Nama : Christian lumbantoruan

Nim : 20024080

Kelas : 5 TI 3

Laporan project 11

1. Kerucut

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

typedef struct {
	float m[4][4];
} matrix3D_t;

typedef struct {
	float v[4];
} vector3D_t;

typedef struct {
	float x;
	float y;
	float z;
} point3D_t;

typedef struct {
```

Edit dengan WPS Office

```
        float x;
        float y;
} point2D_t;

typedef struct {
        float r;
        float g;
        float b;
} color_t;

////////////////// matrices and vectors 3D ver 2 /////////////////
matrix3D_t createIdentity(void)
{
        matrix3D_t u;
        int i,j;
        for (i=0;i<4;i++) {
                for(j=0;j<4;j++) u.m[i][j]=0.;
                u.m[i][i]=1.;
        }
        return u;
}

matrix3D_t operator * (matrix3D_t a,matrix3D_t b)
{
```

Edit dengan WPS Office

```
    matrix3D_t c;//c=a*b
    int i,j,k;
    for (i=0;i<4;i++) for (j=0;j<4;j++) {
        c.m[i][j]=0;
        for (k=0;k<4;k++) c.m[i][j]+=a.m[i][k]*b.m[k][j];
    }
    return c;
}

vector3D_t operator * (matrix3D_t a, vector3D_t b)
{
    vector3D_t c;//c=a*b
    int i,j;
    for (i=0;i<4;i++) {
        c.v[i]=0;
        for (j=0;j<4;j++) c.v[i]+=a.m[i][j]*b.v[j];
    }
    return c;
}

matrix3D_t translationMTX(float dx,float dy,float dz)
{
    matrix3D_t trans=createIdentity();
    trans.m[0][3]=dx;
```

Edit dengan WPS Office

```
    trans.m[1][3]=dy;
    trans.m[2][3]=dz;
    return trans;
}

matrix3D_t rotationXMTX(float theta)
{
    matrix3D_t rotate=createIdentity();
    float cs=cos(theta);
    float sn=sin(theta);
    rotate.m[1][1]=cs; rotate.m[1][2]=-sn;
    rotate.m[2][1]=sn; rotate.m[2][2]=cs;
    return rotate;
}

matrix3D_t rotationYMTX(float theta)
{
    matrix3D_t rotate=createIdentity();
    float cs=cos(theta);
    float sn=sin(theta);
    rotate.m[0][0]=cs; rotate.m[0][2]=sn;
    rotate.m[2][0]=-sn; rotate.m[2][2]=cs;
    return rotate;
}
```

Edit dengan WPS Office

```
matrix3D_t rotationZMTX(float theta)
{
        matrix3D_t rotate=createIdentity();
        float cs=cos(theta);
        float sn=sin(theta);
        rotate.m[0][0]=cs; rotate.m[0][1]=-sn;
        rotate.m[1][0]=sn; rotate.m[1][1]=cs;
        return rotate;
}

matrix3D_t scalingMTX(float factorx,float factory,float factorz)
{
        matrix3D_t scale=createIdentity();
        scale.m[0][0]=factorx;
        scale.m[1][1]=factory;
        scale.m[2][2]=factorz;
        return scale;
}

matrix3D_t perspectiveMTX(float eyelength)
{
        matrix3D_t perspective=createIdentity();
        perspective.m[3][2]=-1./eyelength;
```

Edit dengan WPS Office

```
        return perspective;
}

point2D_t Vector2Point2D(vector3D_t vec)
{
        point2D_t pnt;
        pnt.x=vec.v[0];
        pnt.y=vec.v[1];
        return pnt;
}

point3D_t Vector2Point3D(vector3D_t vec)
{
        point3D_t pnt;
        pnt.x=vec.v[0];
        pnt.y=vec.v[1];
        pnt.z=vec.v[2];
        return pnt;
}

vector3D_t Point2Vector(point3D_t pnt)
{
        vector3D_t vec;
        vec.v[0]=pnt.x;
```

Edit dengan WPS Office

```
    vec.v[1]=pnt.y;
    vec.v[2]=pnt.z;
    vec.v[3]=1.;
    return vec;
}

vector3D_t homogenizeVector(vector3D_t vec)
{
    int i;
    for (i=0;i<3;i++) {
        vec.v[i]/=vec.v[3];
    }
    vec.v[3]=1.;
    return vec;
}

vector3D_t unitVector(vector3D_t vec)
{
    int i;
    float vec2=0.;
    float vec1,invvec1;
    for (i=0;i<3;i++) {
        vec2+=vec.v[i]*vec.v[i];
    }
```

Edit dengan WPS Office

```
    vec1=sqrt(vec2);
    if (vec1!=0.) {
        invvec1=1./vec1;
        for (i=0;i<3;i++) {
            vec.v[i]*=invvec1;
        }
    }
    vec.v[3]=1.;
    return vec;
}

vector3D_t operator ^ (vector3D_t a, vector3D_t b)
{
    vector3D_t c;//c=a*b
    c.v[0]=a.v[1]*b.v[2]-a.v[2]*b.v[1];
    c.v[1]=a.v[2]*b.v[0]-a.v[0]*b.v[2];
    c.v[2]=a.v[0]*b.v[1]-a.v[1]*b.v[0];
    c.v[3]=1.;
    return c;
}

vector3D_t operator - (vector3D_t v1,vector3D_t v0)
{
    vector3D_t c;//c=v1-v0
```

Edit dengan WPS Office

```
        c.v[0]=v1.v[0]-v0.v[0];
        c.v[1]=v1.v[1]-v0.v[1];
        c.v[2]=v1.v[2]-v0.v[2];
        c.v[3]=1.;
        return c;
}

vector3D_t operator - (vector3D_t v)
{
        vector3D_t c;//c=-v
        c.v[0]=-v.v[0];
        c.v[1]=-v.v[1];
        c.v[2]=-v.v[2];
        c.v[3]=1.;
        return c;
}

vector3D_t operator * (float r, vector3D_t b)
{
        vector3D_t c;//c=r*b
        int i;
        for (i=0;i<3;i++) {
                c.v[i]=r*b.v[i];
        }
```

Edit dengan WPS Office

```
    c.v[3]=1.;
    return c;
}

vector3D_t operator * (vector3D_t b, float r)
{
    vector3D_t c;//c=r*b
    int i;
    for (i=0;i<3;i++) {
        c.v[i]=r*b.v[i];
    }
    c.v[3]=1.;
    return c;
}

void setColor(float red,float green,float blue)
{
    glColor3f(red, green, blue);
}

void setColor(color_t col)
{
    glColor3f(col.r, col.g, col.b);
}
```

Edit dengan WPS Office

```
void drawDot(float x,float y)
{
        glBegin(GL_POINTS);
                glVertex2f(x, y);
        glEnd();
}

void drawDot(point2D_t p)
{
        glBegin(GL_POINTS);
                glVertex2f(p.x, p.y);
        glEnd();
}

void drawLine(float x1, float y1, float x2, float y2)
{
        glBegin(GL_LINES);
                glVertex2f(x1, y1);
                glVertex2f(x2, y2);
        glEnd();
}

void drawLine(point2D_t p1,point2D_t p2)
```

Edit dengan WPS Office

```
{
        drawLine(p1.x,p1.y,p2.x,p2.y);
}

//n: number of points
void drawPolyline(point2D_t pnt[],int n)
{
        int i;
        glBegin(GL_LINE_STRIP);
                for (i=0;i<n;i++) {
                        glVertex2f(pnt[i].x, pnt[i].y);
                }
        glEnd();
}

//n: number of vertices
void drawPolygon(point2D_t pnt[],int n)
{
        int i;
        glBegin(GL_LINE_LOOP);
                for (i=0;i<n;i++) {
                        glVertex2f(pnt[i].x, pnt[i].y);
                }
        glEnd();
```

Edit dengan WPS Office

```
}

// The function fillPolygon can fills only convex polygons
//n: number of vertices
void fillPolygon(point2D_t pnt[],int n,color_t color)
{
    int i;
    setColor(color);
    glBegin(GL_POLYGON);
        for (i=0;i<n;i++) {
            glVertex2f(pnt[i].x, pnt[i].y);
        }
    glEnd();
}

// The function gradatePolygon can fills only convex polygons
// The vertices will be painted with corresponding given colors.
// The points inside the polygon will be painted with the mixed color.
//n: number of vertices
void gradatePolygon(point2D_t pnt[],int num,color_t col[])
{
    int i;
    glBegin(GL_POLYGON);
        for (i=0;i<num;i++) {
```

Edit dengan WPS Office

```
            setColor(col[i]);
            glVertex2f(pnt[i].x, pnt[i].y);
        }
    glEnd();
}

//////////// End of OpenGL drawShape Functions ver 1 ////////////

void userdraw(void);

void display(void)
{
    glClear( GL_COLOR_BUFFER_BIT);
    userdraw();
    glutSwapBuffers();
}

/////////////////////////////////////////////////////////////////////
void drawcharX(float x,float y)
{
    drawLine(x,y,x+10,y+12);drawLine(x,y+12,x+10,y);
}

void drawcharY(float x,float y)
```

Edit dengan WPS Office

```
{

drawLine(x+5,y,x+5,y+7);drawLine(x,y+12,x+5,y+7);drawLine(x+10,y+12,x+5,y+7);
}


void drawcharZ(float x,float y)
{

drawLine(x,y+12,x+10,y+12);drawLine(x+10,y+12,x,y);drawLine(x,y,x+10,y);
}


void drawAxes(matrix3D_t view)
{
#define HALFAXIS  220
#define HALFAXIS1 (HALFAXIS-10)
        point3D_t axes[14]={
                {-HALFAXIS,0,0},{HALFAXIS,0,0},{HALFAXIS1,5,0},{HALFAXIS1,0,0},{0,0,0},
                {0,-HALFAXIS,0},{0,HALFAXIS,0},{0,HALFAXIS1,5},{0,HALFAXIS1,0},{0,0,0},
                {0,0,-HALFAXIS},{0,0,HALFAXIS},{5,0,HALFAXIS1},{0,0,HALFAXIS1}
        };
        vector3D_t vec[14];
```

Edit dengan WPS Office

```
        point2D_t buff[14];
        int i;
        for (i=0;i<14;i++) {
                vec[i]=Point2Vector(axes[i]);
                vec[i]=view*vec[i];
                buff[i]=Vector2Point2D(vec[i]);
        }
        drawPolyline(buff,14);
        drawcharX(buff[1].x,buff[1].y);
        drawcharY(buff[6].x,buff[6].y);
        drawcharZ(buff[11].x-14,buff[11].y);
}

/////////////////////////////////////////////////////////////////
typedef struct {
        int NumberofVertices; //in the face
        short int pnt[50];
        color_t col;
} face_t;
typedef struct {
        int NumberofVertices; //of the object
        point3D_t pnt[1600];
        color_t col[1600];
        int NumberofFaces; //of the object
```

Edit dengan WPS Office

```
    face_t fc[1000];
} object3D_t;

void draw3D(object3D_t obyek,matrix3D_t mat){
    vector3D_t vec[1600], vecbuff[50];
    vector3D_t vecNormal;
    point2D_t p[50];
    int i,j;
    for(i=0;i<obyek.NumberofVertices;i++){
        vec[i]=Point2Vector(obyek.pnt[i]);
        vec[i]=mat*vec[i];
    }
    setColor(1,0,0);
    for(i=0;i<obyek.NumberofFaces;i++){
        for(j=0;j<obyek.fc[i].NumberofVertices;j++)
            vecbuff[j]=vec[obyek.fc[i].pnt[j]];
        vecNormal=(vecbuff[1]-vecbuff[0])^(vecbuff[2]-vecbuff[0]);
        if(vecNormal.v[2]<0){
            for(j=0;j<obyek.fc[i].NumberofVertices;j++){
                p[j]=Vector2Point2D(vecbuff[j]);
            }
            drawPolygon(p,obyek.fc[i].NumberofVertices);
        }
    }
```

Edit dengan WPS Office

```
    setColor(1,1,1);
    for(i=0;i<obyek.NumberofFaces;i++){
        for(j=0;j<obyek.fc[i].NumberofVertices;j++)
            vecbuff[j]=vec[obyek.fc[i].pnt[j]];
        vecNormal=(vecbuff[1]-vecbuff[0])^(vecbuff[2]-vecbuff[0]);
        if(vecNormal.v[2]>=0){
            for(j=0;j<obyek.fc[i].NumberofVertices;j++){
                p[j]=Vector2Point2D(vecbuff[j]);
            }
            drawPolygon(p,obyek.fc[i].NumberofVertices);
        }
    }
}

//KODING KERUCUT
void createCone(object3D_t &kerucut, int n, float r, float h){
 float a=6.28/n;
 int i;
 kerucut.pnt[0].x=0;
 kerucut.pnt[0].y=h;
 kerucut.pnt[0].z=0;
 for(i=1;i<=n;i++){
 kerucut.pnt[i].x=r*cos(i*a);
 kerucut.pnt[i].y=0;
```

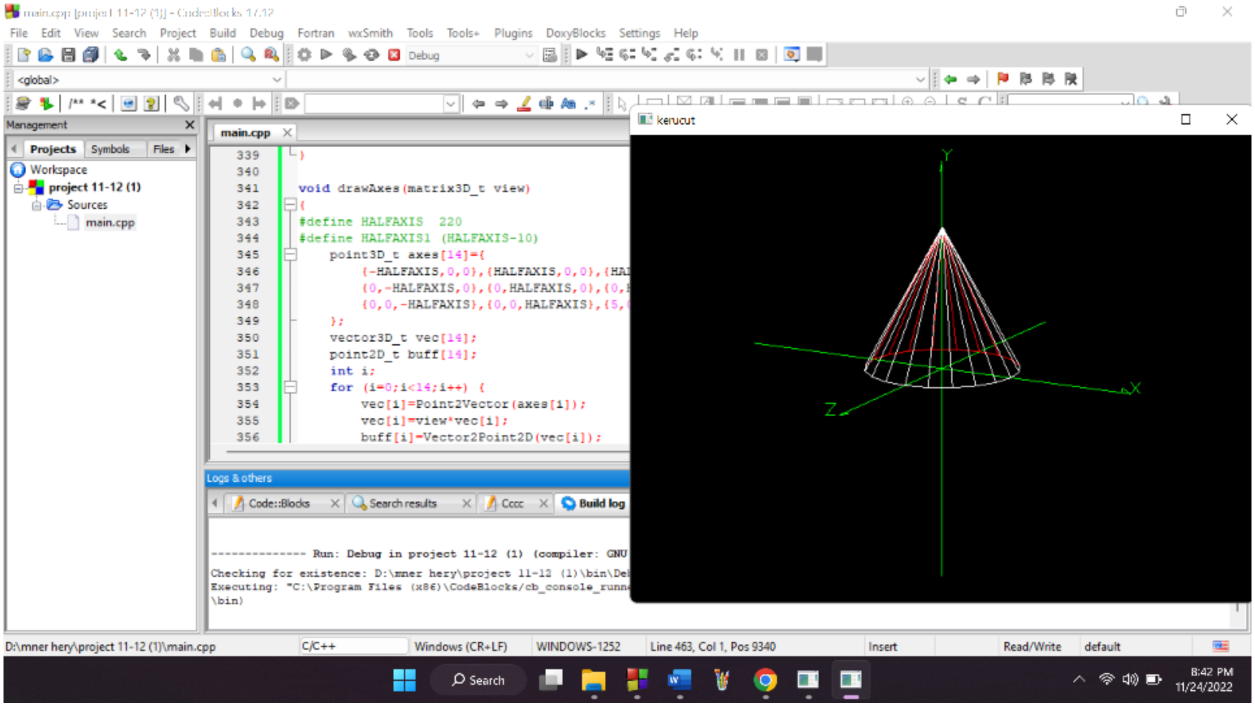
Edit dengan WPS Office

```
 kerucut.pnt[i].z=r*sin(i*a);
 }
 for(i=0;i<n;i++){
 kerucut.fc[i].NumberofVertices=3;
 kerucut.fc[i].pnt[0]=0;
 kerucut.fc[i].pnt[1]=i+2;
 kerucut.fc[i].pnt[2]=i+1;
 if(i==(n-1)) kerucut.fc[i].pnt[1]=1;
 }
 kerucut.fc[n].NumberofVertices=n;
 for(i=0;i<n;i++) kerucut.fc[n].pnt[i]=i+1;
 kerucut.NumberofVertices=n+1;
  kerucut.NumberofFaces=n+1;
}

void userdraw(void)
{
matrix3D_t tilting=rotationXMTX(0.25)*rotationYMTX(-0.5);
setColor(0,1,0);
drawAxes(tilting);
object3D_t kerucut;
createCone(kerucut,20,80,150);
setColor(1,1,1);
draw3D(kerucut,tilting);
```

Edit dengan WPS Office

```
}

int main(int argc, char **argv)
{
        glutInit(&argc,argv);
        glutInitDisplayMode ( GLUT_DOUBLE | GLUT_RGB );
        glutInitWindowPosition(100,100);
        glutInitWindowSize(640,480);
        glutCreateWindow ("kerucut");
        glClearColor(0.0, 0.0, 0.0, 0.0);
        gluOrtho2D(-320., 320., -240.0, 240.0);
        glutIdleFunc(display);
        glutDisplayFunc(display);
        glutMainLoop();
        return 0;
}
```

Edit dengan WPS Office



```
}

void drawAxes(matrix3D_t view)
{
#define HALFAXIS  220
#define HALFAXIS1 (HALFAXIS-10)
    point3D_t axes[14]={
        {-HALFAXIS,0,0},{HALFAXIS,0,0},{HA
        {0,-HALFAXIS,0},{0,HALFAXIS,0},{0,H
        {0,0,-HALFAXIS},{0,0,HALFAXIS},{5,(
    };
    vector3D_t vec[14];
    point2D_t buff[14];
    int i;
    for (i=0;i<14;i++) {
        vec[i]=Point2Vector(axes[i]);
        vec[i]=view*vec[i];
        buff[i]=Vector2Point2D(vec[i]);
```

```
-------------- Run: Debug in project 11-12 (1) (compiler: GNU
Checking for existence: D:\mner hery\project 11-12 (1)\bin\De
Executing: "C:\Program Files (x86)\CodeBlocks/cb_console_runn
\bin)
```

2.silinder bertumpuk

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

typedef struct {
	float m[4][4];
} matrix3D_t;

typedef struct {
	float v[4];
} vector3D_t;
```

Edit dengan WPS Office

```
typedef struct {
	float x;
	float y;
	float z;
} point3D_t;

typedef struct {
	float x;
	float y;
} point2D_t;

typedef struct {
	float r;
	float g;
	float b;
} color_t;

////////////////// matrices and vectors 3D ver 2 /////////////////
matrix3D_t createIdentity(void)
{
	matrix3D_t u;
	int i,j;
	for (i=0;i<4;i++) {
```

Edit dengan WPS Office

```
        for(j=0;j<4;j++) u.m[i][j]=0.;
        u.m[i][i]=1.;
    }
    return u;
}

matrix3D_t operator * (matrix3D_t a,matrix3D_t b)
{
    matrix3D_t c;//c=a*b
    int i,j,k;
    for (i=0;i<4;i++) for (j=0;j<4;j++) {
        c.m[i][j]=0;
        for (k=0;k<4;k++) c.m[i][j]+=a.m[i][k]*b.m[k][j];
    }
    return c;
}

vector3D_t operator * (matrix3D_t a, vector3D_t b)
{
    vector3D_t c;//c=a*b
    int i,j;
    for (i=0;i<4;i++) {
        c.v[i]=0;
        for (j=0;j<4;j++) c.v[i]+=a.m[i][j]*b.v[j];
```

Edit dengan WPS Office

```
	}
	return c;
}

matrix3D_t translationMTX(float dx,float dy,float dz)
{
	matrix3D_t trans=createIdentity();
	trans.m[0][3]=dx;
	trans.m[1][3]=dy;
	trans.m[2][3]=dz;
	return trans;
}

matrix3D_t rotationXMTX(float theta)
{
	matrix3D_t rotate=createIdentity();
	float cs=cos(theta);
	float sn=sin(theta);
	rotate.m[1][1]=cs; rotate.m[1][2]=-sn;
	rotate.m[2][1]=sn; rotate.m[2][2]=cs;
	return rotate;
}

matrix3D_t rotationYMTX(float theta)
```

Edit dengan WPS Office

```
{
	matrix3D_t rotate=createIdentity();
	float cs=cos(theta);
	float sn=sin(theta);
	rotate.m[0][0]=cs; rotate.m[0][2]=sn;
	rotate.m[2][0]=-sn; rotate.m[2][2]=cs;
	return rotate;
}

matrix3D_t rotationZMTX(float theta)
{
	matrix3D_t rotate=createIdentity();
	float cs=cos(theta);
	float sn=sin(theta);
	rotate.m[0][0]=cs; rotate.m[0][1]=-sn;
	rotate.m[1][0]=sn; rotate.m[1][1]=cs;
	return rotate;
}

matrix3D_t scalingMTX(float factorx,float factory,float factorz)
{
	matrix3D_t scale=createIdentity();
	scale.m[0][0]=factorx;
	scale.m[1][1]=factory;
```

Edit dengan WPS Office

```
        scale.m[2][2]=factorz;
        return scale;
}

matrix3D_t perspectiveMTX(float eyelength)
{
        matrix3D_t perspective=createIdentity();
        perspective.m[3][2]=-1./eyelength;
        return perspective;
}

point2D_t Vector2Point2D(vector3D_t vec)
{
        point2D_t pnt;
        pnt.x=vec.v[0];
        pnt.y=vec.v[1];
        return pnt;
}

point3D_t Vector2Point3D(vector3D_t vec)
{
        point3D_t pnt;
        pnt.x=vec.v[0];
        pnt.y=vec.v[1];
```

Edit dengan WPS Office

```
    pnt.z=vec.v[2];
    return pnt;
}

vector3D_t Point2Vector(point3D_t pnt)
{
    vector3D_t vec;
    vec.v[0]=pnt.x;
    vec.v[1]=pnt.y;
    vec.v[2]=pnt.z;
    vec.v[3]=1.;
    return vec;
}

vector3D_t homogenizeVector(vector3D_t vec)
{
    int i;
    for (i=0;i<3;i++) {
        vec.v[i]/=vec.v[3];
    }
    vec.v[3]=1.;
    return vec;
}
```

Edit dengan WPS Office

```
vector3D_t unitVector(vector3D_t vec)
{
	int i;
	float vec2=0.;
	float vec1,invvec1;
	for (i=0;i<3;i++) {
		vec2+=vec.v[i]*vec.v[i];
	}
	vec1=sqrt(vec2);
	if (vec1!=0.) {
		invvec1=1./vec1;
		for (i=0;i<3;i++) {
			vec.v[i]*=invvec1;
		}
	}
	vec.v[3]=1.;
	return vec;
}

vector3D_t operator ^ (vector3D_t a, vector3D_t b)
{
	vector3D_t c;//c=a*b
	c.v[0]=a.v[1]*b.v[2]-a.v[2]*b.v[1];
	c.v[1]=a.v[2]*b.v[0]-a.v[0]*b.v[2];
```

Edit dengan WPS Office

```
        c.v[2]=a.v[0]*b.v[1]-a.v[1]*b.v[0];
        c.v[3]=1.;
        return c;
}

vector3D_t operator - (vector3D_t v1,vector3D_t v0)
{
        vector3D_t c;//c=v1-v0
        c.v[0]=v1.v[0]-v0.v[0];
        c.v[1]=v1.v[1]-v0.v[1];
        c.v[2]=v1.v[2]-v0.v[2];
        c.v[3]=1.;
        return c;
}

vector3D_t operator - (vector3D_t v)
{
        vector3D_t c;//c=-v
        c.v[0]=-v.v[0];
        c.v[1]=-v.v[1];
        c.v[2]=-v.v[2];
        c.v[3]=1.;
        return c;
}
```

Edit dengan WPS Office

```
vector3D_t operator * (float r, vector3D_t b)
{
        vector3D_t c;//c=r*b
        int i;
        for (i=0;i<3;i++) {
                c.v[i]=r*b.v[i];
        }
        c.v[3]=1.;
        return c;
}

vector3D_t operator * (vector3D_t b, float r)
{
        vector3D_t c;//c=r*b
        int i;
        for (i=0;i<3;i++) {
                c.v[i]=r*b.v[i];
        }
        c.v[3]=1.;
        return c;
}

void setColor(float red,float green,float blue)
```

Edit dengan WPS Office

```
{
        glColor3f(red, green, blue);
}

void setColor(color_t col)
{
        glColor3f(col.r, col.g, col.b);
}

void drawDot(float x,float y)
{
        glBegin(GL_POINTS);
                glVertex2f(x, y);
        glEnd();
}

void drawDot(point2D_t p)
{
        glBegin(GL_POINTS);
                glVertex2f(p.x, p.y);
        glEnd();
}

void drawLine(float x1, float y1, float x2, float y2)
```

Edit dengan WPS Office

```
{
    glBegin(GL_LINES);
        glVertex2f(x1, y1);
        glVertex2f(x2, y2);
    glEnd();
}

void drawLine(point2D_t p1,point2D_t p2)
{
    drawLine(p1.x,p1.y,p2.x,p2.y);
}

//n: number of points
void drawPolyline(point2D_t pnt[],int n)
{
    int i;
    glBegin(GL_LINE_STRIP);
        for (i=0;i<n;i++) {
            glVertex2f(pnt[i].x, pnt[i].y);
        }
    glEnd();
}

//n: number of vertices
```

Edit dengan WPS Office

```
void drawPolygon(point2D_t pnt[],int n)
{
    int i;
    glBegin(GL_LINE_LOOP);
        for (i=0;i<n;i++) {
            glVertex2f(pnt[i].x, pnt[i].y);
        }
    glEnd();
}

// The function fillPolygon can fills only convex polygons
//n: number of vertices
void fillPolygon(point2D_t pnt[],int n,color_t color)
{
    int i;
    setColor(color);
    glBegin(GL_POLYGON);
        for (i=0;i<n;i++) {
            glVertex2f(pnt[i].x, pnt[i].y);
        }
    glEnd();
}

// The function gradatePolygon can fills only convex polygons
```

Edit dengan WPS Office

```
// The vertices will be painted with corresponding given colors.
// The points inside the polygon will be painted with the mixed color.
//n: number of vertices
void gradatePolygon(point2D_t pnt[],int num,color_t col[])
{
	int i;
	glBegin(GL_POLYGON);
		for (i=0;i<num;i++) {
			setColor(col[i]);
			glVertex2f(pnt[i].x, pnt[i].y);
		}
	glEnd();
}

//////////// End of OpenGL drawShape Functions ver 1 ////////////

void userdraw(void);

void display(void)
{
	glClear( GL_COLOR_BUFFER_BIT);
	userdraw();
	glutSwapBuffers();
}
```

Edit dengan WPS Office

```
//////////////////////////////////////////////////////////////////
void drawcharX(float x,float y)
{
        drawLine(x,y,x+10,y+12);drawLine(x,y+12,x+10,y);
}

void drawcharY(float x,float y)
{

drawLine(x+5,y,x+5,y+7);drawLine(x,y+12,x+5,y+7);drawLine(x+10,y+1
2,x+5,y+7);
}

void drawcharZ(float x,float y)
{

drawLine(x,y+12,x+10,y+12);drawLine(x+10,y+12,x,y);drawLine(x,y,x+1
0,y);
}

void drawAxes(matrix3D_t view)
{
#define HALFAXIS  220
#define HALFAXIS1 (HALFAXIS-10)
```

Edit dengan WPS Office

```
	point3D_t axes[14]={
		{-
HALFAXIS,0,0},{HALFAXIS,0,0},{HALFAXIS1,5,0},{HALFAXIS1,0,0},{0,0,0}
,
		{0,-
HALFAXIS,0},{0,HALFAXIS,0},{0,HALFAXIS1,5},{0,HALFAXIS1,0},{0,0,0},
		{0,0,-
HALFAXIS},{0,0,HALFAXIS},{5,0,HALFAXIS1},{0,0,HALFAXIS1}
	};
	vector3D_t vec[14];
	point2D_t buff[14];
	int i;
	for (i=0;i<14;i++) {
		vec[i]=Point2Vector(axes[i]);
		vec[i]=view*vec[i];
		buff[i]=Vector2Point2D(vec[i]);
	}
	drawPolyline(buff,14);
	drawcharX(buff[1].x,buff[1].y);
	drawcharY(buff[6].x,buff[6].y);
	drawcharZ(buff[11].x-14,buff[11].y);
}

////////////////////////////////////////////////////////////////////
typedef struct {
```

Edit dengan WPS Office

```
        int NumberofVertices; //in the face
        short int pnt[50];
        color_t col;
} face_t;
typedef struct {
        int NumberofVertices; //of the object
        point3D_t pnt[1600];
        color_t col[1600];
        int NumberofFaces; //of the object
        face_t fc[1000];
} object3D_t;

void draw3D(object3D_t obyek,matrix3D_t mat){
        vector3D_t vec[1600], vecbuff[50];
        vector3D_t vecNormal;
        point2D_t p[50];
        int i,j;
        for(i=0;i<obyek.NumberofVertices;i++){
                vec[i]=Point2Vector(obyek.pnt[i]);
                vec[i]=mat*vec[i];
        }
        setColor(1,0,0);
        for(i=0;i<obyek.NumberofFaces;i++){
                for(j=0;j<obyek.fc[i].NumberofVertices;j++)
```

Edit dengan WPS Office

```
            vecbuff[j]=vec[obyek.fc[i].pnt[j]];
        vecNormal=(vecbuff[1]-vecbuff[0])^(vecbuff[2]-vecbuff[0]);
        if(vecNormal.v[2]<0){
            for(j=0;j<obyek.fc[i].NumberofVertices;j++){
                p[j]=Vector2Point2D(vecbuff[j]);
            }
            drawPolygon(p,obyek.fc[i].NumberofVertices);
        }
    }
    setColor(1,1,1);
    for(i=0;i<obyek.NumberofFaces;i++){
        for(j=0;j<obyek.fc[i].NumberofVertices;j++)
            vecbuff[j]=vec[obyek.fc[i].pnt[j]];
        vecNormal=(vecbuff[1]-vecbuff[0])^(vecbuff[2]-vecbuff[0]);
        if(vecNormal.v[2]>=0){
            for(j=0;j<obyek.fc[i].NumberofVertices;j++){
                p[j]=Vector2Point2D(vecbuff[j]);
            }
            drawPolygon(p,obyek.fc[i].NumberofVertices);
        }
    }
}

//KODING SILINDER BERTUMPUK
```

Edit dengan WPS Office

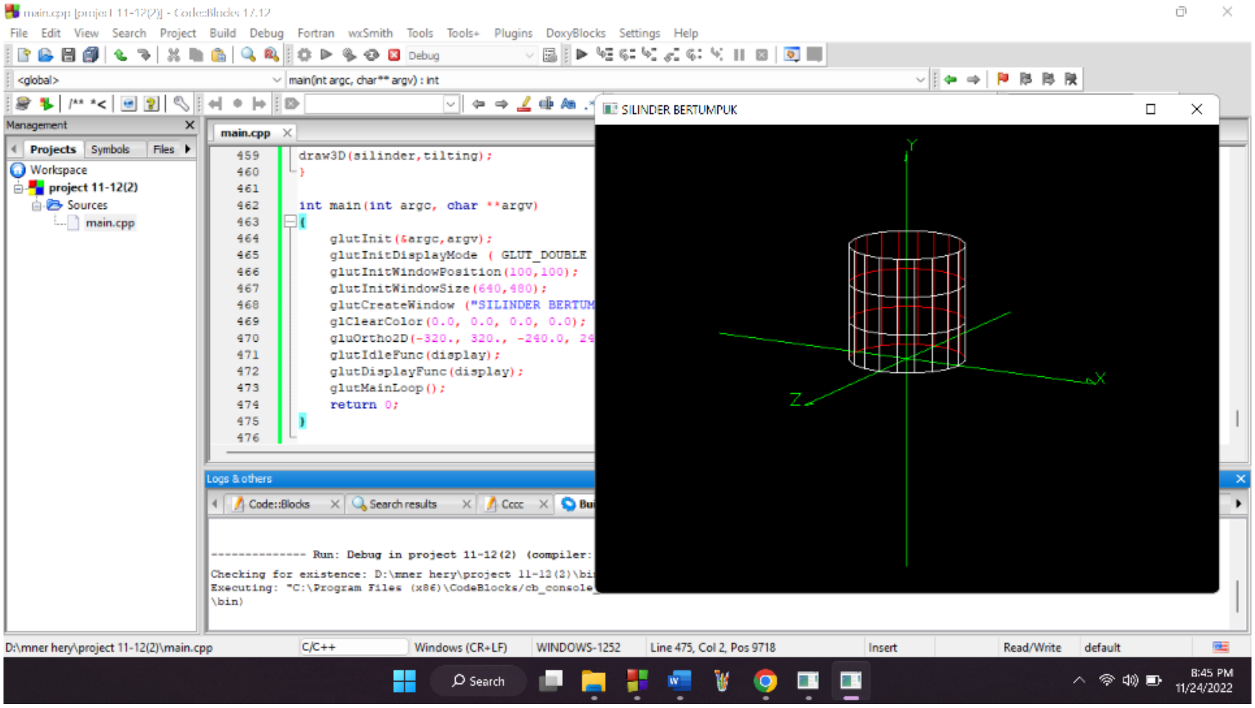
```
void createCylinderN (object3D_t &silinder,int m,int n,float r[],float
h[]){
 float a=6.26/n;
 float b=0;
 int i,j;
 silinder.NumberofVertices=(m+1)*n;
 for(i=0;i<=m;i++){
 if(i>0) b=b+h[i-1];
 for(j=0;j<n;j++){
 silinder.pnt[i*n+j].x=r[i]*cos(j*a);
 silinder.pnt[i*n+j].y=b;
 silinder.pnt[i*n+j].z=r[i]*sin(j*a);
 }
 }
 silinder.NumberofFaces=m*n+2;
 for(i=0;i<m;i++){
 for(j=0;j<n;j++){
 silinder.fc[i*n+j].NumberofVertices=4;
 silinder.fc[i*n+j].pnt[0]=i*n+j;
 silinder.fc[i*n+j].pnt[1]=(i+1)*n+j;
 silinder.fc[i*n+j].pnt[2]=(i+1)*n+j+1;
 silinder.fc[i*n+j].pnt[3]=i*n+j+1;
 if(j==(n-1)){
 silinder.fc[i*n+j].pnt[2]=i*n+j+1;
```

Edit dengan WPS Office

```
 silinder.fc[i*n+j].pnt[3]=(i-1)*n+j+1;
 }
 }
 }
 silinder.fc[m*n].NumberofVertices=n;
 for(i=0;i<n;i++) silinder.fc[m*n].pnt[i]=i;
 silinder.fc[m*n+1].NumberofVertices=n;
 for(i=0;i<n;i++)
silinder.fc[m*n+1].pnt[i]=(m+1)*n-1-i;
}

void userdraw(void)
{
matrix3D_t tilting=rotationXMTX(0.25)*rotationYMTX(-0.5);
setColor(0,1,0);
drawAxes(tilting);
float r[4]={60,60,60,60};
float h[3]={40,40,40};
object3D_t silinder;
createCylinderN(silinder,3,20,r,h);
setColor(0,0,0);
draw3D(silinder,tilting);
}
```

Edit dengan WPS Office

```
int main(int argc, char **argv)
{
        glutInit(&argc,argv);
        glutInitDisplayMode ( GLUT_DOUBLE | GLUT_RGB );
        glutInitWindowPosition(100,100);
        glutInitWindowSize(640,480);
        glutCreateWindow ("SILINDER BERTUMPUK");
        glClearColor(0.0, 0.0, 0.0, 0.0);
        gluOrtho2D(-320., 320., -240.0, 240.0);
        glutIdleFunc(display);
        glutDisplayFunc(display);
        glutMainLoop();
        return 0;
}
```



```
    draw3D(silinder,tilting);
}

int main(int argc, char **argv)
{
    glutInit(&argc,argv);
    glutInitDisplayMode ( GLUT_DOUBLE
    glutInitWindowPosition(100,100);
    glutInitWindowSize(640,480);
    glutCreateWindow ("SILINDER BERTUM
    glClearColor(0.0, 0.0, 0.0, 0.0);
    gluOrtho2D(-320., 320., -240.0, 24
    glutIdleFunc(display);
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

```
-------------- Run: Debug in project 11-12(2) (compiler:
Checking for existence: D:\mner hery\project 11-12(2)\bi
Executing: "C:\Program Files (x86)\CodeBlocks/cb_console
\bin)
```

D:\mner hery\project 11-12(2)\main.cpp

Line 475, Col 2, Pos 9718

Edit dengan WPS Office