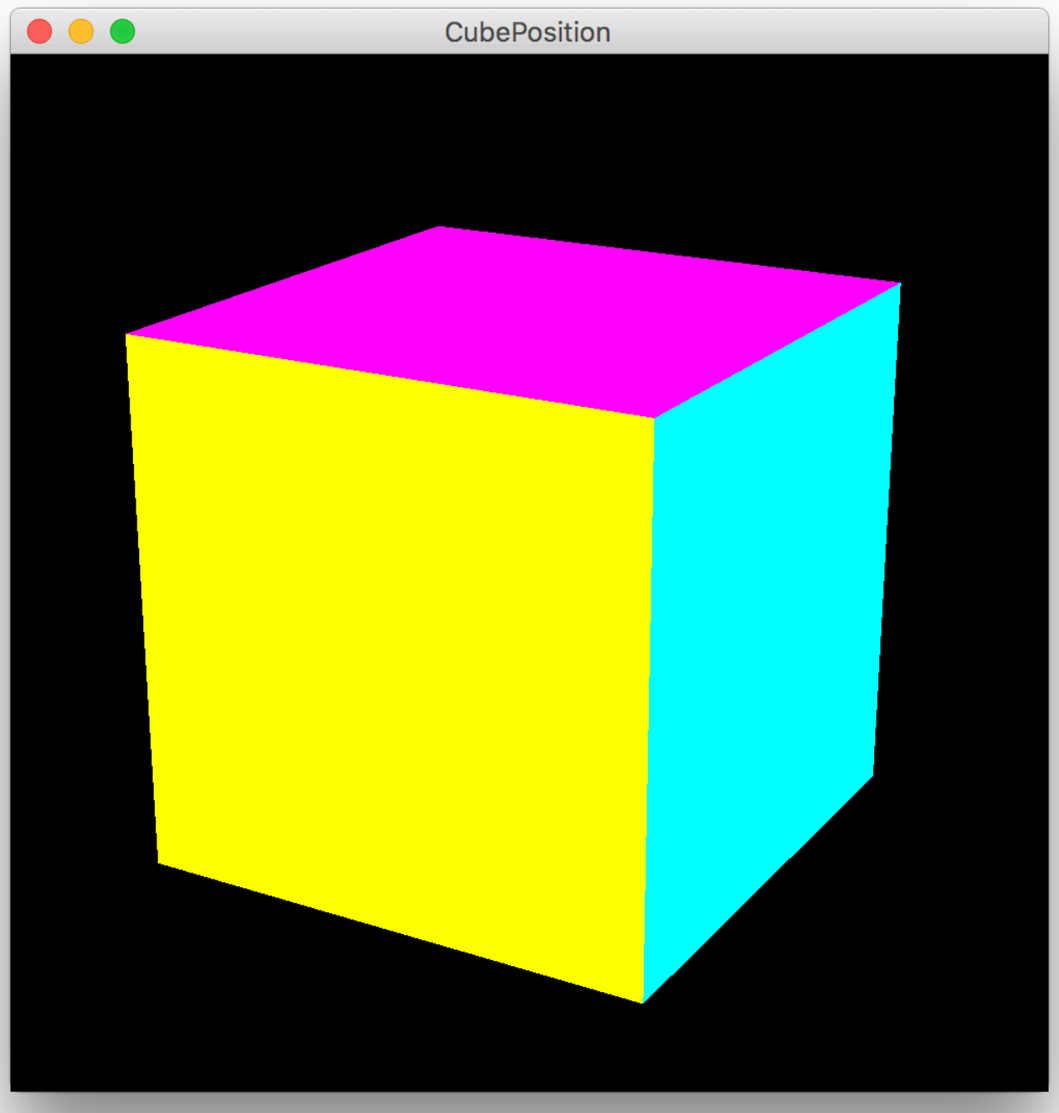
○プログラムリスト

略

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java CubePosition
```

○実行結果



○考察

プログラムとグラフィクスパイプラインの対応を確認しながらプログラムを書いた。

□課題7.3 - 10.3節 例 2: 多角形としての立方体の描画-CubePolygon.java

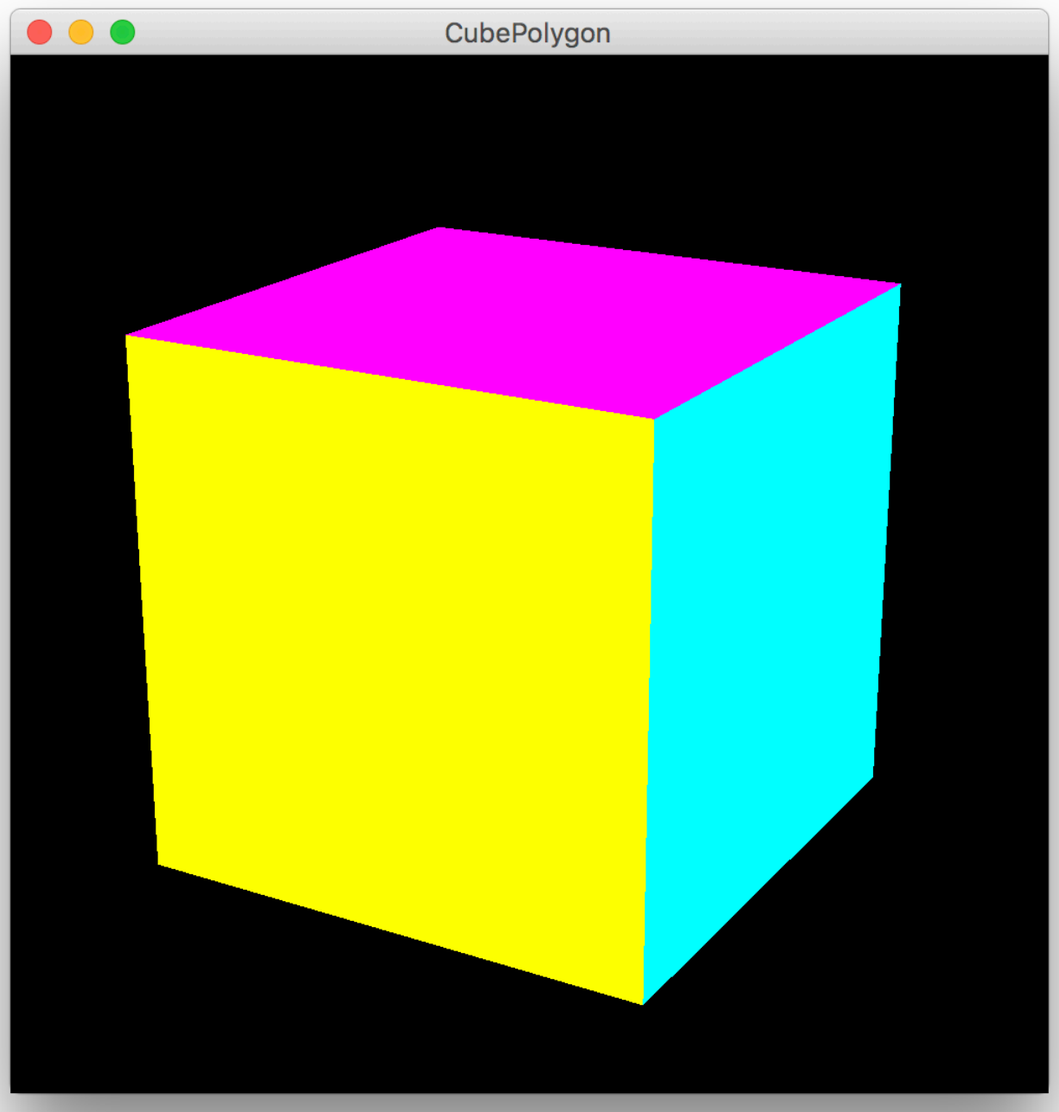
○プログラムリスト

略

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java CubePolygon
```

○実行結果



○考察

四角形を複数使って立体の描画をする(CubePosition)場合と、１つの多角形によって立体の描画をする(CubePolygon)場合には、描画命令の開始と終了のタイミングが異なることに特に注意が必要である。

▫課題7.4 - 章末課題: 視点位置の変更-CubePosition.java

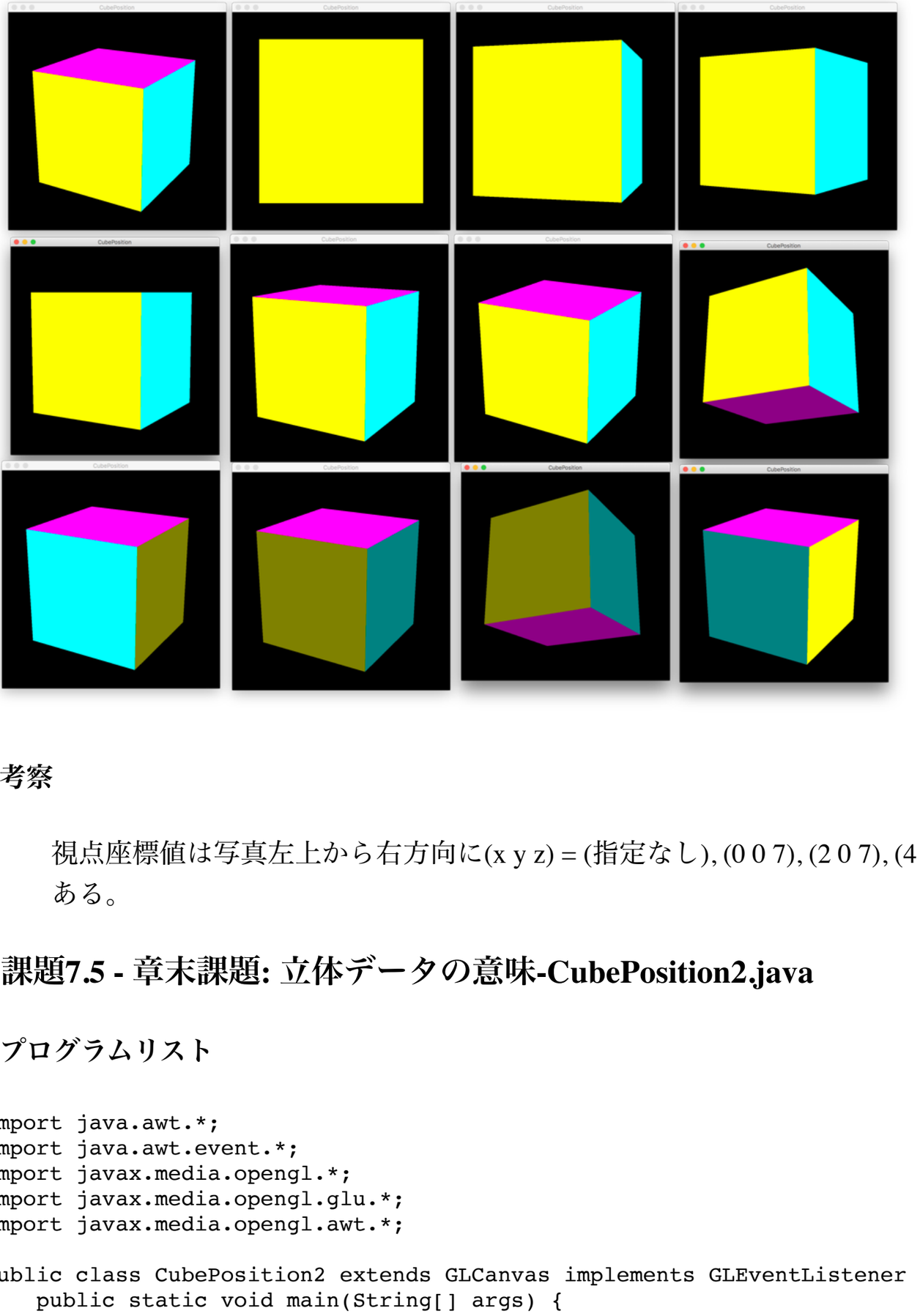
○プログラムリスト

例題プログラムに同じ

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java CubePosition x y z (x, y, z は視点座標値、後述。)
```

○実行結果



○考察

視点座標値は写真左上から右方向に(x y z) = (指定なし), (0 0 7), (2 0 7), (4 0 7), (4 1 7), (4 2 7), (4 3 7), (4 -3 7), (7 3 -4), (-4 3 -7), (-4 -3 -7), (-7 3 4) である。

▫課題7.5 - 章末課題: 立体データの意味-CubePosition2.java

○プログラムリスト

```
import java.awt.*;
import java.awt.event.*;
import javax.media.opengl.*;
import javax.media.opengl.glu.*;
import javax.media.opengl.awt.*;

public class CubePosition2 extends GLCanvas implements GLEventListener {
    public static void main(String[] args) {
      if (args.length == 3)
        (new CubePosition2("CubePosition2", args)).showFrame();
      else
        (new CubePosition2("CubePosition2")).showFrame();
  }

  private Frame f;
  protected GL2 gl;
  protected GLU glu;
  protected double eye_x = 4.0;
  protected double eye_y = 3.0;
  protected double eye_z = 7.0;

  protected CubePosition2(String name) {
      super();
      setSize(500, 500);
      addGLEventListener(this);

      f = new Frame(name);
      f.add(this);
      f.pack();
      f.addWindowListener(new WindowAdapter() {
          public void windowClosing(WindowEvent e) {
              System.exit(0);
          }
      });
```

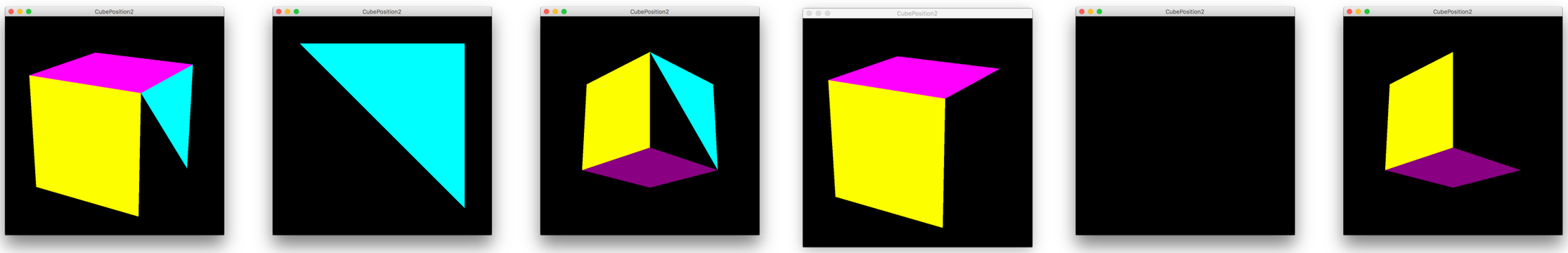
```
    }

    protected CubePosition2(String name, String[] args) {
        this(name);
        eye_x = Double.parseDouble(args[0]);
        eye_y = Double.parseDouble(args[1]);
        eye_z = Double.parseDouble(args[2]);
    }

    protected void showFrame() {
        f.setVisible(true);
    }

    public void init(GLAutoDrawable drawable) {
        gl = drawable.getGL().getGL2();
        glu = new GLU();
        gl.glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
        gl.glEnable(GL2.GL_DEPTH_TEST);
        gl.glEnable(GL2.GL_CULL_FACE);
    }

    public void reshape(GLAutoDrawable drawable, int x, int y, int w, int h) {
        final double fieldOfView = 25.0, near = 1.0, far = 20.0;
        double aspect = (double) w / (double) h;

        gl.glViewport(0, 0, w, h);
        gl.glMatrixMode(GL2.GL_PROJECTION);
        gl.glLoadIdentity();
        glu.gluPerspective(fieldOfView, aspect, near, far);
        gl.glMatrixMode(GL2.GL_MODELVIEW);
        gl.glLoadIdentity();
        glu.gluLookAt(eye_x, eye_y, eye_z, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
    }

    public void display(GLAutoDrawable drawable) {
        gl.glClear(GL2.GL_COLOR_BUFFER_BIT | GL2.GL_DEPTH_BUFFER_BIT);
        cubeDisplay();
    }

    static float[][] vertices = { {-1.0f, -1.0f, -1.0f}, {1.0f, -1.0f, -1.0f}, {1.0f, 1.0f, -1.0f}, {-1.0f, 1.0f, -1.0f}, {-1.0f, -1.0f, 1.0f}
    static int[][] faces = { {5, 6, 2, 1}, {2, 3, 7, 6}, {4, 5, 6, 7}, {0, 4, 7, 3}, {0, 1, 5, 4}, {0, 3, 2, 1} };
    static float[][] colors = { {0.0f, 1.0f, 1.0f}, {1.0f, 0.0f, 1.0f}, {1.0f, 1.0f, 0.0f}, {0.0f, 0.5f, 0.5f}, {0.5f, 0.0f, 0.5f}, {0.5f, 0.5

    protected void cubeDisplay() {
        gl.glBegin(GL2.GL_QUADS);
        for (int i = 0; i < faces.length; i++) {
            gl.glColor3fv(colors[i], 0);
            for (int j = 0; j < faces[i].length; j++)
              gl.glVertex3fv(vertices[faces[i][j]], 0);
        }
        gl.glEnd();
        gl.glFlush();
    }

    public void dispose(GLAutoDrawable drawable) {

    }
}
```

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java CubePosition x y z (x, y, zは視点座標値、後述。)
```

○実行結果

最初の3つはfacesの最初の配列を{1,2,5,6}、後半の3つは{5,6,2,1}としている。視点座標はそれぞれ左から(x,y,z)=(指定なし), (7 0 0), (7 -4 7)としている。



○考察

面の法線ベクトルの向きと頂点を指定する順番が右ねじの関係にあると考えるとわかりやすい。最初の配列を{5,6,2,1}にした場合はその面の法線ベクトルが立方体の内側を向いているので、その面は黒くなってしまう。

□課題7.6 - 章末課題: 立体データの意味-Tetrahedron.java

○プログラムリスト

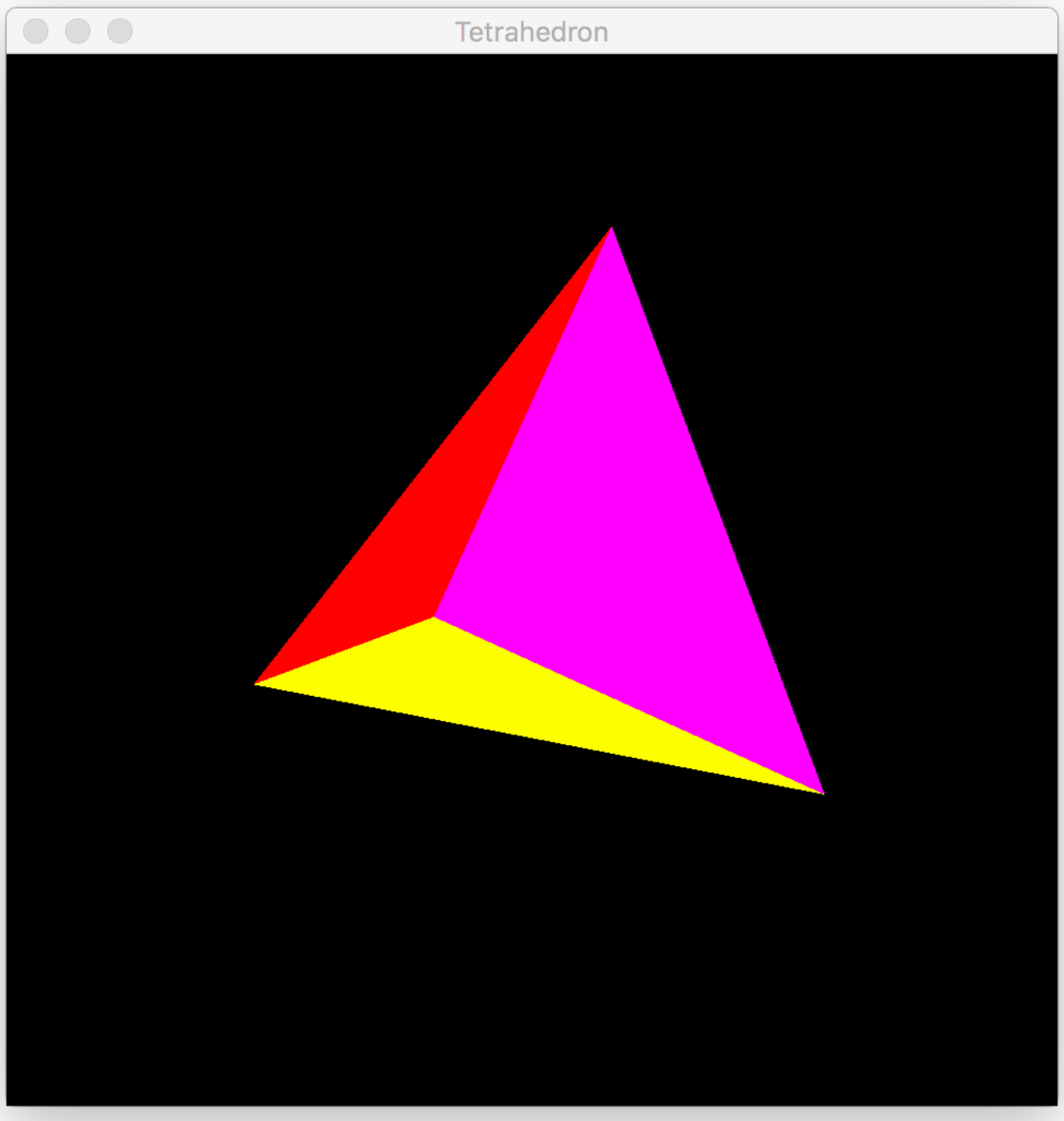
```
import java.awt.*;
import java.awt.event.*;
import javax.media.opengl.*;
import javax.media.opengl.glu.*;
import javax.media.opengl.awt.*;
```

```
public class Tetrahedron extends GLCanvas implements GLEventListener {
    public static void main(String[] args) {
        if (args.length == 3)
          (new Tetrahedron("Tetrahedron", args)).showFrame();
        else
          (new Tetrahedron("Tetrahedron")).showFrame();
    }

    private Frame f;
    protected GL2 gl;
    protected GLU glu;
    protected double eye_x = 4.0;
    protected double eye_y = 3.0;
    protected double eye_z = 7.0;

    protected Tetrahedron(String name) {
        super();
        setSize(500, 500);
        addGLEventListener(this);

        f = new Frame(name);
        f.add(this);
        f.pack();
        f.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
    }

    protected Tetrahedron(String name, String[] args) {
        this(name);
        eye_x = Double.parseDouble(args[0]);
        eye_y = Double.parseDouble(args[1]);
        eye_z = Double.parseDouble(args[2]);
    }

    protected void showFrame() {
        f.setVisible(true);
    }

    public void init(GLAutoDrawable drawable) {
        gl = drawable.getGL().getGL2();
        glu = new GLU();
        gl.glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
        gl.glEnable(GL2.GL_DEPTH_TEST);
        gl.glEnable(GL2.GL_CULL_FACE);
    }

    public void reshape(GLAutoDrawable drawable, int x, int y, int w, int h) {
        final double fieldOfView = 25.0, near = 1.0, far = 20.0;
        double aspect = (double) w / (double) h;
        gl.glViewport(0, 0, w, h);
        gl.glMatrixMode(GL2.GL_PROJECTION);
        gl.glLoadIdentity();
        glu.gluPerspective(fieldOfView, aspect, near, far);
        gl.glMatrixMode(GL2.GL_MODELVIEW);
        gl.glLoadIdentity();
        glu.gluLookAt(eye_x, eye_y, eye_z, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
    }

    public void display(GLAutoDrawable drawable) {
        gl.glClear(GL2.GL_COLOR_BUFFER_BIT | GL2.GL_DEPTH_BUFFER_BIT);
        cubeDisplay();
    }

    static float[][] vertices = { {-1.0f, -1.0f, -1.0f}, {1.0f, 1.0f, -1.0f}, {1.0f, -1.0f, 1.0f}, {-1.0f, 1.0f, 1.0f} };
    static int[][] faces = { {0, 1, 2}, {1, 3, 2}, {0, 2, 3}, {0, 3, 1} };
    static float[][] colors = { {0.0f, 1.0f, 1.0f}, {1.0f, 0.0f, 1.0f}, {1.0f, 1.0f, 0.0f}, {1.0f, 0.0f, 0.0f} };

    protected void cubeDisplay() {
        gl.glBegin(GL2.GL_TRIANGLES);
        for (int i = 0; i < faces.length; i++) {
            gl.glColor3fv(colors[i], 0);
            for (int j = 0; j < faces[i].length; j++)
              gl.glVertex3fv(vertices[faces[i][j]], 0);
        }
        gl.glEnd();
        gl.glFlush();
    }

    public void dispose(GLAutoDrawable drawable) {

    }
}
```

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java Tetrahedron -4 7 7
```

○実行結果



○考察

正四面体であるとわかるような視点位置を探すのにかなり試行錯誤した。

□課題7.7 - 自主課題: 多角形としての正四面体の描画-TetraPolygon.java

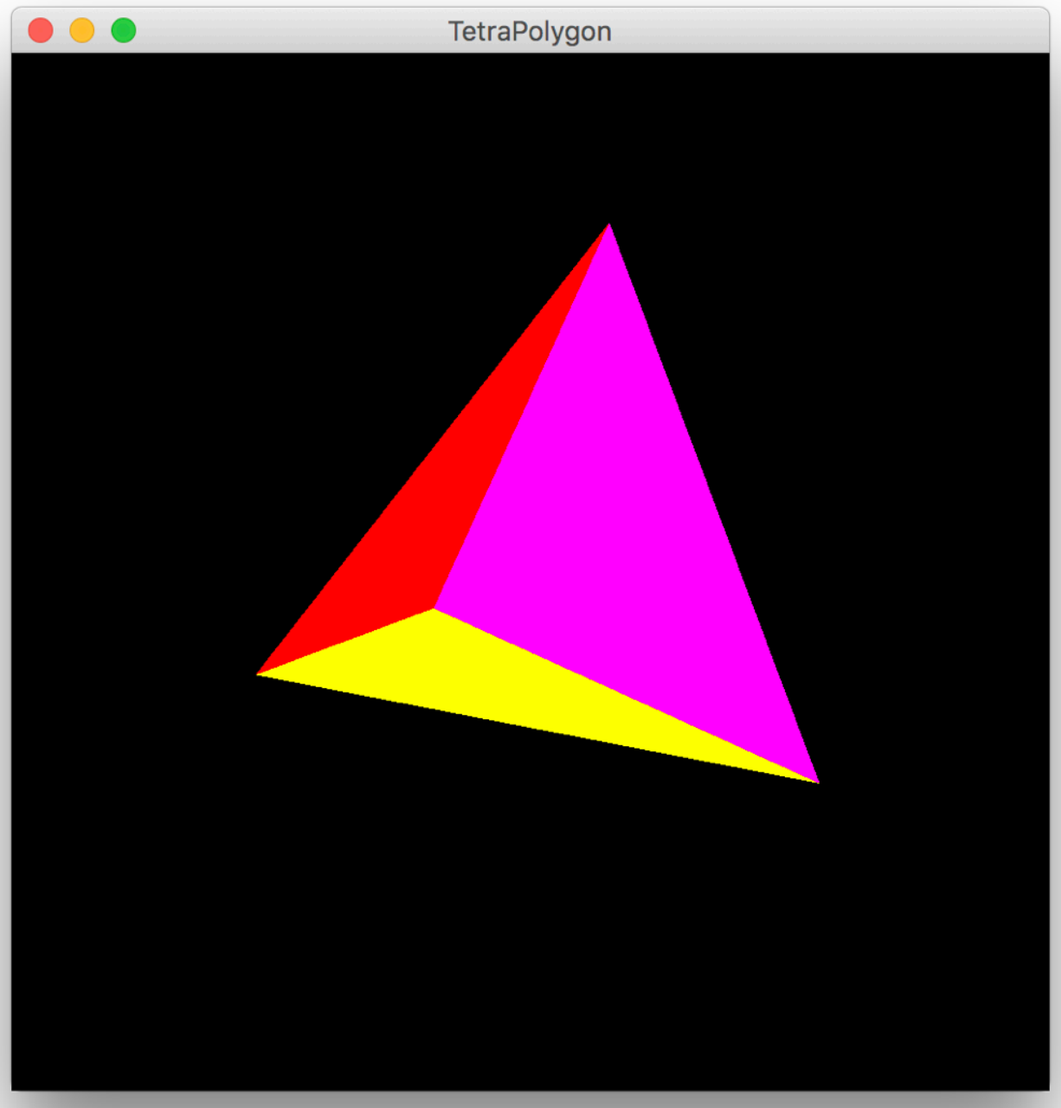
○プログラムリスト

```
import javax.media.opengl.*;

public class TetraPolygon extends Tetrahedron {
    public static void main(String[] args) {
        if (args.length == 3)
          (new TetraPolygon("TetraPolygon", args)).showFrame();
        else
          (new TetraPolygon("TetraPolygon")).showFrame();
    }

    protected TetraPolygon(String name) {
        super(name);
    }

    protected TetraPolygon(String name, String[] args) {
        super(name, args);
    }

    protected void cubeDisplay() {
        for (int i = 0; i < faces.length; i++) {
            gl.glBegin(GL2.GL_POLYGON);
            gl.glColor3fv(colors[i], 0);
            for (int j = 0; j < faces[i].length; j++)
              gl.glVertex3fv(vertices[faces[i][j]], 0);
            gl.glEnd();
        }
        gl.glFlush();
    }
}
```

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap08&10 ochihideji$ java TetraPolygon -4 7 7
```

○実行結果



○考察

Tetrahedronを拡張することで多角形としての正四面体の描画のプログラムも容易に作成できるので、ついでに作成した。

□課題や授業に関して

○レポート作成に要した時間

4時間程度

○特に苦労した点

今までの章と描画の方式が大きく変わったので、プログラムの内容を理解するのに非常に苦労した。本課題の作成につき、細かい部分の理解は後回しにして3次元グラフィクスの実現に必要な最低限の部分の理解に努めた。

○授業についての感想や希望

投影法などについてさらに理解を深めたいと思います。