隠線・隠面消去 (第 11 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11.1 - 14.4節 例 1: 複数円錐のワイヤーフレーム表示(線画表示) - WireCones.java

○プログラムリスト

略

○考察

円錐底面と円錐側面では描画に用いている要素が異なっている。円錐底面では閉じた折れ線を描画するGL_LINE_LOOPが用いられている。

□課題11.2 - 14.4節 例 2: 複数円錐の回転プログラム - WireConesRotate.java

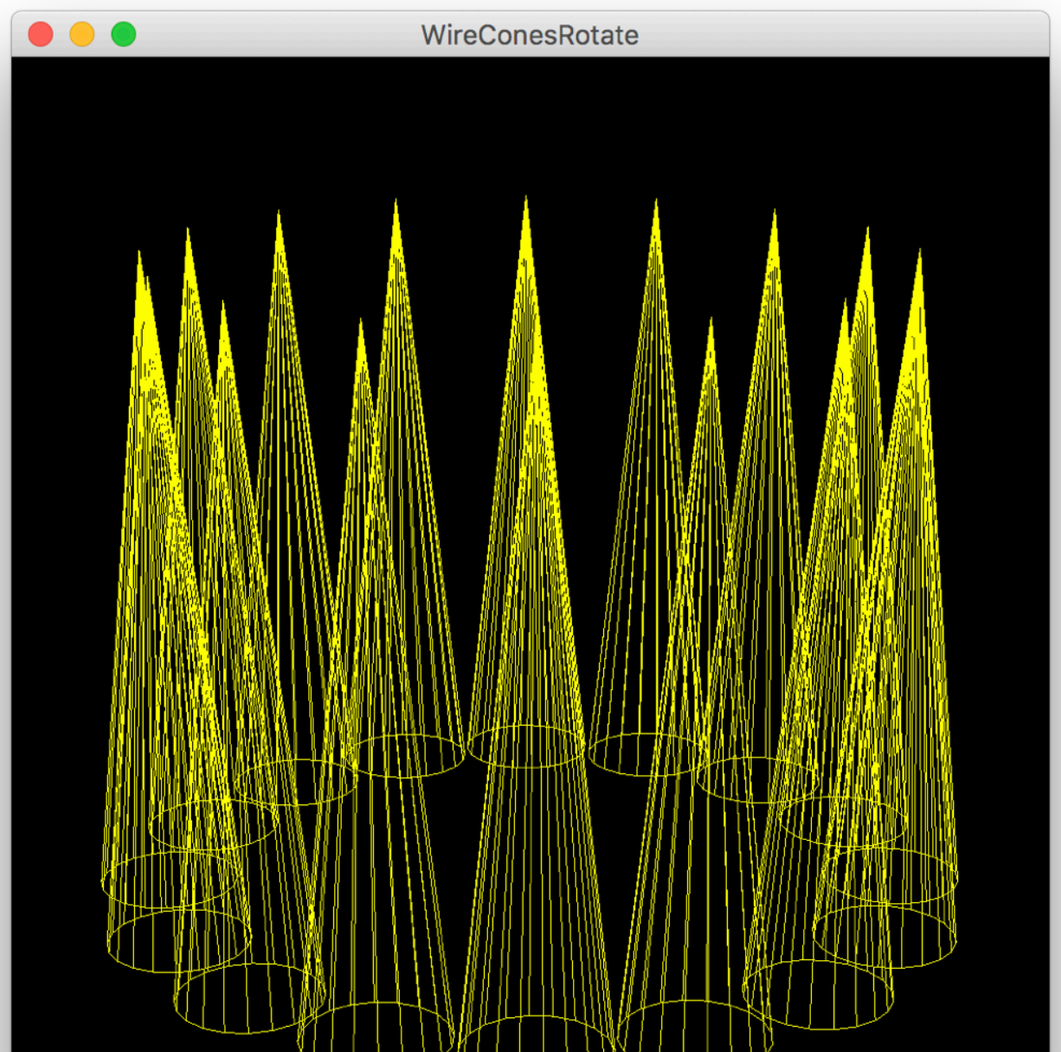
○プログラムリスト

略

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap14 ochihideji$ java WireConesRotate
```

○実行結果



○考察

隠線消去をしない場合、回転させながら観察しなければ表示画像を把握するのが難しい。

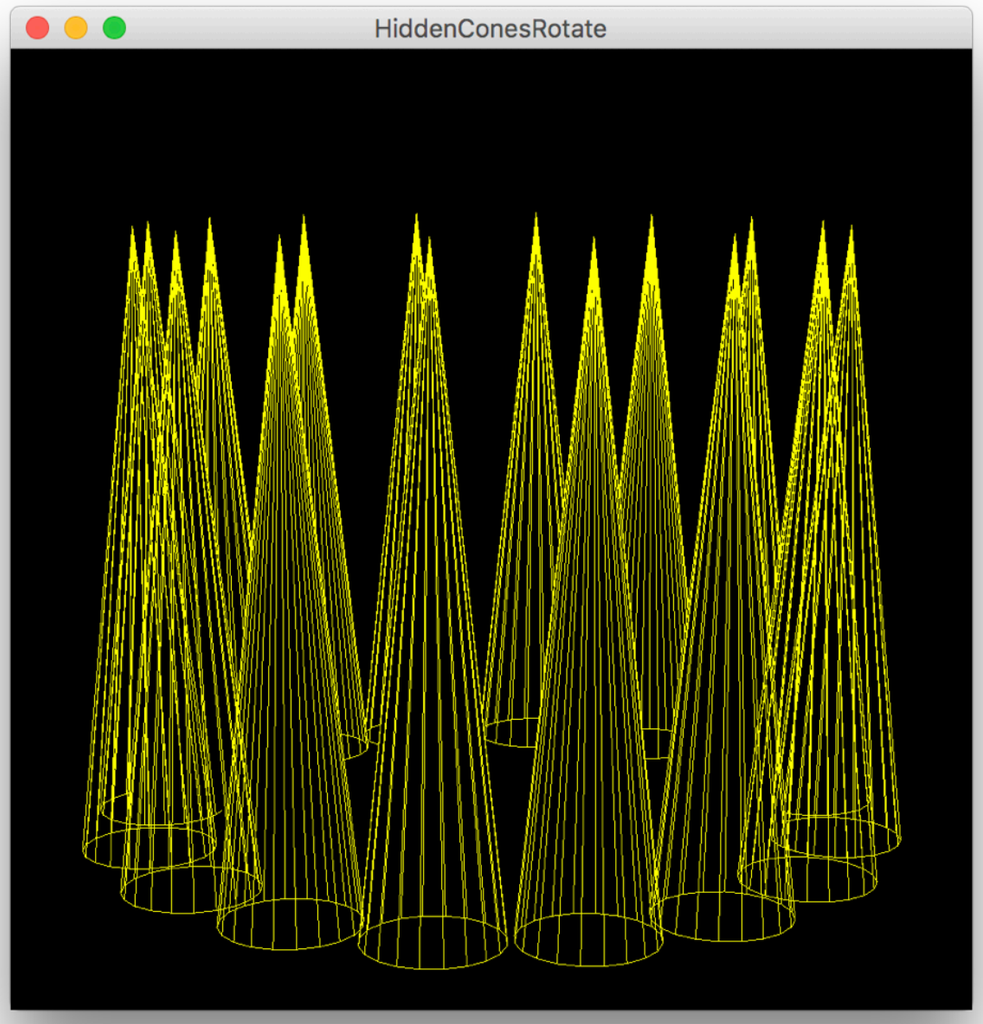
□課題11.3 - 14.4節 例 3: 複数円錐の隠線消去 - HiddenCones.java

○プログラムリスト

略

○考察

面のoffset量を増やすと中途半端に隠線が消去された不自然な画像となる。



▫課題11.4 - 14.4節 例 4: 隠線消去の回転プログラム - HiddenConesRotate.java

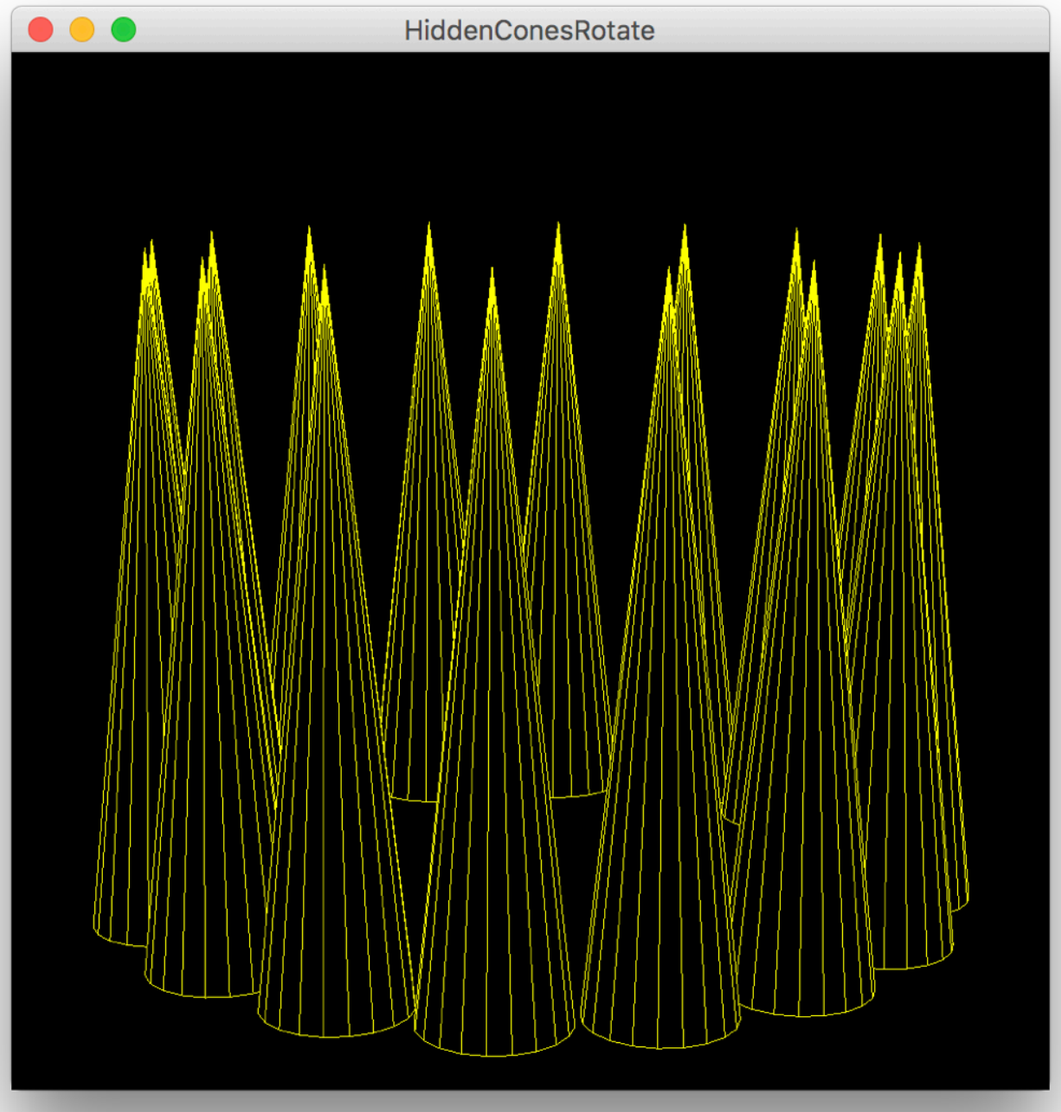
○プログラムリスト

略

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap14 ochihideji$ java HiddenConesRotate
```

○実行結果



○考察

基本的にマウスによる回転プログラムはObjectRotateを拡張したプログラムでobjectとして回転させたい立体を登録するだけで容易に実現できる。

▫課題11.5 - 章末課題 - 隠面消去の効果

○プログラムリスト

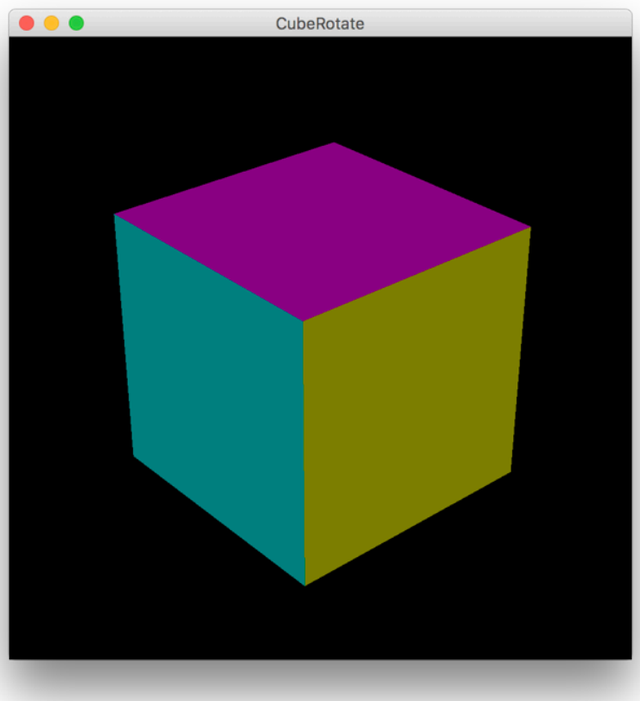
ObjectRotate.javaの63,64行目のgl.glEnable(GL2.GL_DEPTH_TEST); および　gl.glEnable(GL2.GL_CULL_FACE); をコメントアウトしただけであるので割愛する。

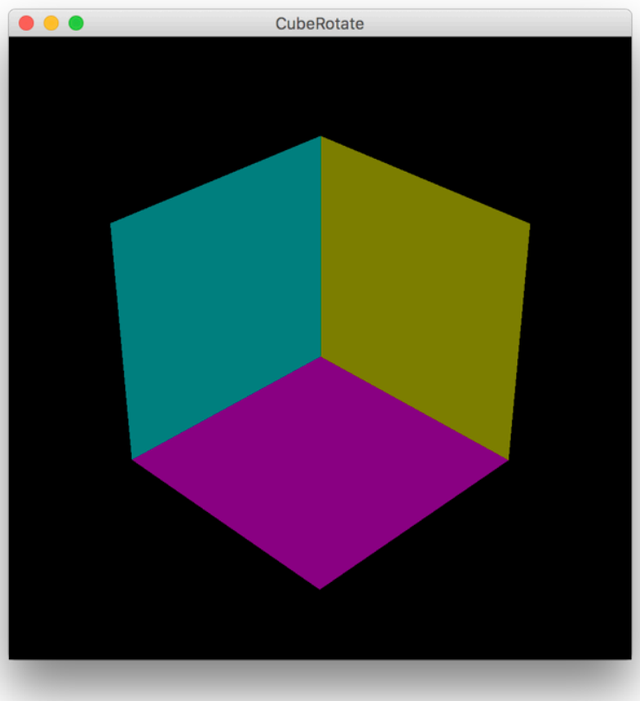
○実行コマンド

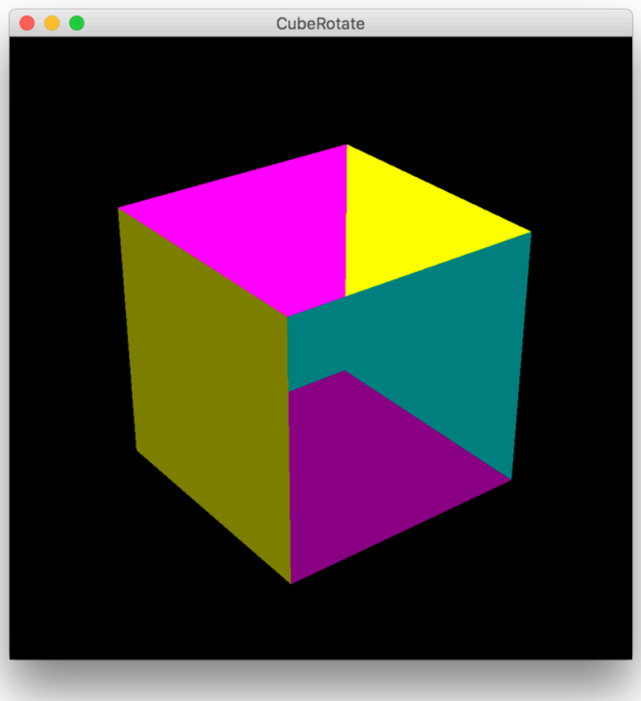
```
ochihidejinoMacBook-Pro:Chap14 ochihideji$ java CubeRotate
```

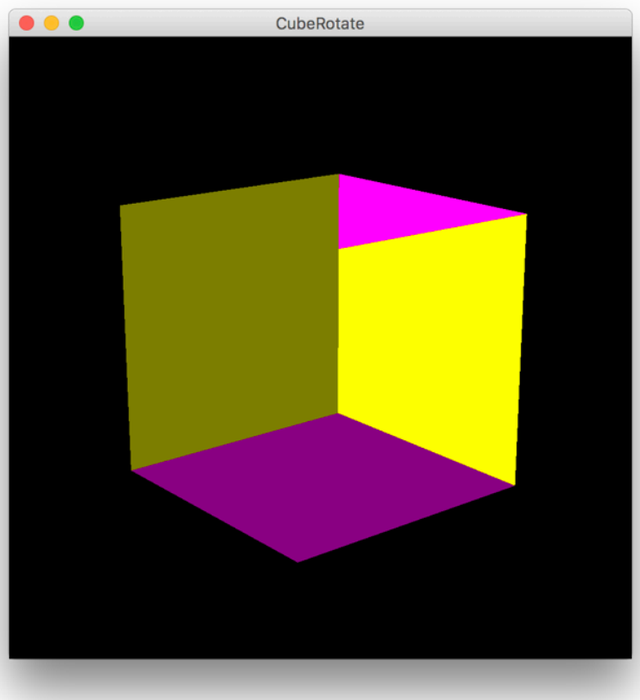
○実行結果

1番目は立方体を裏側から見た時の図。立方体が正しく描画されている。
2番目は立方体を表側(視点の初期位置近傍)から見た図。明るいシアン、マゼンダ、イエローの面が消えている。
3番目、4番目はさらに別角度から見た図。これらの図をよく観察してみると、暗いイエロー ＞ 暗いマゼンダ ＞ 暗いシアン ＞ 明るいイエロー ＞ 明るいマゼンダ　の順で面が優先して描か
また、どの角度から観察しても明るいシアンの面は見えなかった。









○考察

実行結果から、隠面消去を実行しない場合はCubeクラスのcubeFaces配列で登録した順番に面が奥から手前へと描かれていくとわかる。例えば明るいシアンの面は一番奥側に描かれるので、隠面消去を実行しない場合にはそのほかの面に隠れて見ることができなくなる。

▫課題11.5 - 章末課題: 立方体の隠れ線消去表示 - HiddenCubeRotate.java

○プログラムリスト

HiddenCubeRotate.java

```
public class HiddenCubeRotate extends ObjectRotate {
    public static void main(String[] args) {
        (new HiddenCubeRotate("HiddenCubeRotate")).showFrame();
    }

    protected HiddenCubeRotate(String name) {
        super(name);
        object = new HiddenCube();
    }
}
```

HiddenCube.java

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

public class HiddenCube extends HiddenPolyhedron {
  private static final float[][] cubeVertices =
```

```
  { { -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 }, { 1.0f, -1.0f,  1.0f },
    {  1.0f,  1.0f,  1.0f }, { -1.0f,  1.0f,  1.0f } };

  private static final int[][] cubeEdges =
  { { 0, 1 }, { 1, 2 }, { 2, 3 }, { 3, 0 }, { 0, 4 }, { 1, 5 }, { 2, 6 },
    { 3, 7 }, { 4, 5 }, { 5, 6 }, { 6, 7 }, { 7, 4 } };

  private static final int[][] cubeFaces =
  { { 1, 2, 6, 5 }, { 2, 3, 7, 6 }, { 4, 5, 6, 7 }, { 0, 4, 7, 3 },
    { 0, 1, 5, 4 }, { 0, 3, 2, 1 } };

  protected HiddenCube() {
      vertices   = cubeVertices;
      faces      = cubeFaces;
      edges      = cubeEdges;
  }
}
```

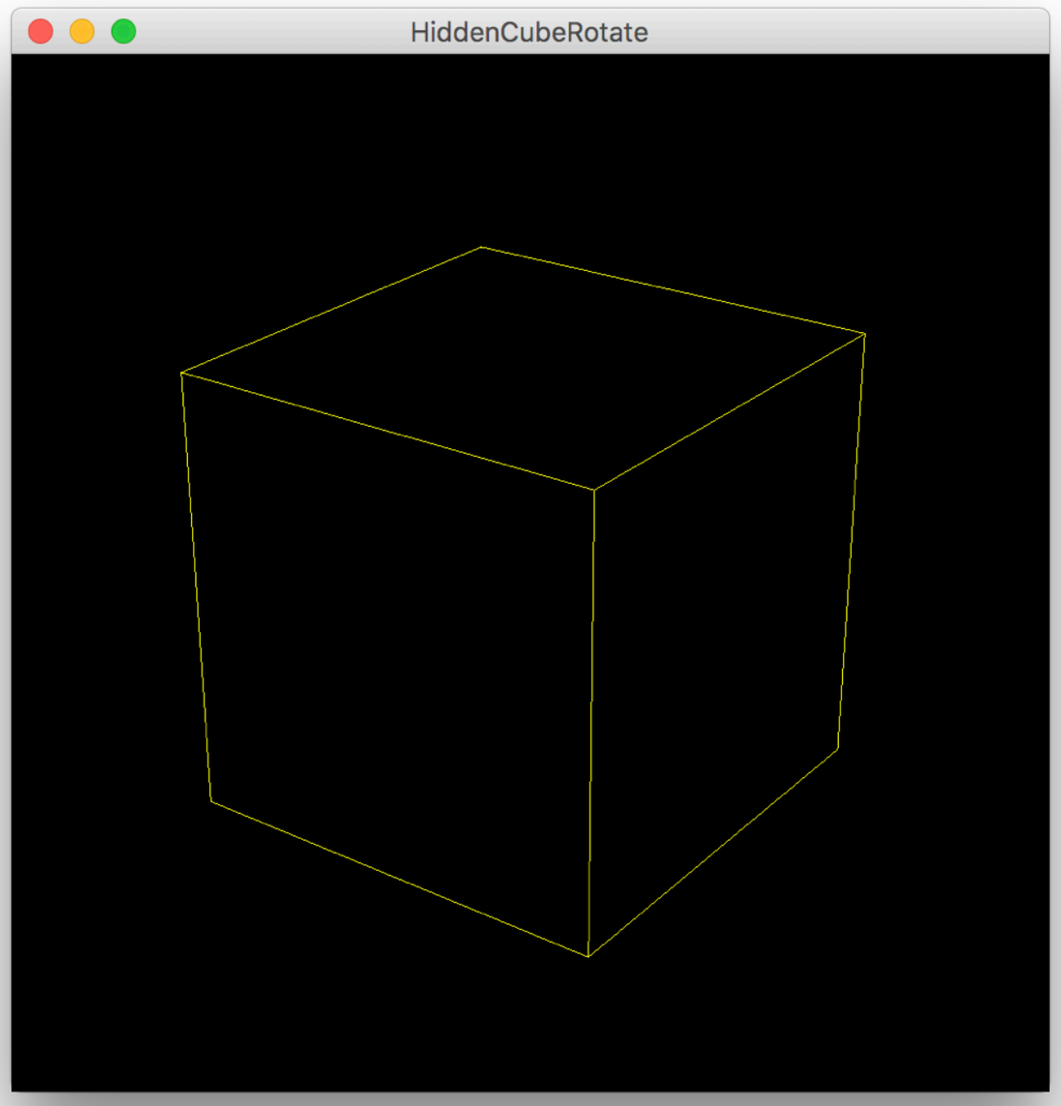
HiddenPolyhedron.java

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

public abstract class HiddenPolyhedron implements DisplayObject {
    protected float[][] vertices;
    protected int[][]   faces;
    protected int[][]   edges;

    public void display(GL2 gl) {
        gl.glEnable(GL2.GL_POLYGON_OFFSET_FILL);
        gl.glPolygonOffset(1.0f, 1.0f);
        float[] background = {0.0f, 0.0f, 0.0f};
        float[] foreground = {1.0f, 1.0f, 0.0f};
        gl.glPushMatrix();
        gl.glColor3fv(foreground, 0);
        displayEdges(gl);
        gl.glColor3fv(background, 0);
        displayFaces(gl);
        gl.glPopMatrix();
        }

    public void displayFaces(GL2 gl) {
        for (int i = 0; i < faces.length; i++) {
            gl.glBegin(GL2.GL_POLYGON);
            for (int j = 0; j < faces[i].length; j++) {
                gl.glVertex3fv(vertices[faces[i][j]], 0);
            }
            gl.glEnd();
        }
    }

    protected void displayEdges(GL2 gl) {
        gl.glBegin(GL2.GL_LINES);
        for (int i = 0; i < edges.length; i++) {
             gl.glVertex3fv(vertices[edges[i][0]], 0);
             gl.glVertex3fv(vertices[edges[i][1]], 0);
        }
        gl.glEnd();
    }

    public int vertices() {
        return vertices.length;
    }
    public float vertex(int id, int j) {
        return vertices[id][j];
    }
    public int faces() {
        return faces.length;
    }
    public int faceVertices(int id) {
        return faces[id].length;
    }
    public float faceVertex(int id, int i, int j) {
        return vertices[faces[id][i]][j];
    }
    public int edges() {
        return edges.length;
    }
    public float edgeVertex(int id, int i, int j) {
        return vertices[edges[id][i]][j];
    }
}
```

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap14 ochihideji$ java HiddenCubeRotate
```

○実行結果



○考察

作成途中でクラス継承などに関して混乱してしまい、HiddenPolyhedronクラスを拡張するというかなり遠回りなプログラムになってしまった。

□課題11.6 - 章末課題: メンガースポンジの隠線消去 - WMSRotate.java

○プログラムリスト

WMSRotate.java

```
public class WMSRotate extends ObjectRotate {
    public static void main(String[] args) {
        if (args.length == 0)
          System.err.println("Usage: java WMSRotate #iteration");
        else
          (new WMSRotate("WMSRotate", Integer.parseInt(args[0]))).showFrame();
    }

    protected WMSRotate(String name, int times) {
        super(name);
        object = new WireMengerSponge(times);
    }
}
```

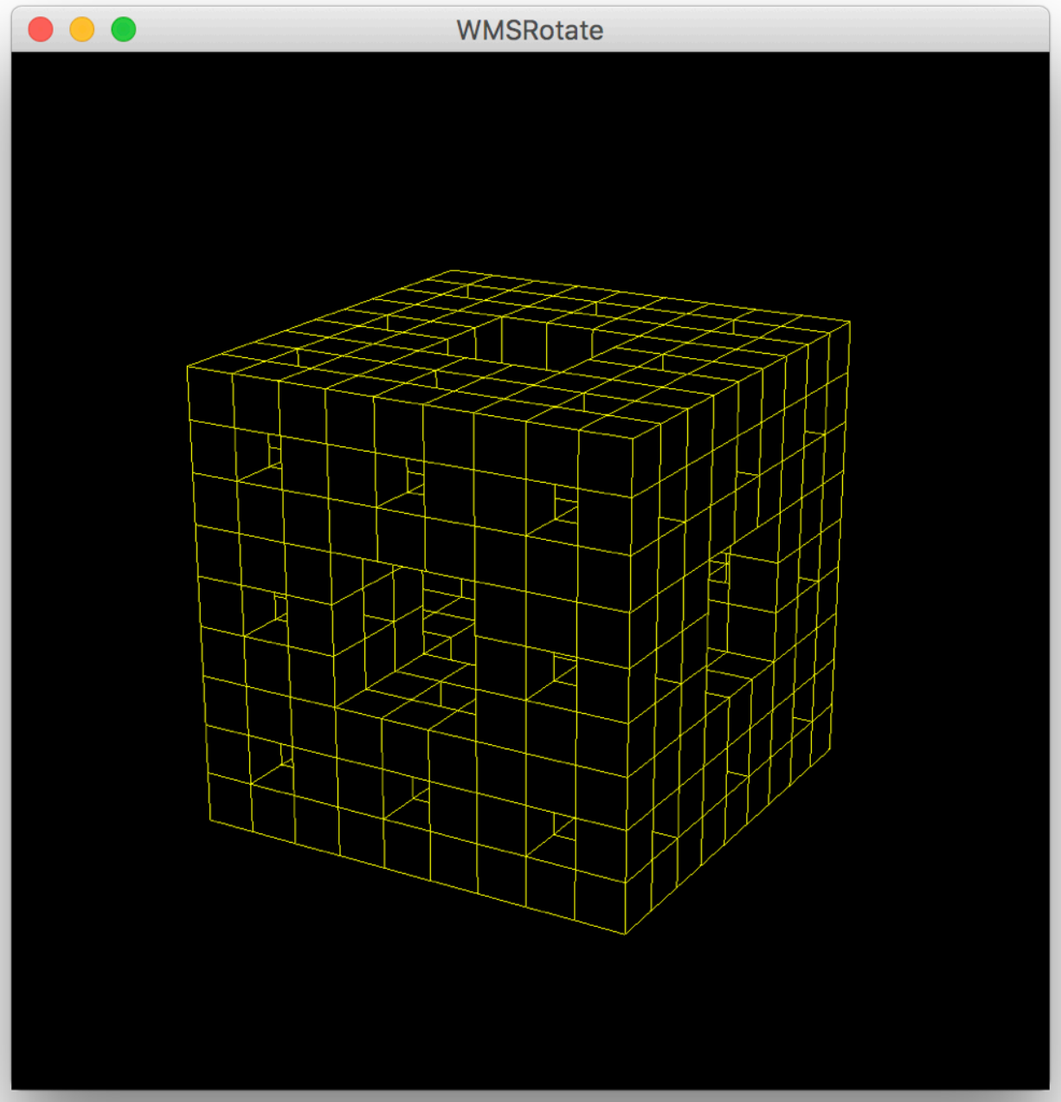
WireMengerSponge.java

```
public class WireMengerSponge extends FractalObject {
    protected WireMengerSponge(int times) {
        super(times);

        primitive = new HiddenCube();
        int nv = primitive.vertices();
        int ne = primitive.edges();
        s = 1.0 / 3.0;
        v = new double[nv+ne][3];
        for (int i = 0; i < nv; i ++)
          for (int j = 0; j < 3; j++)
            v[i][j] = primitive.vertex(i, j) * (1.0 - s);
        for (int i = 0; i < ne; i++)
          for (int j = 0; j < 3; j++)
            v[i+nv][j] = (primitive.edgeVertex(i, 0, j) + primitive.edgeVertex(i, 1, j)) / 2.0 * (1.0 - s);
    }
}
```

○実行コマンド

```
ochihidejinoMacBook-Pro:Chap14 ochihideji$ java WMSRotate 2
```

○実行結果



○考察

primitiveとしてCubeではなくHiddenCubeを登録するだけなので、プログラム的にはさほど難しくない。ただし、このプログラムでは小立方体同士の間に残った線を消去することはできない。

▫課題11.6 - 章末課題: 正十二面体片の隠線消去 - HiddenDodecahedronFractal.java

○プログラムリスト

HiddenDodecahedronFractal.java

```
public class HiddenDodecahedronFractal extends FractalObject {
    protected HiddenDodecahedronFractal(int times) {
        super(times);

        primitive = new HiddenDodecahedron();
        int nv = primitive.vertices();
        float phi = (float)((1.0 + Math.sqrt(5)) / 2.0);
        s = 1.0 / (2.0 + phi);
        v = new double[nv][3];
        for (int i = 0; i < nv; i ++)
          for (int j = 0; j < 3; j++)
            v[i][j] = primitive.vertex(i, j) * (1.0 - s);
    }
}
```

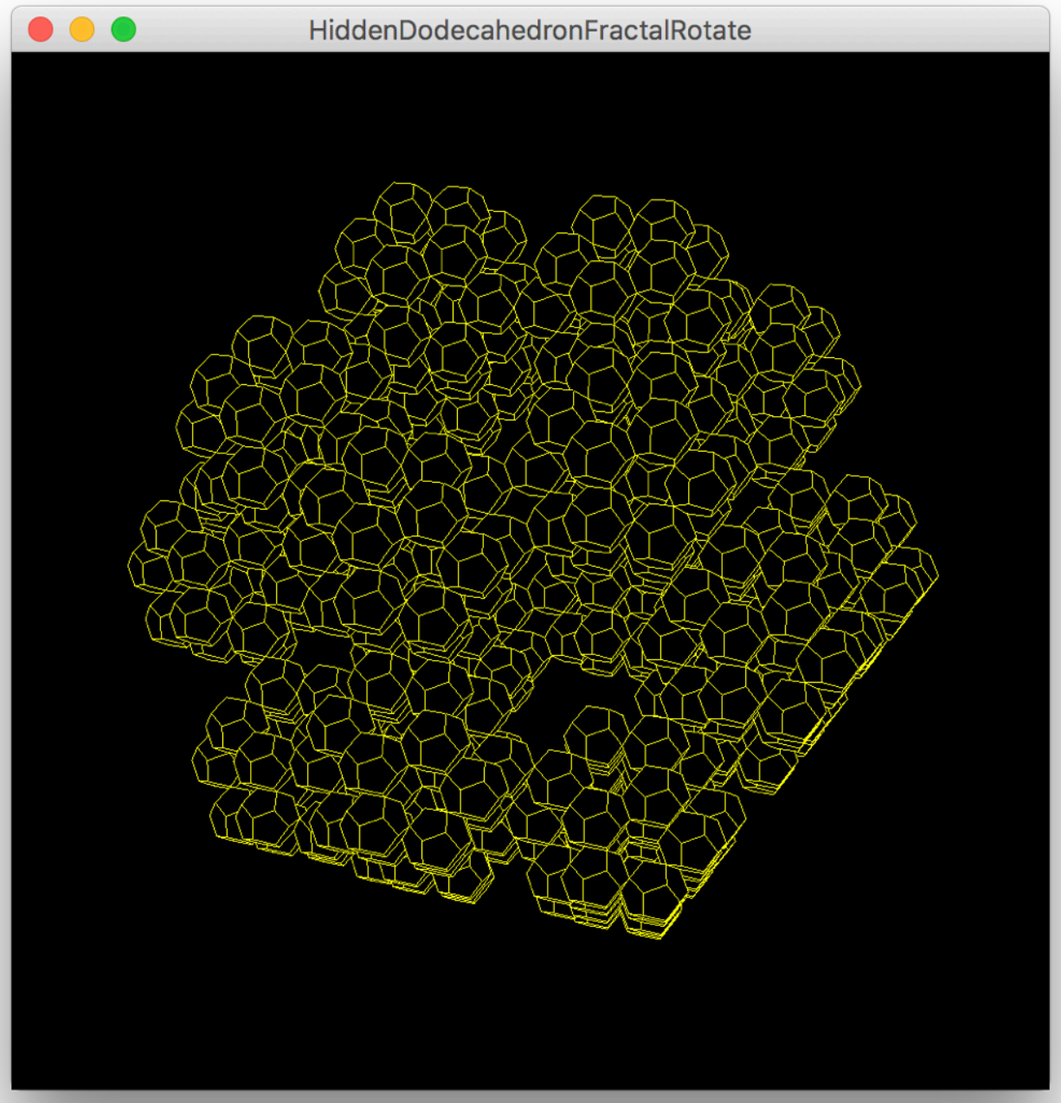
HiddenDodecahedron.java

```
public class HiddenDodecahedron extends HiddenPolyhedron {
    protected static final float phi = (1 + (float)Math.sqrt(5)) / 2;
    private static final float[][] dodecaVertices =
    { {-1/phi, phi, 0}, {1/phi, phi, 0}, {-1, 1, 1}, {1, 1, 1}, {1, 1, -1}, {-1, 1, -1}, {0, 1/phi, phi}, {0, 1/phi, -phi}, {-phi, 0, 1/phi},
    private static final int[][] dodecaFaces =
    { {0, 2, 6, 3, 1}, {1, 3, 9, 10, 4}, {1, 4, 7, 5, 0}, {0, 5, 11, 8, 2}, {6, 2, 8, 14, 12}, {6, 12, 15, 9, 3}, {4, 10, 16, 13, 7}, {5, 7, 1
    private static final int[][] dodecaEdges =
    { {0, 1}, {0, 2}, {0, 5}, {1, 3}, {1, 4}, {2, 6}, {3, 6}, {4, 7}, {5, 7}, {17, 18},
    {16, 19}, {2, 8}, {5, 11}, {3, 9}, {4, 10}, {11, 8}, {9, 10}, {6, 12}, {7, 13}, {8, 14},
    {11, 17}, {9, 15}, {10, 16}, {13, 16}, {13, 17}, {12, 14}, {12, 15}, {14, 18}, {15, 19}, {18, 19} };
    protected HiddenDodecahedron() {
        vertices   = dodecaVertices;
        faces      = dodecaFaces;
        edges      = dodecaEdges;
    }
}
```

○実行コマンド

```
ochihidejinoMacBook-Pro:HiddenRotate ochihideji$ java HiddenDodecahedronFractalRotate 2
```

○実行結果



○考察

立体が複雑なので回転させなければ立体形状を把握することが困難である。

□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

立方体の隠線消去のプログラム作成に苦労した。

○授業についての感想や希望

特になし。