#include<stdio.h>

#include<iostream.h>

#include<dos.h>

#include<process.h>

#include<conio.h>

#include<graphics.h>

#include<math.h>

//void render(float,float,float, float,float,float,

float,float,float,float,float,float);

void initialize(void);

void firstpage(void);

void call\_first(void);

float intensity,alpha,thita,tempy,tempz,tempx;

char ch='4';

char ch1='1';

char ch2='1';

int pts1[5][3];

float tx,ty,tz,d=.5;

void assign(float,float,float,float,float,float,float,float,float);

void scan\_line(float,float,float,float,float,float,float,float,float);

void drawpyramid(float,float,float,float,float,float);

void call\_assign(void);

void display(void);

void tranform(void);

void draw(void);

void drawscale(void);

float pts[5][3]={-100,0,0, 0,0,45, 100,0,0, 0,0,-45, 0,130,0};

float pts2[5][3]={228,273,0, 305,295,0, 428,273,0, 350,250,0

,328,143,0};

float pt[5][3]={-100,0,0, 0,0,45,100,0,0,0,0-45,0,130,0};

void main()

{

int i;

float sx,sy,sz=1;

struct palettetype pal;

int gd,gm;

detectgraph(&gd,&gm);

initgraph(&gd,&gm,"c:\tc\bgi");

getpalette(&pal);

firstpage();

for(i=16;i>0;i--)

setrgbpalette(pal.colors[i],0,4\*i,0);

L1: display();

while(ch1!='4')

{

ch='2';

L2: call\_assign();

clearviewport();

gotoxy(1,2);

cout<<"1. Translation";

cout<<"2. Rotation";

cout<<"3. Scaling ";

cout<<"4. Back ";

ch1=getch();

if(ch1=='4')

{

clearviewport();

goto L1;

}

if(ch1=='1')

{

clearviewport();

while(ch1!='4')

{

gotoxy(2,2);

cout<<"a. X+"; cout<<" b. X-";

cout<<" c. Y+"; cout<<" d. Y- ";

cout<<" e. Z+"; cout<<" f. Z-";

cout<<" g. Back";

call\_assign();

ch1=getch();

clearviewport();

if(ch1=='g')

goto L2;

if(ch1=='a')

tx=5;

if(ch1=='b')

tx=-5;

if(ch1=='c')

ty=5;

if(ch1=='d')

ty=-5;

if(ch1=='e')

tz=10;

if(ch1=='f')

tz=-10;

for(i=0;i<5;i++)

{

pts[i][0]+=tx;

pts[i][1]+=ty;

pts[i][1]+=tz;

}

}

}

if(ch1=='3')

{

clearviewport();

cout<<"Enter sx:";

cin>>sx;

cout<<"Enter sy:";

cin>>sy;

for(i=0;i<5;i++)

{

pts2[i][0]=abs(pts2[i][0]\*sx+200\*(1-sx));

pts2[i][1]=abs(pts2[i][1]\*sy+200\*(1-sy));

}

drawscale();

getch();

}

if(ch1=='2')

{

while(ch2!='4')

{

clearviewport();

gotoxy(1,2);

cout<<"1.X-axis rotation";

gotoxy(1,3);

cout<<"2.Y-axis rotation";

gotoxy(1,4);

cout<<"3.Z-axis rotation";

gotoxy(1,5);

cout<<"4.Back";

ch2=getch();

if(ch2=='4')

break;

if(ch2=='1')

{

alpha=0;

while(alpha<360)

{

alpha=alpha+10;

thita=(alpha\*3.142)/180;

initialize();

for(i=0;i<5;i++)

{

tempy=(pts1[i][1]\*cos(thita)+pts1[i][2]\*sin(thita));

pts1[i][2]=(pts1[i][1]\*sin(thita)-pts1[i][2]\*cos(thita));

pts1[i][1]=tempy;

}

clearviewport();

draw();

delay(100);

}

}

if(ch2=='2')

{

alpha=0;

while(alpha<360)

{

alpha=alpha+10;

thita=(alpha\*3.142)/180;

initialize();

for(i=0;i<5;i++)

{

tempz=(pts1[i][2]\*cos(thita)+pts1[i][0]\*sin(thita));

pts1[i][0]=(pts1[i][2]\*sin(thita)-pts1[i][0]\*cos(thita));

pts1[i][2]=tempz;

}

clearviewport();

draw();

delay(100);

}

}

if(ch2=='3')

{

alpha=0;

while(alpha<360)

{

alpha=alpha+10;

thita=(alpha\*3.142)/180;

initialize();

for(i=0;i<5;i++)

{

tempx=(pts1[i][0]\*cos(thita)-pts1[i][1]\*sin(thita));

pts1[i][1]=(pts1[i][0]\*sin(thita)+pts1[i][1]\*cos(thita));

pts1[i][0]=tempx;

}

clearviewport();

draw();

delay(100);

clearviewport();

draw();

}

}

}

}

}

closegraph();

restorecrtmode();

}

void initialize()

{

pts1[0][0]=-100;

pts1[0][1]=-65;

pts1[0][2]=0;

pts1[1][0]=0;

pts1[1][1]=-65;

pts1[1][2]=-45;

pts1[2][0]=100;

pts1[2][1]=-65;

pts1[2][2]=0;

pts1[3][0]=0;

pts1[3][1]=-65;

pts1[3][2]=45;

pts1[4][0]=0;

pts1[4][1]=65;

pts1[4][2]=0;

}

void firstpage()

{

clearviewport();

setcolor(WHITE);

settextstyle(2,HORIZ\_DIR,5);

outtextxy(250,15,"A Project on");

setcolor(GREEN);

settextstyle(3,HORIZ\_DIR,4);

outtextxy(170,25,"PYRAMID MODELING");

rectangle(300,120,580,320);

rectangle(295,115,585,325);

setcolor(6);

settextstyle(4,HORIZ\_DIR,3);

outtextxy(50,100, "OPTIONS");

settextstyle(3,HORIZ\_DIR,1);

setcolor(11);

outtextxy(20,150,"1. VISIBLE SURFACE DETECTION");

outtextxy(20,190,"2. SURFACE RENDERING");

outtextxy(20,230,"3. TRANSFORMATIONS");

outtextxy(20,270,"4. WIREFRAME DISPLAY");

outtextxy(20,310,"5. EXIT");

settextstyle(2,HORIZ\_DIR,4);

outtextxy(400,370,"Group Memebers");

setcolor(YELLOW);

outtextxy(410,385,"Made By Niket Shah");

call\_first();

//display();

setcolor(WHITE);

getch();

cleardevice();

clearviewport();

}

void display(void)

{ while(ch!='3')

{ clearviewport();

gotoxy(2,2);

cout<<"1. Visible Surface Detection ";

gotoxy(2,3);

cout<<"2. Surface Rendering";

gotoxy(2,4);

cout<<"3. Transformations";

gotoxy(2,5);

cout<<"4. Wireframe Display";

gotoxy(2,6);

cout<<"5. Exit ";

call\_assign();

ch=getch();

if(ch=='5')

exit(0);

clearviewport();

if(ch=='3')

break;

}

}

void call\_assign(void)

{

assign(pts[0][0],pts[0][1],pts[0][2],pts[1][0],pts[1][1],pts[1][2],pts[4][

0],pts[4][1],pts[4][2]);

assign(pts[1][0],pts[1][1],pts[1][2],pts[2][0],pts[2][1],pts[2][2],pts[4][

0],pts[4][1],pts[4][2]);

assign(pts[2][0],pts[2][1],pts[2][2],pts[3][0],pts[3][1],pts[3][2],pts[4][

0],pts[4][1],pts[4][2]);

assign(pts[0][0],pts[0][1],pts[0][2],pts[4][0],pts[4][1],pts[4][2],pts[3][

0],pts[3][1],pts[3][2]);

}

void call\_first(void)

{

assign(pt[0][0],pt[0][1],pt[0][2],pt[1][0],pt[1][1],pt[1][2],pt[4][0],pt[4

][1],pt[4][2]);

assign(pt[1][0],pt[1][1],pt[1][2],pt[2][0],pt[2][1],pt[2][2],pt[4][0],pt[4

][1],pt[4][2]);

assign(pt[2][0],pt[2][1],pt[2][2],pt[3][0],pt[3][1],pt[3][2],pt[4][0],pt[4

][1],pt[4][2]);

assign(pt[0][0],pt[0][1],pt[0][2],pt[4][0],pt[4][1],pt[4][2],pt[3][0],pt[3

][1],pt[3][2]);

}

void drawpyramid(float x1,float y1,float x2,float y2,float x3,float y3)

{

line(x1,y1,x2,y2);

line(x2,y2,x3,y3);

line(x3,y3,x1,y1);

}

void assign(float x1,float y1,float z1,float x2,float y2,float z2,float

x3,float y3,float z3)

{

float A,B,C;

float temp,An,Bn,Cn,X,Y,Z;

float Xl=-6,Yl=10,Zl=50;

float templ;

A=y1\*(z2-z3)+y2\*(z3-z1)+y3\*(z1-z2);

B=z1\*(x2-x3)+z2\*(x3-x1)+z3\*(x1-x2);

C=x1\*(y2-y3)+x2\*(y3-y1)+x3\*(y1-y2);

temp=sqrt(A\*A+B\*B+C\*C);

templ=sqrt(Xl\*Xl+Yl\*Yl+Zl\*Zl);

X=(float)Xl/templ; Y=(float)Yl/templ; Z=(float)Zl/templ;

An=(A/temp); Bn=(float)B/temp; Cn=(float)C/temp;

intensity=15\*(An\*X+Bn\*Y+Cn\*Z);

if (intensity<0)

intensity=0;

if (intensity>15)

intensity=15;

z1=55-z1;

x1=x1+300+(d\*z1); y1=300-y1-(d\*z1);

z2=55-z2;

x2=x2+300+(d\*z2); y2=300-y2-(d\*z2);

z3=55-z3;

x3=x3+300+(d\*z3); y3=300-y3-(d\*z3);

if(ch=='1')

{ if(intensity==0) return;

drawpyramid(x1,y1,x2,y2,x3,y3);

return;

}

if(ch=='3')

exit(0);

if(ch=='4')

drawpyramid(x1,y1,x2,y2,x3,y3);

if(ch=='2')

{

if(intensity==0) return;

if ((y1>y2) && (y1>y3) && (y2>y3))

scan\_line(x1,y1,z1,x2,y2,z2,x3,y3,z3);

if ((y1>y2) && (y1>y3) && (y3>y2))

scan\_line(x1,y1,z1,x3,y3,z3,x2,y2,z2);

if ((y2>y1) && (y2>y3) && (y1>y3))

scan\_line(x2,y2,z2,x1,y1,z1,x3,y3,z3);

if ((y2>y1) && (y2>y3) && (y3>y1))

scan\_line(x2,y2,z2,x3,y3,z3,x1,y1,z1);

if ((y3>y1) && (y3>y2) && (y1>y2))

scan\_line(x3,y3,z3,x1,y1,z1,x2,y2,z2);

if ((y3>y1) && (y3>y2) && (y2>y1))

scan\_line(x3,y3,z3,x2,y2,z2,x1,y1,z1);

}

}

void scan\_line(float x1,float y1,float z1,float x2,float y2,float

z2,float

x3,float y3,float z3)

{

int i;

float tempx,tempx1,tempy;

float m1,m2,thita,alpha;

alpha=0;

tempx=x1; tempx1=x1; tempy=y1;

m1=(y2-y1)/(x2-x1);

m2=(y3-y1)/(x3-x1);

while((int)tempy!=(int)y2)

{ alpha=alpha+5;

thita=(alpha\*3.14/180);

tempx=tempx-1/m1;

tempx1=tempx1-1/m2;

if(tempx<tempx1)

{

for(i=0;i+tempx<=tempx1;i++)

{

putpixel(tempx+i,tempy,intensity);

}

}

else

if (tempx1<tempx)

{ for(i=0;i+tempx1<=tempx;i++)

{

putpixel(tempx1+i,tempy,intensity);

}

}

tempy--;

}

m1=(float)(y3-y2)/(x3-x2);

while((int)tempy!=(int)y3)

{

tempx=tempx-1/m1;

tempx1=tempx1-1/m2;

if(tempx<tempx1)

{

for(i=0;i+tempx<=tempx1;i++)

putpixel(tempx+i,tempy,intensity);

}

else

{

for(i=0;i+tempx1<=tempx;i++)

putpixel(tempx1+i,tempy,intensity);

}

tempy--;

}

}

void draw()

{ int i;

for(i=0;i<5;i++)

{

pts1[i][2]=50+pts1[i][2]+50;

pts1[i][0]=pts1[i][0]+300+.5\*pts1[i][2];

pts1[i][1]=200+65-pts1[i][1]-.5\*pts1[i][2];

}

line(pts1[0][0],pts1[0][1],pts1[1][0],pts1[1][1]);

line(pts1[1][0],pts1[1][1],pts1[2][0],pts1[2][1]);

line(pts1[2][0],pts1[2][1],pts1[3][0],pts1[3][1]);

line(pts1[3][0],pts1[3][1],pts1[0][0],pts1[0][1]);

line(pts1[0][0],pts1[0][1],pts1[4][0],pts1[4][1]);

line(pts1[1][0],pts1[1][1],pts1[4][0],pts1[4][1]);

line(pts1[2][0],pts1[2][1],pts1[4][0],pts1[4][1]);

line(pts1[3][0],pts1[3][1],pts1[4][0],pts1[4][1]);

}

void drawscale()

{

line(pts2[0][0],pts2[0][1],pts2[1][0],pts2[1][1]);

line(pts2[1][0],pts2[1][1],pts2[2][0],pts2[2][1]);

line(pts2[2][0],pts2[2][1],pts2[3][0],pts2[3][1]);

line(pts2[3][0],pts2[3][1],pts2[0][0],pts2[0][1]);

line(pts2[0][0],pts2[0][1],pts2[4][0],pts2[4][1]);

line(pts2[1][0],pts2[1][1],pts2[4][0],pts2[4][1]);

line(pts2[2][0],pts2[2][1],pts2[4][0],pts2[4][1]);

line(pts2[3][0],pts2[3][1],pts2[4][0],pts2[4][1]);

}