第 7 回 宿題

「第 7 回」の宿題の雛形プログラムの cgsample07.c の最後の部分にある，実際に図形描画を行う関数 draw() は次のようになっています.

```
#define NVERTEX (sizeof vertex / sizeof vertex[0])    /* 頂点の数 */
#define NEDGE (sizeof edge / sizeof edge[0])          /* 稜線の数 */

/* 画面上の座標 */
static double screen[NVERTEX][3];

/*
** 図形の描画
*/
void draw(int width, int height)
{
  double rotation[4][4];                    /* 回転の変換行列       */
  double scale[4][4];                       /* 拡大縮小の変換行列   */
  double translation[4][4];                 /* 平行移動の変換行列   */
  double viewport[4][4];                    /* ビューポート変換行列 */
  double matrix[4][4];                      /* 合成した変換行列     */

  /*
  ** ビューポート変換行列を求める
  */

  /* scale にウィンドウサイズにあわせて図形を拡大縮小する変換行列を求める */
  setScale(scale, width * 0.08, height * 0.08, 1.0);
  /* translation に図形がウィンドウに収まるように平行移動する変換行列を求める */
  setTranslation(translation, width * 0.5, height * 0.5, 0.0);
  /* 拡大縮小の変換行列に平行移動の変換行列を掛けてビューポート変換行列を求める */
  multiply(viewport, translation, scale);

  /*
  ** 一つ目の図形のモデル変換行列を求める
  */

  /* rotation に図形を z 軸中心に 45 度回転する変換行列を求める */
  setRotationZ(rotation, 45.0);

  /*
  ** 一つ目の図形の座標変換と描画
  */

  /* 回転の変換行列にビューポート変換行列を掛ける */
  multiply(matrix, viewport, rotation);
  /* 画面上の点の位置を求める */
  projection(screen, matrix, vertex, NVERTEX);
  /* 求めた画面上の点の位置を使ってワイヤーフレームで図形を描く */
  wireframe(screen, edge, NEDGE);

  /*
  ** 二つ目の図形のモデル変換行列を求める
  */

  /* 図形を z 軸中心に -135 度回転する変換行列 */
  setRotationZ(rotation, -135.0);
```

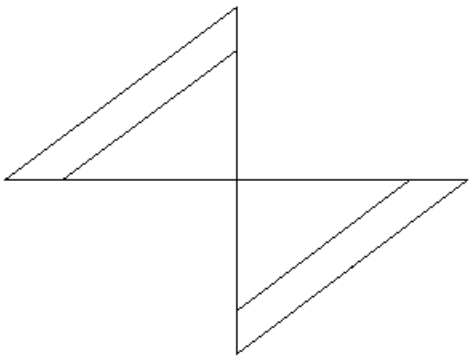
```
    /*
    ** 二つ目の図形の座標変換と描画
    */

    /* 回転の変換行列に拡大縮小・平行移動の積の変換行列を掛ける */
    multiply(matrix, viewport, rotation);
    /* 画面上の点の位置を求める */
    projection(screen, matrix, vertex, NVERTEX);
    /* 求めた画面上の点の位置を使ってワイヤーフレームで図形を描く */
    wireframe(screen, edge, NEDGE);
}
```

これは「第６回」の宿題で実装した線画による図形描画のプログラムにおいて，実際に図形を描画する部分を wireframe() という関数に分離し，さらに画面上の座標値を行列計算により求めようとするものです．関数 wireframe() や同次座標系のベクトルに変換行列を掛ける関数 transform()，４×４要素の行列同士の積を求める関数 multiply() は既に用意していますが，他のいくつかの関数は中身が実装されていません．これらの関数を実装して，「第７回」の宿題プログラムを完成させてください．

（１） ２点 (x0, y0), (x1, y1) を端点とする色 c の線分を描く関数 line()（「第３回」の宿題）
（２） (x, y, z) に平行移動する変換行列 m を求める関数 setTranslation()
（３） (a, b, c) 倍に拡大縮小する変換行列 m を求める関数 setScale()
（４） x 軸を中心に angle 度回転する変換行列を求める関数 setRotationX()
（５） y 軸を中心に angle 度回転する変換行列を求める関数 setRotationY()
（６） z 軸を中心に angle 度回転する変換行列を求める関数 setRotationZ()
（７） 配列 v の個々の要素（同次座標）に変換行列 m を乗じ，結果の x, y, z 成分をそれぞれ w 要素で割ったもの（実座標）を配列 s の対応する要素に代入する関数 projection()

関数 draw() では角度 angle を度で与えていますが，C 言語の三角関数 sin(), cos() の引数に与える値の単位はラジアンなので，angle をラジアンに換算する必要があります．また projection() はすでに用意してある関数 transform() を使うと簡単です．数学関数を使うときには math.h を #include することを忘れないでください．

ここまで実装してプログラムを実行すれば，右の図が表示されると思います．これが表示できたら，配列変数 vertex と edge に「第６回」に自分で作成した図形のデータを設定して，その図形を 45 度回転したものと 135度回転したものが二つ表示されるようにしてください（８）（９）．このとき，配列変数 vertex には座標値を**同次座標で設定する**必要があります．また，図形が画面にうまく収まるように，ビューポート変換行列の設定を調整してください．



プログラムが期待通り動作したら，作成したソースファイル（cgsample07.c）を授業のホームページのアップローダからアップロードしてください．期限は 12 月 1 日（木）中です．