1 課題 1

1.1 課題 1-1

プログラミングのツール：BCC には MAKE ユーティリティの他，grep や touch などプログラム開発を行う際に役に立つコマンドラインツールが含まれている．それぞれの機能や利用法について学習せよ．

1.1.1 grep

テキストファイルから一致する文字列を検索するためのコマンド．オプションを用いることで行頭や行末を検索するなどの検索法の指定ができる．

1.1.2 touch

指定したファイルのタイムスタンプを更新する．オプションを使用することで特定の日時を指定することが可能．指定されたファイルが存在しない場合は新規に作成する．

1.2 課題 1-2

C 言語：変数の「有効範囲（スコープ）」，「記憶域期間（記憶寿命）」について学習し，一般的な（ローカル）変数とグローバル変数，スタティック変数の違いを整理せよ．

有効範囲とは変数が利用できる範囲のことであり，記憶域機関とは変数がメモリに存在している期間である．ローカル変数の有効範囲は宣言された関数内である．他の関数でも変数を利用した場合はグローバル変数として宣言すればよい．ローカル変数は関数が終了するとメモリ上から消去されてしまうが，スタティック変数を用いることでプログラム実行中変数が保存されるようになる．

1.3 課題 1-3

星型正多角形を描き，回転させるようなプログラムを作成せよ．

星型正多角形を描き回転させるプログラムの主要部をソースコード 1 に示す．

ソースコード 1　星型正多角形描画の主要部

```
1  #define N_VERTEX 7
2  double rotAng = 0.0;
3
4  void display(void) {
5      int i;
6      double theta, dt, x, y;
7      glClear(GL_COLOR_BUFFER_BIT
           );
8      glColor3d(1.0, 0.0, 0.0);
9      dt = 2.0 * M_PI / N_VERTEX;
10     theta = rotAng;
11     glBegin(GL_LINE_LOOP);
12     for (i = 0; i < N_VERTEX; i
           ++) {
13         x = cos(theta);
14         y = sin(theta);
15         glVertex2d(x, y);
16         theta += dt * 2;
17     }
18     glEnd();
19     glFlush();
20     rotAng += 2 * M_PI / 120;
21 }
22
23 static void timer(int dummy) {
24     glutTimerFunc(10, timer, 0);
25     glutPostRedisplay();
26 }
```

1.4 課題 1-4

完全グラフを描き，回転させるようなプログラムを作成せよ．

完全グラフを描き回転させるプログラムの主要部をソースコード 1 に示す．各頂点について，まだ線で結ばれていない頂点との線分を描画する処理を行うことで完全グラフを描画した．

ソースコード 2　完全グラフ描画の主要部

```
1  #define N_VERTEX 7
2  double rotAng = 0.0;
3
4  void display(void) {
5      int i;
6      double theta, dt, x, y,delta
           ;
7      glClear(GL_COLOR_BUFFER_BIT
           );
8      glColor3d(1.0, 0.0, 0.0);
9      dt = 2.0 * M_PI / N_VERTEX;
10     glBegin ( GL_LINE_STRIP );
11     for (i = 0; i < N_VERTEX -1;
           i++) {
12         theta = rotAng + dt*i;
13         for (delta = i+1; delta
               < N_VERTEX; delta++)
                {
14             x = cos( theta );
15             y = sin( theta );
16             glVertex2d (x, y);
17             x = cos(theta -
                   delta *dt );
18             y = sin(theta -
                   delta *dt );
19             glVertex2d (x, y);
20         }
21     }
22     glEnd();
23     glFlush();
24     rotAng += M_PI / 120;
25 }
```

1.5 課題 1-5

数学関数のグラフを描くプログラムを作成せよ．例えば θ を 0 から 2π まで変化させながら，次の関数をプロットするとどんな図形が描かれるだろうか．

カージオイドを描画するプログラムの主要部をソースコード 3 に示す．14 行から 18 行までが描画する関数に依存する部分であり，xy を媒介変数で表すことで任意の関数を描画できる．

ソースコード 3　完全グラフ描画の主要部

```
1  #include<math.h>
2
3  void display(void) {
4      double theta,x, y,c_theta;
5      glClear(GL_COLOR_BUFFER_BIT
           );
6      glColor3d(1.0, 0.0, 0.0);
7      glBegin ( GL_LINE_STRIP );
8      for (theta = 0.0; theta <=
           2*M_PI; theta+=0.05) {
9          c_theta=cos(theta);
10         x=c_theta*(1+c_theta
               )*0.2;
11         y=sin(theta)*(1+c_theta
               )*0.2;
12         glVertex2d(x,y);
13     }
14     glEnd();
15     glFlush();
16 }
```

2　課題 2

2.1 課題 2-1

正四面体 $ABCD$ について，δBCD の重心 G を原点 O と一致させ，GA を x 軸，GB を y 軸とする座標系を考える．正四面体の一辺の長さを w とするとき，各頂点の 3 次元座標を導出せよ．

正四面体の断面図を図 1 に示す．

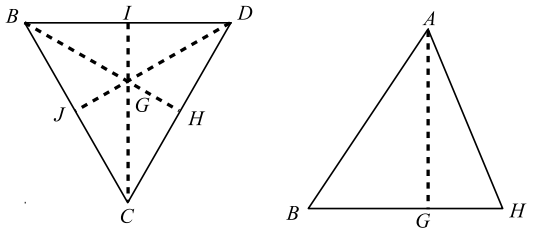

図 1　正四面体の断面図

まず $\triangle BCD$ を考える．G は BH を $2:1$ に内分する点であり，$BH = \frac{\sqrt{3}}{2}\ w$ であるから，

$$GH = \frac{1}{3} \times \frac{\sqrt{3}}{2} w = \frac{\sqrt{3}}{6}\ w$$
$$GB = \frac{2}{3} \times \frac{\sqrt{3}}{2} w = \frac{\sqrt{3}}{3}\ w$$

ここで三平方の定理より，

$$GA = \sqrt{AB^2 - GB^2} = \sqrt{w^2 - \frac{1}{3}w^2} = \frac{\sqrt{6}}{3} w$$

よって各頂点の三次元座標は，

$$\text{点 } A : \left(\frac{\sqrt{6}}{3} w, 0, 0 \right) \qquad \text{点 } B : \left(0, \frac{\sqrt{3}}{3} w, 0 \right)$$
$$\text{点 } C : \left(0, -\frac{\sqrt{3}}{6} w, \frac{1}{2} w \right) \quad \text{点 } D : \left(0, -\frac{\sqrt{3}}{6} w, -\frac{1}{2} w \right)$$

2.2 課題 2-2

前問で求めた結果をもとに，原点 O を中心とする半径 1 の球に内接する正四面体の頂点を導出せよ．

正四面体において，外接球の中心点 G' と各頂点の間の距離は外接球の半径 r に等しい．このとき三平方の定理より，

$$G'B^2 = GB^2 + G'G^2$$
$$r^2 = \frac{1}{3} w^2 + \left(\frac{\sqrt{6}}{3} w - r \right)^2$$
$$r = \frac{\sqrt{6}}{4} w$$
$$GG' = \frac{\sqrt{6}}{12}$$

ここで原点 O と G' の位置を合わせるために，x 軸方向に $-\sqrt{6}/12\ w$ シフトさせ，$r = 1$ とすると，

$$\text{点 } A : (1, 0, 0) \qquad \text{点 } B : \left(-\frac{1}{3}, \frac{2\sqrt{2}}{3}, 0 \right)$$
$$\text{点 } C : \left(-\frac{1}{3}, -\frac{\sqrt{2}}{3}, \frac{\sqrt{6}}{3} \right) \quad \text{点 } D : \left(-\frac{1}{3}, -\frac{\sqrt{2}}{3}, -\frac{\sqrt{6}}{3} \right)$$

となる．

2.3 課題 2-3

アームロボットについて，キー入力で台座の回転と，各関節の傾斜角を制御できるような　対話プログラムを作成してみよう．

アームロボットを回転制御するプログラムの主要部をソースコード 4 に示す．keyin 関数を用いて

台座の回転状態を格納する変数を操作することで，キーボードを用いてアームロボットを制御することができる．

ソースコード 4　アームロボット制御の主要部

```
1  //stat 0:停止  1;回転  -1:逆回転

2  GLfloat rotAng[3];
3  int b_stat = 0, l_stat = 0,
       u_stat = 0, pause = 0;
4
5  void display(void)
6  {
7
8      glClear(GL_COLOR_BUFFER_BIT
           );
9
10     glMatrixMode(GL_MODELVIEW);
11     glLoadIdentity();
12     gluLookAt(1, 1, 1.5, 0, 0,
           0, 0, 1, 0);
13     glRotated(rotAng[0], 0, 1,
           0);
14
15     glRotated(rotAng[0], 0, 1,
           0);
16     glCallList(ID_B);
17
18     //台座
19     if (pause == 0) {
20         if (b_stat == 1) rotAng
               [0] += 3.0;
21         if (b_stat == -1) rotAng
               [0] -= 3.0;
22     }
23
24     //下腕
25     glTranslated(0, HEIGHT_B,
           0);
26     glRotated(rotAng[1], 0, 0,
           1);
27     glCallList(ID_L);
28
29     if (pause == 0) {
30         if (l_stat == 1) rotAng
               [1] += 3.0;
31         if (l_stat == -1) rotAng
               [1] -= 3.0;
32     }
33
34     //上腕
35     glTranslated(0, HEIGHT_L,
           0);
36     glRotated(rotAng[2], 0, 0,
           1);
37     glCallList(ID_U);
38
39     if (pause == 0) {
40         if (u_stat == 1) rotAng
               [2] += 3.0;
41         if (u_stat == -1) rotAng
               [2] -= 3.0;
42     }
43
44     glutSwapBuffers();
45
46 }
47
48 void keyin(unsigned char key,
       int x, int y)
49 {
50     switch (key) {
51     case 'q':
52     case 'Q':exit(0); break;
53     case 'w':b_stat = 1;; break;
54     case 'e':b_stat = -1; break;
55     case 'r':b_stat = 0; break;
56     case 's':l_stat = 1; break;
57     case 'd':l_stat = -1; break;
58     case 'f':l_stat = 0; break;
59     case 'x':u_stat = 1; break;
60     case 'c':u_stat = -1; break;
61     case 'v':u_stat = 0; break;
62     case 'z': {dance = !dance;
           b_stat = 0;
63         l_stat = 0; u_stat = 0;
               break;
64     }
65     case 'p':pause = !pause;
           break;
66     }
67 }
```

2.4　課題 2-4

タイマコールバック関数を使って，アームロボットを制御するようなプログラムを作成してみよう．アームロボットが踊っているかのように見せるには，どんな工夫ができるだろうか．

アームロボットを踊らせるプログラムの主要部をソースコード**??**に示す．直線的，規則的な動きでは機械感が出てしまい踊っているように見えないと感じたため，乱数を用いて回転させる方法を考えた．ソースコード 4 のアームロボット制御プログラムの回転角制御時にこの関数を呼び出すことでランダムに回転するようになる．

3　課題 3

3.1　課題 3-1

空間中の任意の 3 点 $P_i = (x_i, y_i, x_i)$, $i = 1, 2, 3$ が与えられたとき，$\triangle P_1P_2P_3$ の単位法線ベクトルをすべて求める式を導出せよ．

ベクトルの外積を用いることで求められる．

外積を求めるため，3 点で作られるベクトルのう

ち 2 つを考える．

$$\overrightarrow{P_1P_2} = \vec{P_2} - \vec{P_1}$$
$$\overrightarrow{P_2P_3} = \vec{P_3} - \vec{P_2}$$

この 2 つのベクトルの外積を計算し，大きさを 1 とすればよいから，

$$\vec{n} = \pm\frac{\overrightarrow{P_1P_2} \times \overrightarrow{P_2P_3}}{|\overrightarrow{P_1P_2} \times \overrightarrow{P_2P_3}|}$$

となる．

3.2 課題 3-2

空間中の任意の 3 点 $P_i = (x_i, y_i, x_i)$, $i = 1, 2, 3$ が与えられたとき，$\triangle P_1P_2P_3$ のどちらが表（CCW）で，どちらが裏（CW）と判定できるか，判定方法を導出せよ．

頂点の順番をあらかじめ定めておき，頂点を時計回りにたどる 2 つのベクトルの外積ベクトルが向く方向を表（裏）と定めればよい．カメラから見た時の表裏判別が必要な場合は，カメラからの視線ベクトルを考え，ポリゴンと視線ベクトルの内積を計算する．外積ベクトルが向く方向を表と定めた場合，カメラからポリゴンの表面が見えているなら，視線ベクトルとポリゴンの外積ベクトルのなす角は 90 度未満になるため内積は正の値をとる．

3.3 課題 3-3

phong の照光モデルについて，詳しく調べよ．

phong の照光モデルは，3DCG における，モデリングされた面 (surface) 上の点に影をつけるための照明と陰影モデルである．このモデルにおいて反射は，鏡面反射，拡散反射，環境反射の 3 つで定義される．この 3 つの反射モデルによって面上の点の陰影を描画する．

まず，光源ごとに鏡面反射成分 i_s と拡散反射成分 i_d，アンビエント照明 (室内外の環境照明)i_a を定義する．次に，光が当たる材質について，鏡面反射係数 k_s，拡散反射係数 k_d，環境反射係数 k_a，光沢度 α を定義する，そして，光源群に対して，光が当たる点からの位置ベクトル L，法線を N，光が完全に反射される方向を R，光が当たる点から視点に向かう位置ベクトルを V とする．これらの定義された情報から，表面上の光の強度 I_p は次の式で計算できる．

$$I_p = k_a i_a + \sum_{Lights} (k_d(L \cdot N)i_d + k_s(R \cdot V)^{\alpha} i_s)$$

3.4 課題 3-4

正四面体，正六面体，正八面体をスムーズシェーディングで表示するために，各頂点の単位法線ベクトルをどのように定めればよいか考え，実装せよ．

正四面体をスムーズシェーディングで表示するためのプログラムの主要部をソースコード 5 に示す．頂点が含まれる 3 つの面の法線ベクトルの平均値をその頂点の法線ベクトルとして実装した．

ソースコード 5　正四面体スムーズシェーディング表示プログラムの主要部

```
1  void display(void) {
2      int i, j;
3      glClear(GL_COLOR_BUFFER_BIT
           | GL_DEPTH_BUFFER_BIT);
4
5      glBegin(GL_TRIANGLES);
6      for (i = 0; i < 4; i++) {
7          for (j = 0; j < 3; j++)
               {
8              glNormal3dv(vert_n[
                   tP[i][j]]);
9              glVertex3dv(vP[tP[i
                   ][j]]);
10         }
11     }
12     glEnd();
13     glutSwapBuffers();
14 }
```

3.5 課題 3-5

アームロボットをシェーディング表示するものに変更してみよ．アームロボットをシェーディング表示するために，アームロボットプログラムに光源を追加するためのソースコードをソースコード 6 に示す．

ソースコード 6　アームロボットシェーディング表示プログラムの追加部

```
1  GLfloat lP1[] =
       {0.0,1.0,2.0,1.0};
2  GLfloat lC1[] =
       {1.0,1.0,1.0,1.0};
3  void init(void) {
4      glEnable(GL_LIGHTING);
5      glLightfv(GL_LIGHT1,
           GL_POSITION, lP1);
6      glLightfv(GL_LIGHT1,
           GL_DIFFUSE, lC1);
7      glLightfv(GL_LIGHT1,
           GL_SPECULAR, lC1);
8      glEnable(GL_LIGHT1);
9      glEnable(GL_CULL_FACE);
10     //...
11 }
```

4　感想

OpenGL の基本事項について理解でき，実装できたので画像処理への理解が深まったと思う．ただ，複雑な画像処理アプリケーションを実装する際には大変そうだなと感じた．

5　改善案

配布資料の PDF ファイルに誤植があった．1.1 準備の 4. において

glut32.dll:32bitOS では c:\Widnows\

となっているが，正しくは Windows だと思われる．