```
#include<windows.h>
#include<GL/glut.h>
#include<stdio.h>
#include<stdlib.h>
#include <math.h>
#include<string.h>
#define M_PI 3.14
#define DEG2RAD 3.14159/180.0
float hexagon_r=20;
float op=0;
float hexagon_dx,hexagon_dy,hexagon_gx,hexagon_gy;
#define RADPERDEG 0.0174533
double theta=0;
int frameNumber =0,i;
int f=0,a=0,b=0,c=0,d=0;

void *fonts[] = { GLUT_BITMAP_9_BY_15,
                                GLUT_BITMAP_TIMES_ROMAN_10,
                                GLUT_BITMAP_TIMES_ROMAN_24,
                                GLUT_BITMAP_HELVETICA_18,
            GLUT_BITMAP_HELVETICA_12,
            GLUT_BITMAP_HELVETICA_10 };

typedef struct Point {
  GLfloat x;
  GLfloat y;
} Point;



float yLocation = 0.0f;
```

```
float bloby=0,blobx=0,cx1=0,cy1=0,cx2=0,cy2=0;
int c0=0,c1=0,c2=0,c3=0;
void init();
void draw_hexagon(float,float);
void drawCircle(float, float , float , int);
void drawCancerCells(float,float,int,int);

void setBackgroundColor();
void frame0();
void frame1();
void frame2();
void frame3();
void frame4();
void output(int, int ,const char *);
void verticleLine(int ,int ,int );
void hexagonCancer(int ,int);
void drawCircle(GLfloat , GLfloat , GLfloat,int ,int ,int );
void drawCircleCancer(GLfloat , GLfloat , GLfloat,int ,int ,int );
void drawFilledCircle(GLfloat , GLfloat , GLfloat);
void drawHollowCircle(GLfloat , GLfloat , GLfloat,int ,int ,int );
void CellCancer(int,int);
void cell(int,int);
void cell(int ,int ,int );
void cancerCellM(int,int);
void cancerCellB(int,int) ;
void melignant();
void secondSet();
void firstSet();
void humanBody();
void blobAnimation();
```

```
void bloodVessel();

void normalCells();

void cellanimation1();
void DrawEllipse(float, float,int,int);
void   hexagon(int,int);
void horzontalLine(int,int,int);

void drawGlass();
void drawCig();
void drawCigs();
void drawSun();
void drawFan();
void Arrow(GLdouble,GLdouble,GLdouble,GLdouble,GLdouble ,GLdouble ,GLdouble );
 void cancerCell(int ,int);



  void drawBitmapText(const char *string, void *font,float x,float y){
    int len, i;
    glRasterPos2f(x, y);
    len = (int)strlen(string);
    for (i = 0; i<len; i++)
    {
      glutBitmapCharacter(font, string[i]);
    }

  }
```

```
void frame0(){
  setBackgroundColor();
  glColor3f(1, 0, 0);
  drawBitmapText( "CANCER AWARENESS", fonts[2],-20, 280);

  glColor3f(1, 0, 0);
  drawBitmapText("SUBMITTED BY", fonts[4],45, 180 );

  glColor3f(1, 0, 0.0);
  drawBitmapText( "Arun M", fonts[3],-50, 100);
  drawBitmapText( "Mamatha H S", fonts[3],-50, 10);

  glColor3f(1, 0, 0.0);
  drawBitmapText( "4MN17CS005", fonts[0],120, 100);
  drawBitmapText( "4MN17CS014", fonts[0],120, 10);

  glColor3f(1, 0, 0);
  drawBitmapText( " PRESS C TO CONTINUE", fonts[3],-10, -100);
  drawBitmapText( " PRESS Q TO CONTINUE", fonts[3],-10, -130);


}
void frame2(){
  glClear(GL_COLOR_BUFFER_BIT);
  setBackgroundColor();
  firstSet();
  secondSet();
  output(-150,400,"BENIGN TUMOR");
  verticleLine(-100,380,150);
```

```
  if(op<=1)
  {
    op+=0.08;
  }else{
    output(250,400,"MELIGNANT TUMOR");
    verticleLine(320,380,250);
  }
}
void frame3(){
  setBackgroundColor();
  humanBody();
  blobAnimation();
  bloodVessel();
  normalCells();
  cellanimation1();
}


void setBackgroundColor(){
  glBegin(GL_POLYGON);
  glColor3f(0,0,0);
  glVertex2f(-499,-499);
  glVertex2f(499,-499);

  glColor3f(0,0,0);
  glVertex2f(499,499);
  glVertex2f(-499,499);
  glEnd();
  glFlush();
}
```

```
void frame4(){
  glColor3f(0, 0, 0);
//  drawGlass(0,0);

  int x, y;
  x=350;
  y=250;
    glColor3f(0, 0, 0);
    //  drawGlass(0,0,0);
    double radius=85;
      glPushMatrix();
          glScalef(1,1.4,1);
    drawFilledCircle(450-x,480-y+25,radius);
    drawFilledCircle(250-x+10,420-y,radius);
    drawFilledCircle(650-x-10,420-y,radius);
    drawFilledCircle(100-x+40,290-y-10,radius);
    drawFilledCircle(800-x-40,290-y-10,radius);
glPopMatrix();
    //drawing glass
    glPushMatrix();
    glTranslated(100-x+40,275-y-10,0);
    glScalef(5,5.5,1);
    drawGlass();
    glPopMatrix();
    //drawing sun
    glPushMatrix();
    glTranslated(250-x+10,420-y+70,0);
        glScalef(1,1.5,1);
    glRotated(-frameNumber*.9,0,0,1);
```

```
drawSun();
glPopMatrix();


//drawing cigarette
glPushMatrix();
glTranslated(385-x,455-y+100,0);
glScaled(2,5,1);
drawCigs();
glPopMatrix();
//drawing fan
glPushMatrix();
glTranslated(800-x-40,290-y-10,0);
glRotated(-frameNumber*.9,0,0,1);
glScaled(90,100,0);
drawFan();
glPopMatrix();


//print asbest
glPushMatrix();
//glScaled(2,2,0);
glTranslated(600-x-10,410-y+70,0);
glColor3f(1, 0, 0);
drawBitmapText("ASBEST",fonts[2],1,1);
glPopMatrix();



//draw arrow
```

```
   glPushMatrix();
   glScalef(.5,.5,0);
   glTranslated(150-x,-(y+70),0);
   Arrow(400,120,0,90,290,0,8);
   glPopMatrix();



   glColor3f(1, 0, 0);
   drawBitmapText( "Drinking a lot of alcohol,too much sun exposure,smoking,", fonts[2],-160, -180);
   drawBitmapText( "carcinogens like asbestos or radioactive radiations", fonts[2],-130, -210);


}
  void bloodVessel()
  {
    //blood vessel cross section start



    glPushMatrix();


    glBegin(GL_POLYGON);


    glColor3f(1,0,0);


    glVertex2f(300,200); //1
    glVertex2f(310,20);


    glVertex2f(270,-150); //2
    glVertex2f(230,-300);
```

```
  glVertex2f(290,-300); //3
  glVertex2f(330,-150);

  glVertex2f(350,20); //4
  glVertex2f(360,200);

  glVertex2f(410,300); //5
  glVertex2f(460,330);

  glVertex2f(510,350); //6
  glVertex2f(510,430);

  glVertex2f(400,360); //7
  glVertex2f(360,320);

  glVertex2f(300,500); //8
  glVertex2f(180,500);
  glEnd();

 glPopMatrix();

 glPushMatrix();

 glBegin(GL_LINE_LOOP);

 glColor3f(0,0,0);

 glVertex2f(300,200); //1
//glVertex2f(310,20);
```

```
//glVertex2f(270,-150); //2
glVertex2f(230,-300);

glVertex2f(290,-300); //3
glVertex2f(330,-150);

glVertex2f(350,20); //4
glVertex2f(360,200);

glVertex2f(395,270); //5
//glVertex2f(460,330);

glVertex2f(510,350); //6
glVertex2f(510,430);

glVertex2f(400,360); //7
glVertex2f(360,320);

glVertex2f(300,500); //8
glVertex2f(180,500);
glEnd();
glPopMatrix();
glPushMatrix();
glColor3f(1,0,0);
output(30,-100,"BLOOD VESSEL");
horzontalLine(175,300,-90);
glColor3f(1,0,0);
output(30,450,"MELIGNANT");

glPopMatrix();
```

```
  }

  void cell(int x,int y,int r){
    hexagon(x,y);
    drawCircle(x,y,r,1,0,0);
    //drawCircle(x+10,y+5,10,1,0.5,0);
  }

  void init(){
    glEnable(GL_BLEND);
  glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    glClearColor(1,1,1,1);

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-300,500,-300,500);
  }
void keyboard(GLubyte key, GLint x,GLint y){
  switch(key){
    case 'c':f++;
    glutPostRedisplay();
    break;
    case 'q':exit(0);

     }

}
```

```
void initialize(){
  hexagon_dx=hexagon_r*cos(30.0*M_PI/180.0);
  hexagon_dy=hexagon_r*sin(30.0*M_PI/180.0);
  hexagon_gx=2.0*hexagon_dx;
  hexagon_gy=2.0*hexagon_dx*sin(60.0*M_PI/180.0);
  // printf("%d  %d  %d  %d\n",hexagon_dx,hexagon_dy,hexagon_gx,hexagon_gy );
}

void drawFilledCircle(GLfloat x, GLfloat y, GLfloat radius){
  int i;
  int triangleAmount = 20; //# of triangles used to draw circle

  //GLfloat radius = 0.8f; //radius
  GLfloat twicePi = 2.0f * M_PI;

  glBegin(GL_TRIANGLE_FAN);
  glVertex2f(x, y); // center of circle
  for(i = 0; i <= triangleAmount;i++) {
    glVertex2f(
      x + (radius * cos(i *  twicePi / triangleAmount)),
      y + (radius * sin(i * twicePi / triangleAmount))
    );
  }
  glEnd();
}
void draw_hexagon(float x,float y){
  glBegin(GL_LINE_LOOP);
  glVertex2f(x-hexagon_dx,y-hexagon_dy);
  glVertex2f(x-hexagon_dx,y+hexagon_dy);
  glVertex2f(x           ,y+hexagon_r );
```

```
  glVertex2f(x+hexagon_dx,y+hexagon_dy);
  glVertex2f(x+hexagon_dx,y-hexagon_dy);
  glVertex2f(x           ,y-hexagon_r );
  glEnd();
  //glFlush();
  drawCircle(x,y,4.0,20);
}

void drawCancerCell(float x,float y,float dx,float dy)
{
//  int r =( rand() % 20)/10.0f;

  glColor3f(0.831f,0.608f,0.627f);
  glBegin(GL_POLYGON);
  glVertex2f(x-hexagon_dx-2*dx,y-hexagon_dy+dy);
  glVertex2f(x-hexagon_dx+dx,y+hexagon_dy+dy);
  glVertex2f(x     +dx      ,y+hexagon_r );
  glVertex2f(x+hexagon_dx-dx,y+hexagon_dy-dy);
  glVertex2f(x+hexagon_dx-dx,y-hexagon_dy+dx);
  glVertex2f(x           ,y-hexagon_r );
  glEnd();

  glColor3f(0.612f,0.220f,0.290f);
  glBegin(GL_LINE_LOOP);
  glVertex2f(x-hexagon_dx-2*dx,y-hexagon_dy+dy);
  glVertex2f(x-hexagon_dx+dx,y+hexagon_dy+dy);
  glVertex2f(x     +dx      ,y+hexagon_r );
  glVertex2f(x+hexagon_dx-dx,y+hexagon_dy-dy);
  glVertex2f(x+hexagon_dx-dx,y-hexagon_dy+dx);
  glVertex2f(x           ,y-hexagon_r );
```

```
  glEnd();
  glColor3f(0.596f,0.537f,0.494f);
  drawFilledCircle(x,y,4);
  glFlush();
}
void drawCancerCells(float x,float y,int ni,int nj){
  int i,j; float x0,shiftP=2.0;
  float x1,y1;
  x-=((float)(ni-1))*hexagon_gx*0.5; // just shift x,y to start position (i=0,j=0)
  x-=((float)(nj-1))*hexagon_dx*0.5;
  y-=((float)(nj-1))*hexagon_gy*0.5;
  x1=x+15*hexagon_gx*0.5,y1=y+9*hexagon_gx*0.5;
  glColor3f(1.0,1.0,0.0);

  shiftP+=2;
  /* cancer cells are drawn here
  add amination here*/
  for (x0=x1,j=5; j<nj-2; j++){
    for (i=5; i<ni-2; i++){
      float dx[]= {2,3,1,2,3,4,5,6};
      float  dy[]={7,4,5,3,2,5,7,8,3,2,3};

      drawCancerCell(x1,y1,dx[j-5],dy[j-5]);
      x1+=hexagon_gx+shiftP-.3*i;
    }
    x0+=hexagon_dx+shiftP +.5*i;
    x1=x0+shiftP;
    y1+=hexagon_gy+shiftP;
  }
}/*
```

```
void Timer(int iUnused)
{
glutPostRedisplay();
drawCancerCells(-100,-100,10,10);
glutTimerFunc(3000, Timer, 300);
   Timer(10);
}*/

void humanBody(){
  glColor3f(1,0.87,0.77);
  glPushMatrix();
  glRotatef(-10,0,0,1);
  DrawEllipse(10,50,0,0); //left leg
  glPopMatrix();

  glPushMatrix();
  glRotatef(10,0,0,1);
  DrawEllipse(10,50,25,-7); //right leg
  glPopMatrix();
  DrawEllipse(20,50,12,70); //center torso

  glPushMatrix();

  glRotatef(60,0,0,1);
  DrawEllipse(40,10,60,55); // left arm
  glPopMatrix();

  glPushMatrix();
  glRotatef(-60,0,0,1);
  DrawEllipse(40,10,-50,75);// right arm
```

```
glPopMatrix();

DrawEllipse(15,25,12,135); //head

glPushMatrix();

glColor3f(1,0,0);
glBegin(GL_LINES);
glVertex2f(12,30); // center blood vessel
glVertex2f(12,120);

glVertex2f(12,118); // left line
glVertex2f(-8,90);

glVertex2f(-8,90); // left arm
glVertex2f(-30,60);

glVertex2f(12,118); // right vessel
glVertex2f(50,65);

glVertex2f(12,35); // right down  blood vessel
glVertex2f(25,10);


glVertex2f(25,10);  // right down calf blood vessel
glVertex2f(30,-20);

glVertex2f(12,35); // left down blood vessel
glVertex2f(2,10);
```

```
   // left down blood vessel
  glVertex2f(2,10);
  glVertex2f(-5,-30);

  glEnd();

}

void blobAnimation(){
  //glPopMatrix();
  //glPushMatrix();
  glColor3f(1,0,0);
  glTranslatef(blobx,bloby,0);
  DrawEllipse(5,5,10,35);
  glPopMatrix();

  //update();
  if(c0<78)
  {

    bloby+=5;
    c0+=5;

  }else if(c1<8)
    {
      bloby-=0.7*5;
      blobx-=0.5*5;
      c1++;

  }else{
```

```
    glPushMatrix();
    glColor3f(1,0,0);
    output(-270,70,"New TUMOUR");
    output(-270,40,"METASTASIS");
    horzontalLine(-110,-15,85);
    glPopMatrix();
  }
}

void horzontalLine(int Lx1,int Lx2,int Ly){
  glBegin(GL_LINES);
  glVertex2f(Lx1,Ly);
  glVertex2f(Lx2,Ly);
  glEnd();
}
void DrawEllipse(float radiusX, float radiusY,int posx,int posy)
{
   int i;

   glBegin(GL_POLYGON);

   for(i=0;i<360;i++)
   {
      float rad = i*DEG2RAD;
      glVertex2f(posx+cos(rad)*radiusX,
                 posy+sin(rad)*radiusY);
   }

   glEnd();
}
```

```
void drawHollowCircle(GLfloat x, GLfloat y, GLfloat radius,int r,int g,int b){
	int i;
	int lineAmount = 100; //# of triangles used to draw circle

	//GLfloat radius = 0.8f; //radius
	GLfloat twicePi = 2.0f * 3.14;
glColor3f(r,g,b);
	glBegin(GL_POLYGON);
		for(i = 0; i <= 300;i++) {
			glVertex2f(
			   x + (radius * cos(i *  twicePi / lineAmount)),
			   y + (radius* sin(i * twicePi / lineAmount))
			);
		}
	glEnd();
}
 void cancerCell(int x,int y){
   int r=15,R=1,G=1,B=0;

   drawHollowCircle(x,y,r,R,G,B);
   drawHollowCircle(x+15,y,r,R,G,B);
   drawHollowCircle(x+5,y-15,r,R,G,B);
   drawHollowCircle(x+13,y-3,r,R,G,B);
   drawHollowCircle(x,y+16,r,R,G,B); // nucleus
   drawHollowCircle(x+3,y,5,0,1,1);

 }
void cellanimation1(){
  int x1=0,y1=0,ctr1=0,ctr2=0;
```

```
glPushMatrix();
glTranslatef(cx1,cy1,0);
cancerCell(300,100);
cancerCell(335,120);
glPopMatrix();

if(c2<20)
{
  cy1+=0.5*10;
  cx1+=0.1*10;

  c2++;
}else if(c2<835){

    cy1+=0.5*15;
    cx1-=0.15*15;
    c2++;
}

glPushMatrix(); // second set of cells
glTranslatef(cx2,cy2,0);
cancerCell(330,35);
cancerCell(320,0);
glPopMatrix();


if(c2<20)
{
  cy1+=0.5*10;
```

```
  cx1+=0.1*10;

  c2++;
 }else if(c2<835){

    cy1+=0.5*15;
    cx1-=0.15*15;
    c2++;
}

    glPopMatrix();

 if(c2<20)
 {
  cy1+=0.5*10;
  cx1+=0.1*10;

  c2++;
 }else if(c2<835){

    cy1+=0.5*15;
    cx1-=0.15*15;
    c2++;
}

 if(c3<40){
  cy2+=0.5*10;
  cx2+=0.01*10;
  c3++;
 }else if(c3<50){
```

```
  cy2+=0.68*10;
  cx2+=0.35*10;
  c3++;
 }else if(c3<700){
  cy2+=0.5*15;
  cx2+=0.70*15;
  c3++;
 }

  if(c3<40){
  cy2+=0.5*10;
  cx2+=0.01*10;
  c3++;
 }else if(c3<50){
  cy2+=0.68*10;
  cx2+=0.35*10;
  c3++;
 }else if(c3<700){
  cy2+=0.5*15;
  cx2+=0.70*15;
  c3++;
 }

if(c3<40){
  cy2+=0.5*10;
  cx2+=0.01*10;
  c3++;
 }else if(c3<50){
  cy2+=0.68*10;
  cx2+=0.35*10;
```

```
  c3++;
 }else if(c3<700){
  cy2+=0.5*15;
  cx2+=0.70*15;
  c3++;
 }

}

void normalCells(){
 cell(80,30,5);
 cell(80,65,5);
 cell(135,65,5);
 cell(170,65,5);
 cell(195,65,5);
 cell(265,85,5);
 cell(280,120,5);

 cell(100,100,5);
 cell(130,130,5);
 cell(160,160,5);
 cell(190,190,5);
 cell(220,220,5);
 cell(260,240,5);

 cell(170,105,5);
 cell(215,145,5);
 cell(100,185,5);
 cell(110,215,5);
 cell(120,255,5);
```

```
 for(int i=0;i<7;i++)
 {
   cell(100+(i*35),0,5);
 }

}



void draw_hexagon_grid(float x,float y,int ni,int nj){
  int i,j; float x0,shiftP=2.0;
  x-=((float)(ni-1))*hexagon_gx*0.5; // just shift x,y to start position (i=0,j=0)
  x-=((float)(nj-1))*hexagon_dx*0.5;
  y-=((float)(nj-1))*hexagon_gy*0.5;
  //x1=x+15*hexagon_gx*0.5,y1=y+9*hexagon_gx*0.5;
  for (x0=x,j=0;j<nj;j++,x0+=hexagon_dx+shiftP,x=x0+shiftP,y+=hexagon_gy+shiftP)
  for (i=0;i<ni;i++,x+=hexagon_gx+shiftP){
    draw_hexagon(x,y);
  }

   /* x-=((float)(ni-1))*hexagon_gx*0.5; // just shift x,y to start position (i=0,j=0)
  x-=((float)(nj-1))*hexagon_dx*0.5;
  y-=((float)(nj-1))*hexagon_gy*0.5;*/

}

void display(){

  glClear(GL_COLOR_BUFFER_BIT);
  glColor3f(0.8, 0.0, 0.0);
```

```
//setBackgroundColor();
switch (f) {
  case 0:frame0();break;
  case 1:   frame1();break;
  break;
  case 2: if(c==0) {op=0;c++;}
         frame2();break;
  case 3:
        if(b==0)
        {
          yLocation = 0.0f;
          bloby=0,blobx=0,cx1=0,cy1=0,cx2=0,cy2=0;
          c0=0,c1=0,c2=0,c3=0;
          b=1;
        }
        frame3();
        break;
  case 4:
        if(b==0)
        {
          yLocation = 0.0f;
          bloby=0,blobx=0,cx1=0,cy1=0,cx2=0,cy2=0;
          c0=0,c1=0,c2=0,c3=0;
          b=1;
        }
        frame3();
        break;
  case 5:
        if(a==0){
          frameNumber=0;
```

```
          a=1;
         }
         frame4();break;
          //default:printf("boo"); /* value */:
  }
  //glFlush();
  glutSwapBuffers();
}



void frame1(){
  draw_hexagon_grid(70,70,15,13);
        if(d==1)
         drawCancerCells(70,70,12,19);
}
void drawCircle(float cx, float cy, float r, int num_segments){
  glBegin(GL_LINE_LOOP);
  for(int ii = 0; ii < num_segments; ii++)
  {
    float theta = 2.0f * 3.1415926f * (float)ii / (float)(num_segments);//get the current angle

    float x = r * cosf(theta);//calculate the x component
    float y = r * sinf(theta);//calculate the y component

    glVertex2f(x + cx, y + cy);//output vertex

  }
  glEnd();
}
```

```
void hexagon(int x,int y){
  float rad;
  float hexagon_r=20;
  float hexagon_dx=hexagon_r*cos(30.0*M_PI/180.0);
  float hexagon_dy=hexagon_r*sin(30.0*M_PI/180.0);
  float hexagon_gx=2.0*hexagon_dx;
  float hexagon_gy=2.0*hexagon_dx*sin(60.0*M_PI/180.0);

  glColor3f(0,0.8,0);
  glBegin(GL_POLYGON);
  glVertex2f(x-hexagon_dx,y-hexagon_dy);
  glVertex2f(x-hexagon_dx,y+hexagon_dy);
  glVertex2f(x           ,y+hexagon_r );
  glVertex2f(x+hexagon_dx,y+hexagon_dy);
  glVertex2f(x+hexagon_dx,y-hexagon_dy);
  glVertex2f(x           ,y-hexagon_r );
  glEnd();
}
void output(int x, int y,const char *string){
  //char string[100]="GLUT_BITMAP_TIMES_ROMAN_10";
  int len, i;
  glRasterPos2f(x, y);
  len = (int)strlen(string);
  for (i = 0; i<len; i++)
  {
    glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,string[i]);
  }

}
```

```
void verticleLine(int x,int y1,int y2){
  glBegin(GL_LINES);
  glVertex2f(x,y1);
  glVertex2f(x,y2);
  glEnd();
}

void hexagonCancer(int x,int y){
  float rad;
  float hexagon_r=15;
  float hexagon_dx=hexagon_r*cos(30.0*M_PI/180.0)-5;
  float hexagon_dy=hexagon_r*sin(30.0*M_PI/180.0)+2;
  float hexagon_gx=2.0*hexagon_dx;
  float hexagon_gy=2.0*hexagon_dx*sin(60.0*M_PI/180.0)+10;

  glColor3f(1,0.5,0);
  glBegin(GL_POLYGON);
  glVertex2f(x-hexagon_dx,y-hexagon_dy);
  glVertex2f(x-hexagon_dx,y+hexagon_dy);
  glVertex2f(x          ,y+hexagon_r );
  glVertex2f(x+hexagon_dx,y+hexagon_dy);
  glVertex2f(x+hexagon_dx,y-hexagon_dy);
  glVertex2f(x          ,y-hexagon_r );
  glEnd();
}


void drawCircle(GLfloat x, GLfloat y, GLfloat radius,int r,int g,int b){
  int i;
```

```
  int lineAmount = 100; //# of triangles used to draw circle

  //GLfloat radius = 0.8f; //radius
  GLfloat twicePi = 2.0f * 3.14;
  glColor3f(r,g,b);
  glBegin(GL_POLYGON);
  for(i = 0; i <= 300;i++) {
    glVertex2f(
      x + (radius * cos(i *  twicePi / lineAmount)),
      y + (radius* sin(i * twicePi / lineAmount))
    );
  }
  glEnd();
}

void drawCircleCancer(GLfloat x, GLfloat y, GLfloat radius,int r,int g,int b){
  int i;
  int lineAmount = 100; //# of triangles used to draw circle

  //GLfloat radius = 0.8f; //radius
  GLfloat twicePi = 2.0f * 3.14;
  glColor4f(r,g,b,op);
  glBegin(GL_POLYGON);
  for(i = 0; i <= 300;i++) {
    glVertex2f(
      x + (radius * cos(i *  twicePi / lineAmount)),
      y + (radius* sin(i * twicePi / lineAmount))
    );
  }
  glEnd();
```

```
}

void CellCancer(int x,int y){
  hexagonCancer(x,y);
  drawCircle(x,y,2,1,0,0);
}
void cell(int x,int y){
  hexagon(x,y);
  drawCircle(x,y,5,1,0,0);
  //drawCircle(x+10,y+5,10,1,0.5,0);
}
void cancerCellM(int x,int y){
  drawCircleCancer(20+x,10+y,13,1,1,0);
  drawCircleCancer(35+x,10+y,13,1,1,0);
  drawCircleCancer(30+x,20+y,10,1,1,0);
  drawCircleCancer(20+x,25+y,13,1,1,0);
  drawCircleCancer(15+x,10+y,10,1,1,0);
  drawCircleCancer(23+x,14+y,5,1,0,0);
}
void cancerCellB(int x,int y){
  drawCircle(20+x,10+y,13,1,1,0);
  drawCircle(35+x,10+y,13,1,1,0);
  drawCircle(30+x,20+y,10,1,1,0);
  drawCircle(20+x,25+y,13,1,1,0);
  drawCircle(15+x,10+y,10,1,1,0);
  drawCircle(23+x,14+y,5,1,0,0);
}
void melignant(){

  glColor3f(0,1,0);
```

```
glBegin(GL_LINES);
glVertex2f(-267,-20); // center blood vessel
glVertex2f(-267,331); //Benign vertical

glVertex2f(-130,330); //benign upper line
glVertex2f(-267,330);

glVertex2f(-267,-20);//benign lower line
glVertex2f(-129,-20);

glVertex2f(82,329);//melignant vertical line
glVertex2f(82,80);

glVertex2f(82,330);//melignant upper line
glVertex2f(221,330);

glVertex2f(82,80);//melignant lower line
glVertex2f(221,80);

glEnd();
cancerCellM(230,100);
cancerCellM(210,140);
cancerCellM(174,170);
cancerCellM(170,215);
cancerCellM(195,255);
cancerCellM(230,280);
cancerCellM(270,270);
cancerCellM(290,230);
cancerCellM(275,120);
cancerCellM(320,180);
```

```
  cancerCellM(290,150);




}
void secondSet(){
  for(i=0;i<5;i++)
  {
    hexagon(100+(i*30),100);
    drawCircle(100+(i*30),100,5,1,0,0);
  }

  for( i=0;i<4;i++)
  {
    hexagon(100+(i*30),140);
    drawCircle(100+(i*30),140,5,1,0,0);
  }

  for(int i=0;i<3;i++)
  {
    hexagon(100+(i*30),180);
    drawCircle(100+(i*30),180,5,1,0,0);
  }

  for(i=0;i<2;i++)
  {
    hexagon(100+(i*30),220);
    drawCircle(100+(i*30),220,5,1,0,0);
```

```
}

for( i=0;i<3;i++)
{
  hexagon(100+(i*30),260);
  drawCircle(100+(i*30),260,5,1,0,0);
}

for( i=0;i<5;i++)
{
  hexagon(100+(i*30),310);
  drawCircle(100+(i*30),310,5,1,0,0);
}

melignant();


glPushMatrix();

CellCancer(230,200);
CellCancer(250,200);
CellCancer(270,250);
glPopMatrix();

glPushMatrix();
CellCancer(280,220);
CellCancer(270,180);
CellCancer(290,250);
glPopMatrix();
```

```
  glPushMatrix();
  CellCancer(230,250);
  CellCancer(290,170);
  CellCancer(290,200);
  glPopMatrix();

  glPushMatrix();
  CellCancer(250,240);
  CellCancer(240,230);
  CellCancer(270,150);
  glPopMatrix();
  //cell(265,250);
  // hexagon(230,100);
  // hexagon(210,140);
  // hexagon(174,170);
  // hexagon(170,215);
  // hexagon(195,255);
  // hexagon(230,280);
  // hexagon(270,270);
  // hexagon(290,230);
  // hexagon(275,120);
  // hexagon(300,160);

}

void firstSet(){
  for( i=0;i<5;i++)
  {
    hexagon(-250+(i*30),310);
    drawCircle(-250+(i*30),310,5,1,0,0);
```

```
}

for( i=0;i<4;i++)
{
  hexagon(-250+(i*30),260);
  drawCircle(-250+(i*30),260,5,1,0,0);
}

for(i=0;i<3;i++)
{
  hexagon(-250+(i*30),220);
  drawCircle(-250+(i*30),220,5,1,0,0);
}

for( i=0;i<2;i++)
{
  hexagon(-250+(i*30),170);
  drawCircle(-250+(i*30),170,5,1,0,0);
}

for( i=0;i<2;i++)
{
  hexagon(-250+(i*30),130);
  drawCircle(-250+(i*30),130,5,1,0,0);
}

for(i=0;i<2;i++)
{
  hexagon(-250+(i*30),90);
  drawCircle(-250+(i*30),90,5,1,0,0);
```

```
}
for( i=0;i<4;i++)
{
  hexagon(-250+(i*30),40);
  drawCircle(-250+(i*30),40,5,1,0,0);
}

for( i=0;i<5;i++)
{
  hexagon(-250+(i*30),0);
  drawCircle(-250+(i*30),0,5,1,0,0);
}
cell(-90,10);
cell(-60,40);
cell(-45,80);
cell(-85,290);
cell(-65,250);
cell(-45,220);

cancerCellB(-110,130);
cancerCellB(-140,70);
cancerCellB(-190,100);
cancerCellB(-160,150);
cancerCellB(-100,160);
cancerCellB(-180,110);
cancerCellB(-200,160);
cancerCellB(-180,130);
cancerCellB(-150,110);
cancerCellB(-145,190);
cancerCellB(-170,170);
```

```
}

void drawHollowElipse(GLfloat x, GLfloat y, GLfloat radiusx,GLfloat radiusy){
  int i;
  int lineAmount = 100; //# of triangles used to draw circle

  //GLfloat radius = 0.8f; //radius
  GLfloat twicePi = 2.0f * M_PI;

  glBegin(GL_LINE_LOOP);
  for(i = 0; i <= lineAmount;i++) {
    glVertex2f(
      x + (radiusx * cos(i *  twicePi / lineAmount)),
      y + (radiusy* sin(i * twicePi / lineAmount))
    );
  }
  glEnd();
}
void drawFilledElipse(GLfloat x, GLfloat y, GLfloat radiusx,GLfloat radiusy){
  int i;
  int triangleAmount = 20; //# of triangles used to draw circle

  //GLfloat radius = 0.8f; //radius
  GLfloat twicePi = 2.0f * M_PI;

  glBegin(GL_TRIANGLE_FAN);
  glVertex2f(x, y); // center of circle
  for(i = 0; i <= triangleAmount;i++) {
    glVertex2f(
      x + (radiusx * cos(i *  twicePi / triangleAmount)),
```

```
      y + (radiusy * sin(i * twicePi / triangleAmount))
    );
  }
  glEnd();
}

void drawCig(){
  int xLL=20,yLL=40,xLR,yLR;
  /*Point p1={85,230},
  p2={115,230},
  p3={135,350},
  p4={65,350},*/
  double factor=8.0f/9.0f;
  Point p1={-5,-5},
  p2={5,-5},
  p3={5,15},
  p4={-5,15},

  p5={(p3.x+p2.x)/2.0f,(p3.y+p2.y)*factor},
  p6={(p1.x+p4.x)/2.0f,(p1.y+p4.y)*factor};
  double radiusy=1.1;
  //filled part
  glColor3f(.894f, .925f,.89f );
  glBegin(GL_POLYGON);
  glColor3f( .953, .933,.824);
  glVertex2f(p5.x,p5.y);
  glVertex2f(p6.x,p6.y);
  glVertex2f(p3.x,p3.y);
  glVertex2f(p4.x,p4.y);
  glEnd();
```

```
//  drawFilledElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, 4);
drawFilledElipse((p5.x+p6.x)/2.0f,p5.y,(p6.x-p5.x)/2.0f,radiusy);
glColor3f(0,0,0);
//  drawHollowElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,8);
glEnd();

// outlline of the quadrilateral
glLineWidth(2.0f);
glBegin(GL_LINE_LOOP);
glColor3f( 0,0,0);
glVertex2f(p1.x,p1.y);
glVertex2f(p2.x,p2.y);
glVertex2f(p3.x,p3.y);
glVertex2f(p4.x,p4.y);
glEnd();

//lower outline
drawHollowElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, radiusy);
glLineWidth(1.0f);

//filled polygon
glBegin(GL_POLYGON);
glColor3f( .953, .933,.824);
glVertex2f(p1.x,p1.y);
glVertex2f(p2.x,p2.y);
glVertex2f(p3.x,p3.y);
glVertex2f(p4.x,p4.y);
glEnd();
```

```
  //lower  ellipse
  drawFilledElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, radiusy);
  drawFilledElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,radiusy);
  glColor3f(0,0,0);
  drawHollowElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,radiusy);

  //filled area
  glBegin(GL_POLYGON);
  glColor3f(0.855f,0.651f,0.322f);
  glVertex2f(p4.x,p4.y);
  glVertex2f(p3.x,p3.y);
  glVertex2f(p5.x,p5.y);
  glVertex2f(p6.x,p6.y);
  glEnd();

  //upper filled elipse
  drawFilledElipse((p5.x+p6.x)/2.0f,p5.y,(p6.x-p5.x)/2.0f,radiusy);
  drawHollowElipse((p5.x+p6.x)/2.0f,p5.y,(p6.x-p5.x)/2.0f,radiusy);
  drawFilledElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,radiusy);
  glColor3f(0,0,0);
  drawHollowElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,radiusy);



}

void drawCigs(){

  //  glTranslated(-5,0,0);
  for (int i = 0; i < 4; i++) {
    glTranslated(i+12, 0, 0);
```

```
    drawCig();
  }
}
void drawFan() {
  int i,frameNumber =0;

  glRotated(frameNumber * (180.0/46), 0, 0, 1);
  glColor3f(.349f, .267f, .231f );
  for (i = 0; i < 3; i++) {
    glRotated(120, 0, 0, 1);  // Note: These rotations accumulate.
    glBegin(GL_POLYGON);
    glVertex2f(0,0);
    glVertex2f(0.5f, 0.4f);
    //glVertex2f(1.5f,0);

    glVertex2f(0.5f, -0.4f);
    /*(glVertex2f(0.3,0.2);
    //   glVertex2f(0.5f, 0.1f);
    glVertex2f(0.7f,0.2);
    glVertex2f(0.5f, -0.1f);*/
    glEnd();
    drawFilledElipse(.5,0,.15,.4);
  }
  double r=.1;
  glColor3f(     .725f, .933f, .871f);
  drawFilledCircle(0,0,(r+.05));
  glColor3f(.349f, .267f, .231f );
  drawFilledCircle(0,0,r);
}
```

```
void drawGlass(){
  int xLL=20,yLL=40,xLR,yLR;
  /*Point p1={85,230},
  p2={115,230},
  p3={135,350},
  p4={65,350},*/
  Point p1={-5,-5},
  p2={5,-5},
  p3={5,15},
  p4={-5,15},

  p5={(p3.x+p2.x)/2.0f,(p3.y+p2.y)/2.0f+1.6},
  p6={(p1.x+p4.x)/2.0f,(p1.y+p4.y)/2.0f+1.6};
  double radiusy=1.5;
  //filled part
  glColor3f(.894f, .925f,.89f );
  glBegin(GL_POLYGON);
  glColor3f( .953, .933,.824);
  glVertex2f(p1.x,p1.y);
  glVertex2f(p2.x,p2.y);
  glVertex2f(p5.x,p5.y);
  glVertex2f(p6.x,p6.y);
  glEnd();

  //  drawFilledElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, 4);
  drawFilledElipse((p5.x+p6.x)/2.0f,p5.y,(p6.x-p5.x)/2.0f,radiusy);
  glColor3f(0,0,0);
  //  drawHollowElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,8);
  glEnd();
```

```
glLineWidth(2.0f);
glBegin(GL_LINE_LOOP);
glColor3f( 0,0,0);
glVertex2f(p1.x,p1.y);
glVertex2f(p2.x,p2.y);
glVertex2f(p3.x,p3.y);
glVertex2f(p4.x,p4.y);
glEnd();

drawHollowElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, radiusy);
glLineWidth(1.0f);

glBegin(GL_POLYGON);
glColor3f( .953, .933,.824);
glVertex2f(p1.x,p1.y);
glVertex2f(p2.x,p2.y);
glVertex2f(p3.x,p3.y);
glVertex2f(p4.x,p4.y);
glEnd();

//drawFilledElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, 4);
drawFilledElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,radiusy);
glColor3f(0,0,0);
drawHollowElipse((p3.x+p4.x)/2,p3.y,(p4.x-p3.x)/2,radiusy);

//filled area
glBegin(GL_POLYGON);
glColor3f(0.757f ,0.471f,0.071f);
//glColor3f(.89f,.92f,.89f );
```

```
  glVertex2f(p1.x,p1.y);
  glVertex2f(p2.x,p2.y);
  glColor3f(.965f,0.733f,0.133f);
  glVertex2f(p5.x,p5.y);
  glVertex2f(p6.x,p6.y);
  glEnd();

  drawFilledElipse((p1.x+p2.x)/2, p1.y, (p2.x-p1.x)/2, radiusy);
  drawFilledElipse((p5.x+p6.x)/2.0f,p5.y,(p6.x-p5.x)/2.0f,radiusy);

  glColor3f(0.918f,0.675f,0.118f);
  drawHollowElipse((p5.x+p6.x)/2.0f,p5.y,(p6.x-p5.x)/2.0f,radiusy);



}

void drawDisk(double radius) {
  int d;
  glBegin(GL_POLYGON);
  for (d = 0; d < 32; d++) {
    double angle = 2*M_PI/32 * d;
    glVertex2d( radius*cos(angle), radius*sin(angle));
  }
  glEnd();
}

void drawSun() {
  int i;
  //glColor3f(1,1,0);
```

```
  // glColor3f( .953, .933,.824);
  glColor3f(.98,.850,.36);
  glLineWidth(3.0f);
  for (i = 0; i < 13; i++) { // Draw 13 rays, with different rotations.
    glRotatef( 360 / 13, 0, 0, 1 ); // Note that the rotations accumulate!
    glBegin(GL_LINES);
    glVertex2f(0, 0);
    glVertex2f(75, 0);
    glEnd();
  }
  glLineWidth(1.0f);
  drawDisk(40);
  glColor3f(0,0,0);
}

void Arrow(GLdouble x1,GLdouble y1,GLdouble z1,GLdouble x2,GLdouble y2,GLdouble
z2,GLdouble D){
  double x=x2-x1;
  double y=y2-y1;
  double z=z2-z1;
  double L=sqrt(x*x+y*y+z*z);
  glColor3f(.694f,.859f,.133f);          //69.4, 85.9, 13.3
  GLUquadricObj *quadObj;

  glPushMatrix ();

  glTranslated(x1,y1,z1);
  //rotation

  if(theta<=120){
    theta= (frameNumber * (180.0/46));
```

```
}
glRotated(-theta, 0, 0, 1);

if((x!=0)||(y!=0)) {
  glRotated(atan2(y,x)/RADPERDEG,0.,0.,1.);
  glRotated(atan2(sqrt(x*x+y*y),z)/RADPERDEG,0.,1.,0.);
} else if (z<0){
  glRotated(180,1.,0.,0.);
}

glTranslatef(0,0,L-4*D);

quadObj = gluNewQuadric ();
gluQuadricDrawStyle (quadObj, GLU_FILL);
gluQuadricNormals (quadObj, GLU_SMOOTH);
gluCylinder(quadObj, 2*D, 0.0, 4*D, 32, 1);
gluDeleteQuadric(quadObj);

quadObj = gluNewQuadric ();
gluQuadricDrawStyle (quadObj, GLU_FILL);
gluQuadricNormals (quadObj, GLU_SMOOTH);
gluDisk(quadObj, 0.0, 2*D, 32, 1);
gluDeleteQuadric(quadObj);

glTranslatef(0,0,-L+4*D);

quadObj = gluNewQuadric ();
gluQuadricDrawStyle (quadObj, GLU_FILL);
gluQuadricNormals (quadObj, GLU_SMOOTH);
gluCylinder(quadObj, D, D, L-4*D, 32, 1);
```

```
  gluDeleteQuadric(quadObj);

  quadObj = gluNewQuadric ();
  gluQuadricDrawStyle (quadObj, GLU_FILL);
  gluQuadricNormals (quadObj, GLU_SMOOTH);
  gluDisk(quadObj, 0.0, D, 32, 1);
  gluDeleteQuadric(quadObj);

  glPopMatrix ();

}

void drawAxes(GLdouble length){
  glPushMatrix();
  glTranslatef(-length,0,0);
  Arrow(0,0,0, 2*length,0,0, 0.2);
  glPopMatrix();

  glPushMatrix();
  glTranslatef(0,-length,0);
  Arrow(0,0,0, 0,2*length,0, 0.2);
  glPopMatrix();

  glPushMatrix();
  glTranslatef(0,0,-length);
  Arrow(0,0,0, 0,0,2*length, 0.2);
  glPopMatrix();
}

void doFrame(int v) {
```

```
  frameNumber++;
  glutPostRedisplay();
  glutTimerFunc(180,doFrame,0);
}




int main(int argc, char** argv)
{
  glutInit(&argc,argv);
  initialize();


  glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB);
  glutInitWindowSize(1000,1000);
  //  glutInitWindowPosition(0,0);
  glutCreateWindow("Polygon");
  //glutTimerFunc(10, Timer, 0);
  glutDisplayFunc(display);

  glutKeyboardFunc(keyboard);
  glClearColor(1.0,1.0,1.0,1.0);
  gluOrtho2D(-300,500.0,-300,500.0);

  glutTimerFunc(500,doFrame,0);
  glutMainLoop();

  return 1;
}
```