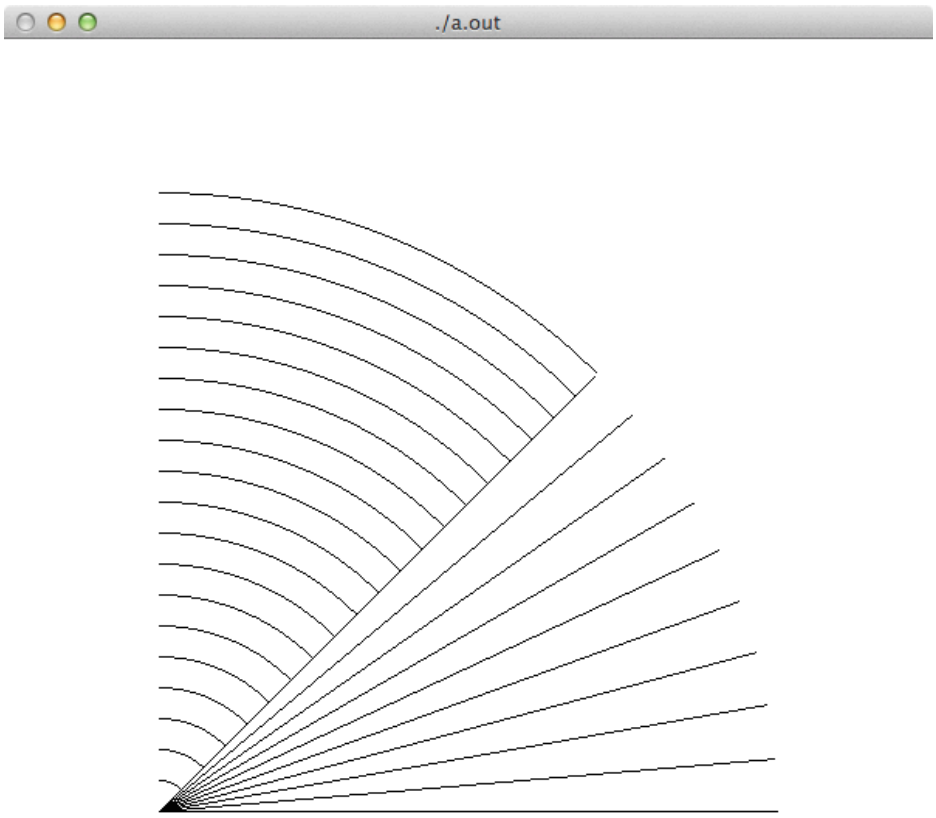
BresenhamMichener-1.c

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

int  width=600, height=600; // Initial window size

void  ploti(int x, int y) {
    glBegin(GL_POINTS); glVertex2i(x,y); glEnd();
}

void  line(int x1, int y1, int x2, int y2) {
    int dx, dy, d, e, x, y;
    dx = x2 - x1;  dy = y2 - y1;  d = 2 * dy;
    e = -dx;  x = x1;  y = y1;
    while (x < x2) {
        ploti(x, y);
        ++x;
        e += d;
        if (e > 0) {
            ++y;
            e -= 2 * dx;
        }
    }
}

void  octant(int r) {
    int x, y, d;
    x = 0;  y = r;  d = 3 - 2*r;
    while (x < y) {
        ploti(x, y);
        if (d < 0) {
            d += 4*x + 6;
        } else {
            d += 4*(x-y) + 10;
            --y;
        }
        ++x;
    }
}

void  display(void) {
    int i,x,y,x2;
    double a=1.99239, u=0., v=0.087156, w;
    double p=1., q=0.99619, r, m=400.;

    glClear(GL_COLOR_BUFFER_BIT);
    line( 0,0,(int)m,0 );
    line( 0,0,(int)(m*q),(int)(m*v) );
    for( i=5; i<45; i+=5 ){
        w = a*v - u;  r = a*q - p;
        x = (int)(m*r); y = (int)(m*w);  line( 0,0,x,y );
        u = v; v = w; p = q; q = r;
    }
    for( i=20; i<=400; i+=20 ) octant( i );
    glutSwapBuffers();
}
```

```
void myinit(void)  {
    glClearColor (1.0, 1.0, 1.0, 0.0); // background color
    glColor3f(0.,0.,0.);             // drawing color
}

void  reshape(int w, int h) {
    h = (h == 0) ? 1 : h;
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    width = w; height = h;
    glOrtho(-100.,width-101.,-100.,height-101.,-1.,1. );
    glMatrixMode(GL_MODELVIEW);
}

void keyboard(unsigned char key, int x, int y) {
    switch (key) {
        case 27:  /* ESC code */
        case 13: case 10: exit(0);  /* RETURN code */
        default:  break;
    }
}

int  main(int argc, char** argv) {
    glutInitWindowSize (width, height);
    glutInitWindowPosition (10, 10);
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);
    glutCreateWindow (argv[0]);
    myinit();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(keyboard);
    glutMainLoop();
    return 0;
}
```



BresenhamMichener-2.c

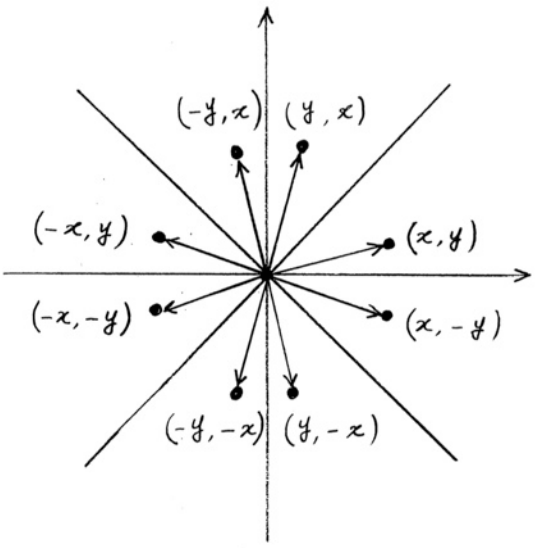
```
#include <GL/glut.h>
#include <math.h>
#include <stdlib.h>

int width=600, height=600;

void  ploti(int x, int y) {
    glBegin(GL_POINTS); glVertex2i(x,y); glEnd();
}

void  linei(int x1, int y1, int x2, int y2) {
    int absx2, absy2, d2, x, y, d, e, tmp, flag = 0;
    x2 = x2 - x1;
    y2 = y2 - y1;
    x2 < 0 ? (absx2 = -x2) : (absx2 = x2);
    y2 < 0 ? (absy2 = -y2) : (absy2 = y2);
    if (0 <= x2 && 0 < y2 && absx2 < absy2) {
        tmp = x2;  x2 = y2;  y2 = tmp;  flag = 1;
    } else if (0 > x2 && 0 < y2 && absx2 <= absy2) {
        tmp = x2;  x2 = y2;  y2 = -tmp;  flag = 2;
    } else if (0 > x2 && 0 <= y2 && absx2 > absy2) {
        x2 = -x2;  flag = 3;
    } else if (0 > x2 && 0 > y2 && absx2 >= absy2) {
        x2 = -x2;  y2 = -y2;  flag = 4;
    } else if (0 >= x2 && 0 > y2 && absx2 < absy2) {
        tmp = x2;  x2 = -y2;  y2 = -tmp;  flag = 5;
    } else if (0 < x2 && 0 > y2 && absx2 <= absy2) {
        tmp = x2;  x2 = -y2;  y2 = tmp;  flag = 6;
    } else if (0 < x2 && 0 > y2 && absx2 > absy2) {
        y2 = -y2;  flag = 7;
    }

    d2 = 2 * x2;
    x = 0;  y = 0;
    d = 2 * y2;  e = -x2;

    switch (flag) {
        case 0:
        while (x < x2) {
            ploti(x1 + x, y1 + y);  ++x;  e += d;
            if (e > 0) { ++y;  e -= d2; }
        }
        case 1:
        while (x < x2) {
            ploti(x1 + y, y1 + x);  ++x;  e += d;
            if (e > 0) { ++y;  e -= d2; }
        }
        case 2:
        while (x < x2) {
            ploti(x1 - y, y1 + x);  ++x;  e += d;
            if (e > 0) { ++y;  e -= d2; }
        }
        case 3:
        while (x < x2) {
            ploti(x1 - x, y1 + y);  ++x;  e += d;
            if (e > 0) { ++y;  e -= d2; }
        }
```

授業のスライドとはlinei()関数内の変数の使い方を変えたので説明する。
授業のスライドでは、x1とx2を入れ替えるなど、線分の方向が変化するので、ploti()の引数を考える時に混乱してしまった。

そこで、点(x1,y1)が原点に来るように点(x1,y1)と点(x2,y2)を平行移動する。

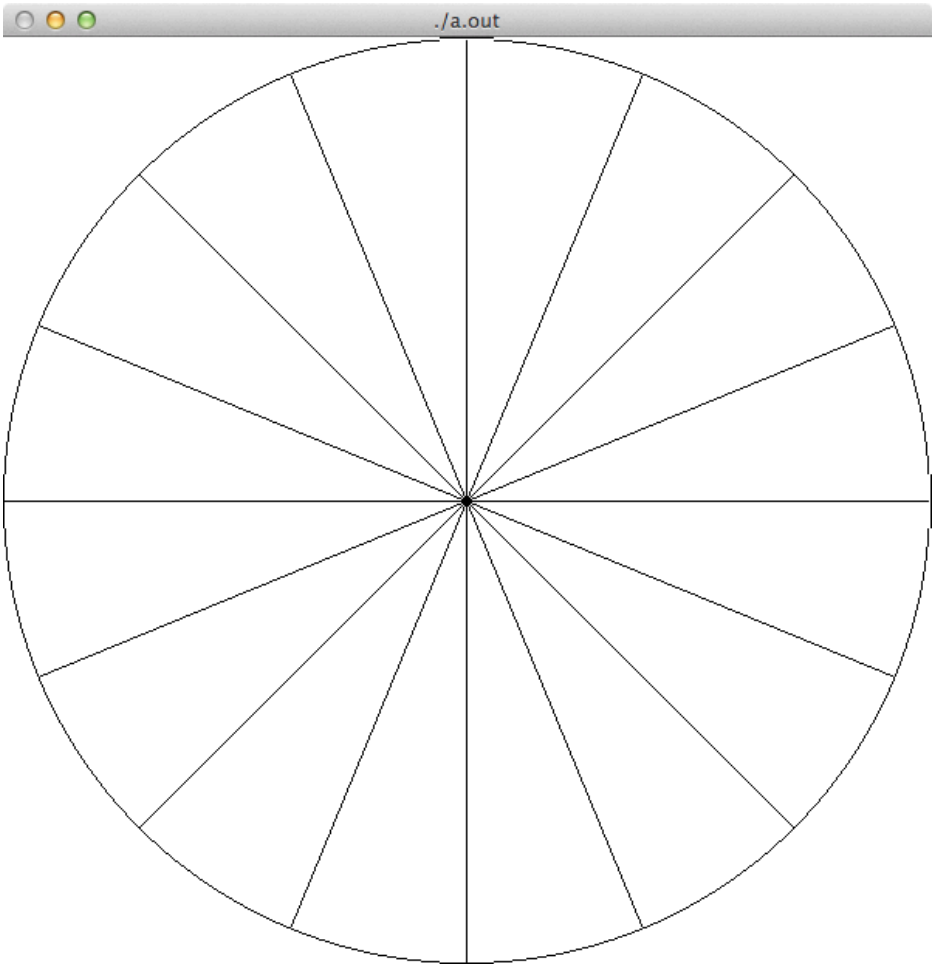
そして、次の図のように点(x2,y2)を変換する。
この図は、位相が(π/2)*n±θ(nは整数)ずれたときに三角関数をどのように変換するか考えるために高校時代から慣れ親しんでいるから混乱せずに済んだ。

また、変数 absx2, absy2は、傾きに関する条件式が複雑にならないように導入したものである。



```
            case 4:
            while (x < x2) {
                ploti(x1 - x, y1 - y);  ++x;  e += d;
                if (e > 0) { ++y;  e -= d2; }
            }
            case 5:
            while (x < x2) {
                ploti(x1 - y, y1 - x);  ++x;  e += d;
                if (e > 0) { ++y;  e -= d2; }
            }
            case 6:
            while (x < x2) {
                ploti(x1 + y, y1 - x);  ++x;  e += d;
                if (e > 0) { ++y;  e -= d2; }
            }
            case 7:
            while (x < x2) {
                ploti(x1 + x, y1 - y);  ++x;  e += d;
                if (e > 0) { ++y;  e -= d2; }
            }
        }
    }

    void  circlei(int x0, int y0, int r) {
        int x, y, d;
        x = 0;  y = r;  d = 3 - 2*r;
        while (x < y) {
            ploti(x0 + x, y0 + y);
            ploti(x0 + y, y0 + x);
            ploti(x0 - y, y0 + x);
            ploti(x0 + x, y0 - y);
            ploti(x0 - x, y0 - y);
            ploti(x0 - y, y0 - x);
            ploti(x0 + y, y0 - x);
            ploti(x0 - x, y0 + y);
            if (d < 0) {
                d += 4*x + 6;
            } else {
                d += 4*(x-y) + 10;
                --y;
            }
            ++x;
        }
    }

    void  display(void) {
        int i,ix,iy, ix1,iy1;
        double t;

        glClear(GL_COLOR_BUFFER_BIT);
        for( i=0; i<16; i++ ){
            t = (float)i*22.5*3.1415926545867932/180.;
            ix = (int)(300.*cos(t)) + width/2;
            iy = (int)(300.*sin(t)) + height/2;
            linei( width/2, height/2, ix, iy );
        }
        circlei( width/2,width/2,width/2 );
        glutSwapBuffers();
    }
```

```
void  myinit(void) {
    glClearColor (1.0, 1.0, 1.0, 0.0);
    glColor3f(0.,0.,0.);
}

void  reshape(int w, int h) {
    h = (h == 0) ? 1 : h;
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    width = w-1; height = h-1;
    glOrtho( 0.,width, 0.,height, -1., 1. );
    glMatrixMode(GL_MODELVIEW);
}

void keyboard(unsigned char key, int x, int y) {
    switch (key) {
        case 27:  // ESC code
        case 13: case 10: exit(0);  // RETURN (Enter) code
        default:break;
    }
}

int  main(int argc, char** argv) {
    glutInitWindowSize (width, height);
    glutInitWindowPosition (10, 10);
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
    glutCreateWindow (argv[0]);
    myinit();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(keyboard);
    glutMainLoop();
    return 0;
}
```



Excercise BresenhamMichener-3.c

BresenhamMichener-2.cをコピーしてdisplay関数のみ書き換えたので、display関数以外は省略する。

```
void  display(void) {
    int i,ix,iy, ix1,iy1;
    double t;

    glClear(GL_COLOR_BUFFER_BIT);                    // 以下の3行のみ書き換えた。
    for( i=0; i<16; i++ ){                           // BresenhamMichener-2.c のコードを
        t = (float)i*22.5*3.1415926545867932/180.;   // コメントに記載する。
        ix = (int)((width/2)*cos(t)) + width/2;      // ix = (int)(300.*cos(t)) + width/2;
        iy = (int)((width/2)*sin(t)) + height/2;     // iy = (int)(300.*sin(t)) + height/2;
        linei( width/2, height/2, ix, iy );
    }
    circlei( width/2,height/2,width/2 );             // circlei( width/2,width/2,width/2 );
    glutSwapBuffers();
}
```