コンピューターグラフィックス基礎 第二回課題

情報メディア創生学類 3 年 202313625 藤川興昌

実行環境

- Ubuntu 22.04.3 LTS
- gcc version 11.4.0

課題 2

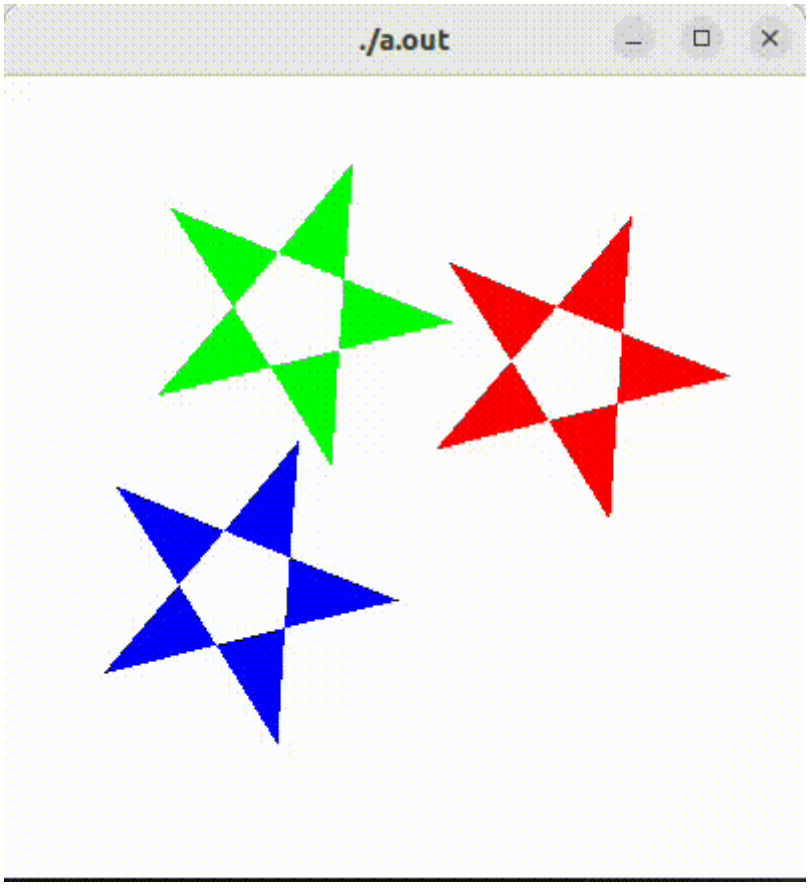
ソースコード

```
#include <math.h>
#include <GL/glut.h> // ���C�u�����p�w�b�_�t�@�C���ǂ݂v���

// ꞔ荏ꞔ`
#ifndef M_PI
#define M_PI 3.14159265358979
#endif

// ꞔfꞔBꞔXꞔvꞔCꞔXꞔgꞔwꞔK
// ꞔ`ꞔ悞ꞔ`ꞔ施ꞔႦ▫▫▫▫AꞔfꞔBꞔXꞔvꞔCꞔXꞔgꞔБꞔč쐬ꞔĂꞔ
// ꞔKꞔvꞔÔꞔ?AꞔꞔꞐꞔᏥꞔꞔĂяoꞔ

#define ID_DRAW_STAR 1 //  glNewList ꞔ▫ꞔŏgꞔpꞔ鄒
ꞔꞔIDꞔBꞔlꞔꞔłꞔ\ꞔÂꞔ

int rotateAngle; // ꞔꞔ]ꞔpꞔxꞔLꞔ^ꞔĂꞔꞔɓꞔ

// ꞔ\ꞔꞔ▫ꞔŋLꞔ
void display(void) {
        glClearColor (1.0, 1.0, 1.0, 1.0);  // ꞔFꞔwꞔ
        glClear(GL_COLOR_BUFFER_BIT);       //
ꞔꞔÐꞔsꞔ

        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();

        glRotated(rotateAngle, 0, 0, 1);
```

```
        glPushMatrix();
        glColor3d(1.0, 0.0, 0.0);
        glTranslated(0.5, 0, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glPopMatrix();

        glPushMatrix();
        glColor3d(0.0, 1.0, 0.0);
        glTranslated(0, 0.5, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glPopMatrix();

        glPushMatrix();
        glColor3d(0.0, 0.0, 1.0);
        glTranslated(-0.5, 0, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glPopMatrix();

        glutSwapBuffers(); // ?o?b?t?@?????L?
}

// ??莞?d??ЬRĂяo?????▫?
void timer(int value) {
        rotateAngle++; // ??]?p?x?X?V

        glutPostRedisplay(); // ?ě`?施??
        glutTimerFunc(100 , timer , 0); //
100?~???b??Ÿ??g?????s????
}

// ?f?B?X?v???C???X?g???쐬????
void buildDisplayList() {
        glNewList(ID_DRAW_STAR,GL_COMPILE);

        double r0 = 0.15; // ?????a
        double r1 = 0.4; // ???O?a
        glBegin(GL_TRIANGLES);
        for(int i = 0; i < 5; i++) { // 5?▫?O?p?`?ő???\???????
                int deg = i * 72;
                glVertex3d(r0 * cos( (deg - 36) * M_PI / 180.0), r0 * sin(
(deg - 36) * M_PI / 180.0), 0);  // ????????_
                glVertex3d(r1 * cos( deg * M_PI / 180.0), r1 * sin( deg *
M_PI / 180.0), 0);  // ?O?????_
                glVertex3d(r0 * cos( (deg + 36) * M_PI / 180.0), r0 * sin(
(deg + 36) * M_PI / 180.0) ,0);  // ????????_
        }
        glEnd();

        glEndList();
}


// ???C???v???O????
int main (int argc, char *argv[]) {
```

```
        glutInit(&argc, argv);          // ???C?u??????????
        glutInitDisplayMode(GLUT_RGBA|GLUT_DOUBLE);

        glutInitWindowSize(400 , 400);  // ?E?B???h?E?T?C?Y???w??
        glutCreateWindow(argv[0]);      // ?E?B???h?E???쐬
        glutDisplayFunc(display);       // ?\???□????w??

        glutTimerFunc(100 , timer , 0); //
100?~???b??Y??s????□??w??

        buildDisplayList();

        rotateAngle = 0;                // ?ß??????l?ې?

        glutMainLoop();                 // ?C?x???g?≠?
        return 0;
}
```

スクリーンショット



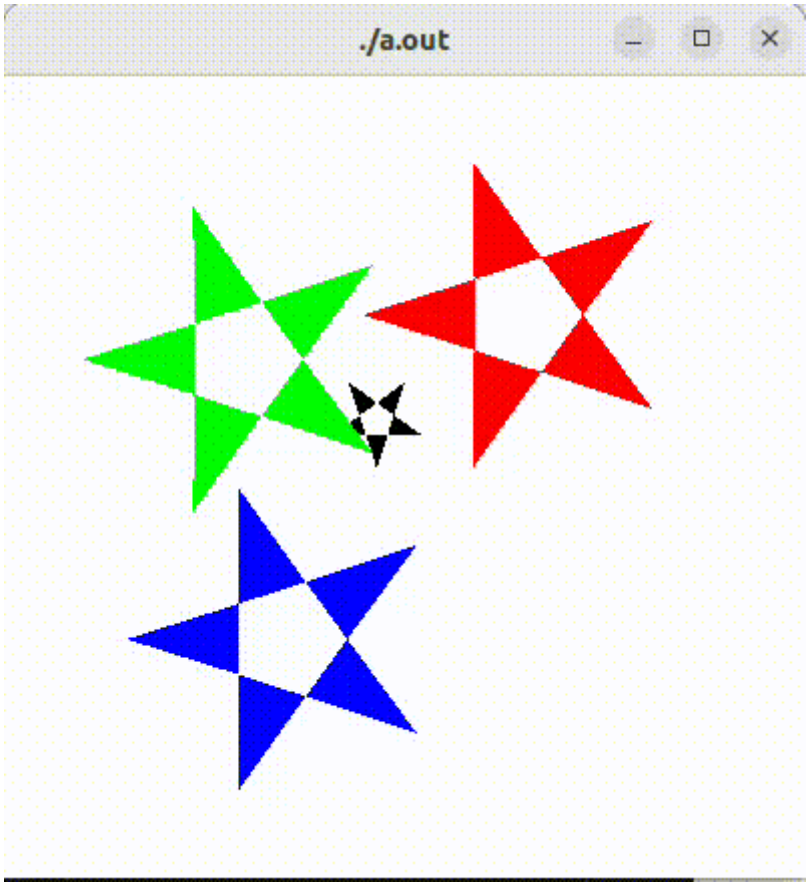
課題 3

ソースコード

```
#include <math.h>
#include <GL/glut.h> // ???C?u?????p?w?b?_?t?@?C???‡v???

// ?揩?▫?`
#ifndef M_PI
#define M_PI 3.14159265358979
#endif

// ?f?B?X?v???C???X?g?w?K
// ????`?愠??`?施?ヲ▫▫▫▫A?f?B?X?v???C???X?g?Б??č쐬???Ă???
// ?K?v?Ô??ʔA?????Ꮅ??Ăяo??

#define ID_DRAW_STAR 1 //  glNewList ?▫??Ŏg?p???鄒
??ID?B?l?????ł??\??Â?

int rotateAngle; // ??]?p?x???L?^???Ă????ß?

// ?\???????????▫??ŋL??
void display(void) {
        glClearColor (1.0, 1.0, 1.0, 1.0);  // ?????F?w??
        glClear(GL_COLOR_BUFFER_BIT);       //
??ȘÐ??s????????????

        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();

        glRotated(rotateAngle, 0, 0, 1);

        glPushMatrix();
        glColor3d(1.0, 0.0, 0.0);
        glTranslated(0.5, 0, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glRotated(rotateAngle * 2, 0, 0, 1);
        glPushMatrix();
        glColor3d(0.0, 0.0, 0.0);
        glScaled(0.3, 0.3, 1.0);
        glTranslated(1.5, 0, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glPopMatrix();
        glPopMatrix();

        glPushMatrix();
        glColor3d(0.0, 1.0, 0.0);
        glTranslated(0, 0.5, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glPopMatrix();

        glPushMatrix();
        glColor3d(0.0, 0.0, 1.0);
        glTranslated(-0.5, 0, 0);
        glRotated(rotateAngle, 0, 0, 1);
        glCallList(ID_DRAW_STAR);
        glPopMatrix();
```

```
        glutSwapBuffers(); // ◘o◘b◘t◘@◘◘◘◘L◘
}

// ◘◘莞◘d◘◘ЬRĂяo◘◘◘◘◘▫◘
void timer(int value) {
        rotateAngle++; // ◘◘]◘p◘x◘X◘V

        glutPostRedisplay(); // ◘ě`◘施◘◘
        glutTimerFunc(100 , timer , 0); //
100◘~◘◘◘b◘◘ϒ◘◘g◘◘◘◘◘s◘◘◘◘
}

// ◘f◘B◘X◘v◘◘◘C◘◘◘X◘g◘◘◘쐬◘◘◘◘
void buildDisplayList() {
        glNewList(ID_DRAW_STAR,GL_COMPILE);

        double r0 = 0.15; // ◘◘◘◘◘a
        double r1 = 0.4; // ◘◘◘O◘a
        glBegin(GL_TRIANGLES);
        for(int i = 0; i < 5; i++) { // 5◘▫Ď0◘p◘`◘Ő◘◘◘\◘◘◘◘◘◘◘
                int deg = i * 72;
                glVertex3d(r0 * cos( (deg - 36) * M_PI / 180.0), r0 * sin(
(deg - 36) * M_PI / 180.0), 0);  // ◘◘◘◘◘◘◘◘_
                glVertex3d(r1 * cos( deg * M_PI / 180.0), r1 * sin( deg *
M_PI / 180.0), 0);  // ◘O◘◘◘◘◘_
                glVertex3d(r0 * cos( (deg + 36) * M_PI / 180.0), r0 * sin(
(deg + 36) * M_PI / 180.0) ,0);  // ◘◘◘◘◘◘◘◘_
        }
        glEnd();

        glEndList();
}


// ◘◘◘C◘◘◘v◘◘◘O◘◘◘◘
int main (int argc, char *argv[]) {
        glutInit(&argc, argv);          // ◘◘◘C◘u◘◘◘◘◘◘◘◘◘◘◘
        glutInitDisplayMode(GLUT_RGBA|GLUT_DOUBLE);

        glutInitWindowSize(400 , 400);  // ◘E◘B◘◘◘h◘E◘T◘C◘Y◘◘◘w◘◘
        glutCreateWindow(argv[0]);      // ◘E◘B◘◘◘h◘E◘◘◘쐬
        glutDisplayFunc(display);       // ◘\◘◘◘▫◘◘◘◘w◘◘

        glutTimerFunc(100 , timer , 0); //
100◘~◘◘◘b◘◘ϒ◘◘s◘◘◘◘▫◘◘w◘◘

        buildDisplayList();

        rotateAngle = 0;                // ◘ß◘◘◘◘◘◘l◘ٮ◘

        glutMainLoop();                 // ◘C◘x◘◘◘g◘≠◘
        return 0;
}
```

スクリーンショット



課題 4

ソースコード

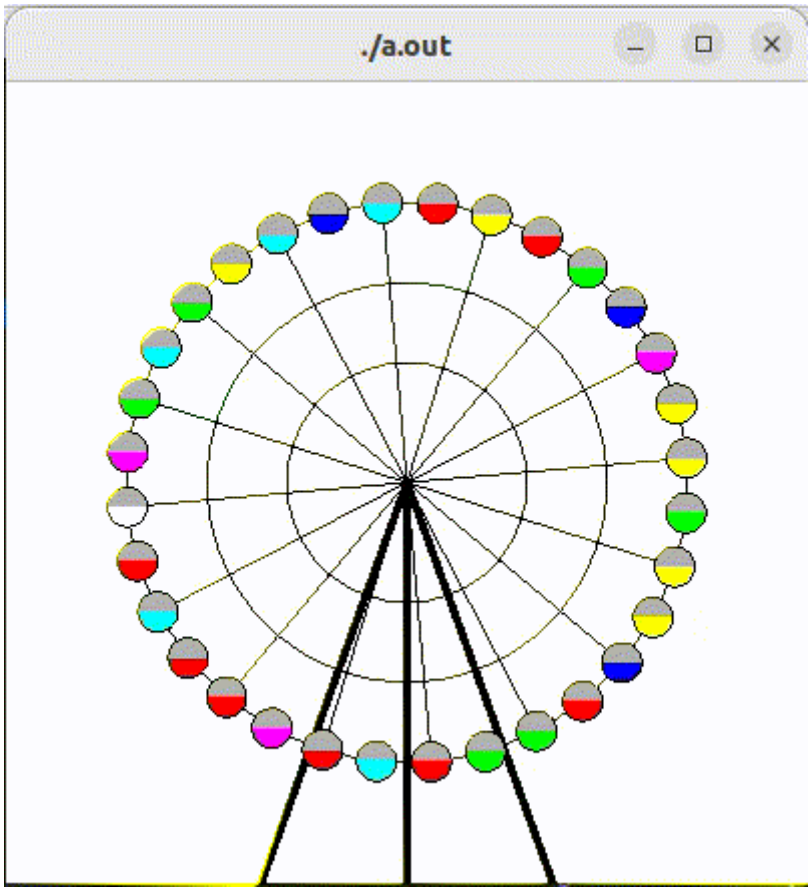
```
#include <math.h>
#include <GL/glut.h>
#include <random>

#ifndef M_PI
#define M_PI 3.14159265358979
#endif

#define ID_DRAW_CIRCLE 1
#define ID_DRAW_BRACKET 2
#define ID_DRAW_BAR 3

#define BRACKET_NUM 32

int rotateAngle;

GLdouble colors[][3] = {
        {1.0, 0, 0},
        {0, 1.0, 0},
```

```
        {0, 0, 1.0},
        {1.0, 1.0, 0},
        {1.0, 0, 1.0},
        {0, 1.0, 1.0},
        {1.0, 1.0, 1.0}
};
GLdouble* ranc;
GLdouble random_colors[BRACKET_NUM][3];

GLdouble* getRandomColor() {
        std::random_device rd;
        return colors[rd() % 7];
}

void display(void) {
        glClearColor (1.0, 1.0, 1.0, 1.0);
        glClear(GL_COLOR_BUFFER_BIT);

        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();

        glPushMatrix();
                glCallList(ID_DRAW_BAR);
                glRotated(-20, 0, 0, 1);
                glCallList(ID_DRAW_BAR);
                glRotated(40, 0, 0, 1);
                glCallList(ID_DRAW_BAR);
        glPopMatrix();

        glRotated(rotateAngle, 0, 0, 1);

        for(int i = 0; i < 3; i++) {
                glPushMatrix();
                        glColor3d(0.0, 0.0, 0.0);
                        glScaled(0.3 + (float)i * 0.2, 0.3 + (float)i * 0.2
,1);
                        glTranslated(0, 0, 0);
                        glCallList(ID_DRAW_CIRCLE);
                glPopMatrix();
        }

        glPushMatrix();
                for(int i = 0; i < BRACKET_NUM; i++) {
                        glPushMatrix();
                                glRotated(360.0 / (float)BRACKET_NUM *
(float)i, 0, 0, 1);
                                glTranslated(0, 0.7, 0);
                                glPushMatrix();
                                        if(i % 2 == 0) {
                                                glColor3d(0.0, 0.0, 0.0);
                                                glBegin(GL_LINE_STRIP);
                                                        glVertex2d(0, 0);
                                                        glVertex2d(0, -0.7);
                                                glEnd();
                                        }
                                        glScaled(0.05, 0.05 ,1);
                                        glRotated(-1.0 * 360.0 /
```

```
(float)BRACKET_NUM * (float)i - rotateAngle, 0, 0, 1);

                                        glColor3d(random_colors[i][0],
random_colors[i][1], random_colors[i][2]);
                                        glCallList(ID_DRAW_BRACKET);
                                glPopMatrix();
                        glPopMatrix();
                }
        glPopMatrix();

        glutSwapBuffers();
}

void timer(int value) {
        rotateAngle++;

        glutPostRedisplay();
        glutTimerFunc(100 , timer , 0);
}

void buildDisplayList() {
        glNewList(ID_DRAW_CIRCLE,GL_COMPILE);
                double r = 1.0;
                glBegin(GL_LINE_LOOP);
                        for(int i = 0; i < 36; i++) {
                                int deg = i * 10;
                                glVertex3d(r*cos(deg*M_PI/180.0),
r*sin(deg*M_PI/180.0), 0);
                        }
                glEnd();
        glEndList();

        glNewList(ID_DRAW_BRACKET,GL_COMPILE);
                r = 1.0;
                glBegin(GL_POLYGON);
                        for(int i = 0; i < 19; i++) {
                                int deg = i * -10;
                                glVertex3d(r*cos(deg*M_PI/180.0),
r*sin(deg*M_PI/180.0), 0);
                        }
                glEnd();
                glColor3d(0.7, 0.7, 0.7);
                glBegin(GL_POLYGON);
                        for(int i = 0; i < 19; i++) {
                                int deg = i * 10;
                                glVertex3d(r*cos(deg*M_PI/180.0),
r*sin(deg*M_PI/180.0), 0);
                        }
                glEnd();
                glColor3d(0.0, 0.0, 0.0);
                glCallList(ID_DRAW_CIRCLE);
        glEndList();

        glNewList(ID_DRAW_BAR, GL_COMPILE);
        glColor3d(0.0, 0.0, 0.0);
                glBegin(GL_POLYGON);
                        glVertex2d(-0.01, 0);
```

```
                        glVertex2d(0.01, 0);
                        glVertex2d(0.01, -2);
                        glVertex2d(-0.01, -2);
                glEnd();
        glEndList();
}


int main (int argc, char *argv[]) {
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_RGBA|GLUT_DOUBLE);

        glutInitWindowSize(400 , 400);
        glutCreateWindow(argv[0]);
        glutDisplayFunc(display);

        for(int i = 0; i < BRACKET_NUM; i++) {
                ranc = getRandomColor();
                for(int j = 0; j < 3; j++) random_colors[i][j] = ranc[j];
        }

        glutTimerFunc(100 , timer , 0);

        buildDisplayList();

        rotateAngle = 0;

        glutMainLoop();
        return 0;
}
```

スクリーンショット


./a.out