НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

кафедра Технічної кібернетики

**Трьохвимірна графіка**

UA.ІК.-32070-01 01

*Курсова робота*

*дисципліна*

«Об’єктно орієнтоване програмування»

Керівник Виконавець:

Доц. Лісовиченко О.І. Боклашко А.Т.

«Захист дозволено» гр. ІК-32

« » 2014р.

Захищено з оцінкою

2014

**Зміст**

**Зм. Арк № докум. Підпис Дата**

**НТУУ “КПІ”**

**ФІОТ**

**Літ. Маса Мірило**

**Перевір. Л**

**Аркуш Аркушів**

**Кафедра**

**Технічної кібернетики**

**Розробник**

1. **Вступ..…………………………...…………………………………..…… 4**
2. **Постановка задачі ……………………………………………………... 5**
3. **Посібник розробника ….………………………………………………. 6**
4. **Посібник користувача .…………………………………………….….. 7**
5. **Висновки ………………………………………………………………...10**
6. **Список літератури …...……………………….………………………..11**
7. **Додаток 1 (Лістинг програми)**
8. **Додаток 2 (Графічний матеріал)**
9. **Додаток 3 (CD-диск з програмою)**

**Вступ**

UA.IK.32070-01 01

4

Лист

Дата

Підп.

№ докум.

Лист

змн

Трьохвимірна комп’ютерна графіка – це область комп’ютерних технологій, яка в наш час набуває стрімкого розвитку. Це обумовлено використанням її результатів у рекламному бізнесі, Інтернеті, телебаченні та кіноіндустрії, в проектуванні та моделюванні різного роду складних систем, як у народному господарстві, так і у військовій сфері.

Для створення та обробки комп’ютерної графіки існує багато пакетів програм та різного роду редакторів: 3D Studio Max, LightWave 3D, Maya, Simply 3D, TrueSpace 3.0 та багато інших. Основними їхніми функціями є створення та редагування трьох- вимірних об’єктів, їх обробка за допомогою різноманітних інструментів, збереження у файл тощо.

Недоліками більшості редакторів є те, що вони доволі складні в освоєнні і мають громіздкий інтерфейс, а тому початківцям важко працювати з ними.

Тому в даній роботі було поставлено завдання – створити нескладний редактор трьохвимірної графіки для побудови, візуалізації, редагування та збереження найпростіших 3D об’єктів.

**Постановка задачі**

UA.IK.32070-01 01

5

Лист

Дата

Підп.

№ докум.

Лист

змн

Потрібно розробити програму – редактор трьохвимірної графіки, яка матиме такі функції:

- Створення площин, паралелепіпедів, дисків, сфер, пірамід, конусів та циліндрів, а також так званих «тіл обертання» та «тіл напряму»;

* Вибір кольору 3D-об’єктів;
* Можливість редагування: обертання, переміщення, зміни розміру, видалення вибраних об’єктів
* Можливість збереження створеної сцени у файл
* Можливість відкриття та завантаження сцени з файлу

**Посібник розробника**

UA.IK.32070-01 01

01

6

Лист

Дата

Підп.

№ докум.

Лист

змн

по основних функціях програми

1. **DI\_Init()**

Ініціалізація інтерфейсу DirectX Input8

1. **gfPlane(Vector3 p1, Vector3 p2)**

Створює у площині Oxy прямокутник з діагональними точками p1і p2

1. **gfBox(Vector3 p1, Vector3 p2)**

Створює паралелепіпед з сторонами паралельними координатним площинам та діагональними вершинами p1 і p2

1. **gfSphere(Vector3 p1, double r, double m, double n)**

Створює сферу з центром p1, радіусом r, та кількістю полігонів m\*n

1. **gfPyramid(Vector3 p1, Vector3 p2, Vector3 p3)**

Створює піраміду з основою – прямокутником з діагональними вершинми p1 і p2 і вершиною у точці p3

1. **gfDisc(Vector3 p1, double r1, double r2, double n, double grad)**

Створює диск з центром у точці p1, зовнішнім радіусом r1, внутрішнім радіусом r2, кількістю полігонів n та кутовим розміром grad

1. **gfCylinder(Vector 3 p1, double r, double n, double grad, double l\_2)**

Створює циліндр з центром у точці p1, радіусом r, висотою 2\*l\_2, кутовим розміром grad та кількістю поліномів n

1. gfConys**(Vector 3 p1, double r1, double r2, double n, double grad, double l\_2**)

Створює конус(зрізаний конус) з центром у точці p1, зовнішнім та внутрішнім радіусами r1 та r2 , висотою 2\*l\_2, кутовим розміром grad та кількістю поліномів n

1. **keyboard()**

Обробник клавіатури з використанням DirectX Input8

1. **CreateContext()**

Ініціалізує інтерфейс OpenGl, що застосовується для рендеринга 3D-об‘єктів

1. **Myfunc()**

Основний цикл програми де відбувається рендеринг

**Посібник користувача**

**Системні вимоги**

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

**Операційна система**: Windows XP SP3, Windows Vista (всі версії), Windows 7 ;

**Процесор** з тактовою частотою 2 ГГц і більше;

**Память**: 1024 Мб і більше;

**Відеокарта**: сумісна з OpenGL 3.0 і DirectX 9.0c, з обсягом відеопам′яті 512 Мб і більше;

**Дисплей** з роздільною здатністю 1280x1024;

**Вільний простір на жорсткому диску** для установки програми ( 21 Мб ) і додатковий вільний дисковий простір для збереження результатів роботи;

**Миша** обов′язково;

**CD-DVD-ROM** або **USB-port** для установки програми**;**

NETFramework 4.5;

Операційна система: Windows 7, Windows 8.

Вид головного вікна програми:

UA.IK.32070-01 01

7

Лист

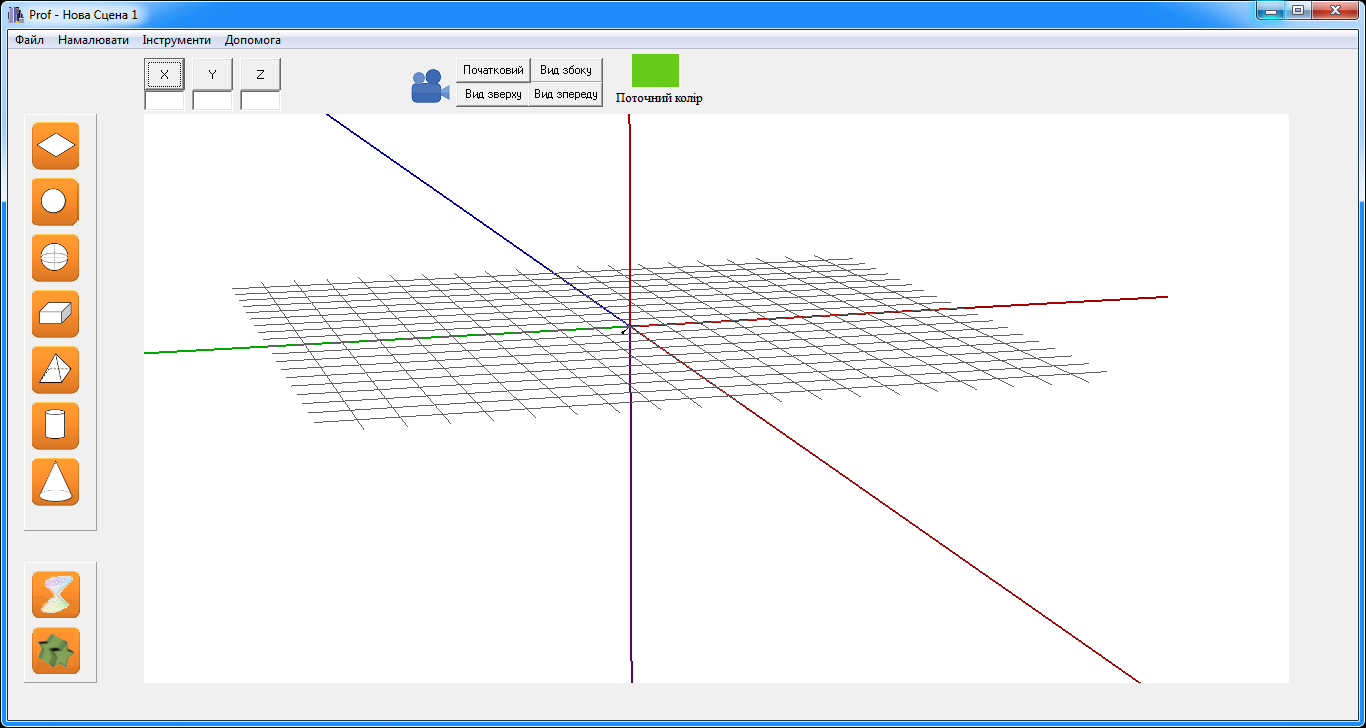
Дата

Підп.

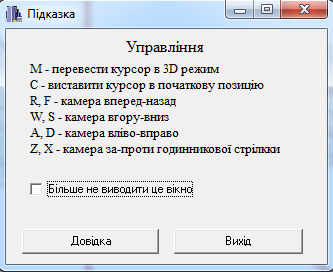
№ докум.

Лист

змн



При першому запуску програми виводитиметься підказка користувачеві:



**Використання програми**

Основним елементом робочого середовища є 3D простір, у якому і здійснюватимуться всі побудови і всі дії, пов′язані з їх переглядом та редагуванням. 3D простір програми - це білий фон, на якому нанесені координатні осі та сітка. Координати в тривимірному просторі формують трійку (x, y, z). Координати x, y, z для тривимірної декартової системи можна розуміти як відстані від точки до відповідних площин: yz, xz, та xy. Тривимірна декартова система координат є дуже популярною, тому що відповідає звичним уявам про просторові виміри — висоту, ширину та довжину (тобто три виміри).

Знизу вверх розташовується вісь ординат (Y), зліва направо розташовується вісь абсцис (X) і з екрану «на користувача» – проходить вісь аплікат (Z). Додатні напрями цих осей зображені червоним кольором, а від’ємні: вісь X – зеленим кольором, вісь Y фіолетовим і синім вісь Z. Також на площину XZ нанесена координатна сітка. Одна клітинка цієї сітки має розміри 1×1.

І найважливішим елементом 3D простору є 3D курсор.3D курсор має вигляд білої стрілки. Для того, щоб перейти від звичайного курсора до 3D курсора і навпаки, потрібно натиснути клавішу з англійською літерою « M ». 3D курсор рухається наступним чином: при русі мишею ззаду вперед і зліва направо – 3D курсор рухатиметься з від’ємного у додатній напрям осей X і Z відповідно. При русі коліщатком миші ззаду вперед - 3D курсор рухатиметься з від’ємного у додатному напрямі осі Y. Якщо ж коліщатко миші затиснути, то при русі миші ззаду вперед 3D курсор буде рухатися знизу вверх по осі ординат.

**Завантаження сцени з файлу:**

UA.IK.32070-01 01

7

Лист

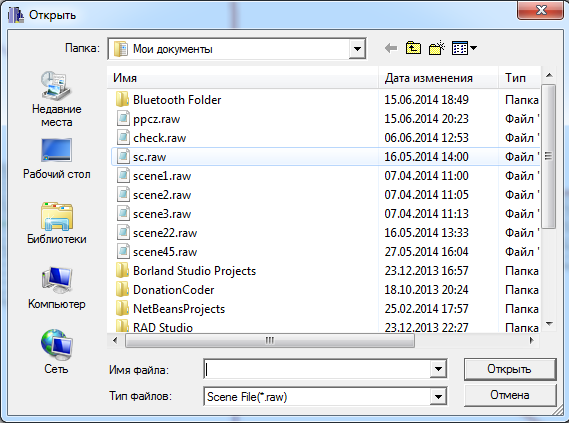
Дата

Підп.

№ докум.

Лист

змн



**Збереження сцени у файл:**

UA.IK.32070-01 01

7

Лист

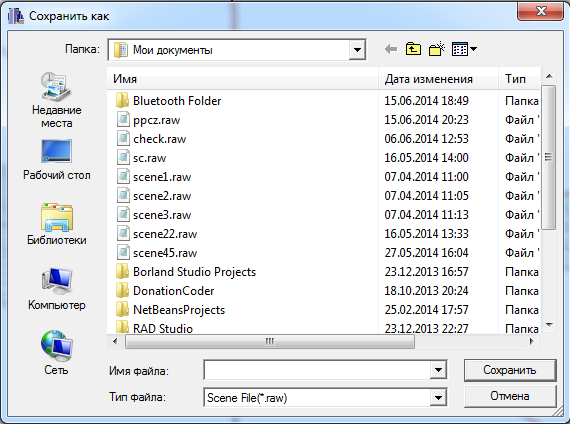
Дата

Підп.

№ докум.

Лист

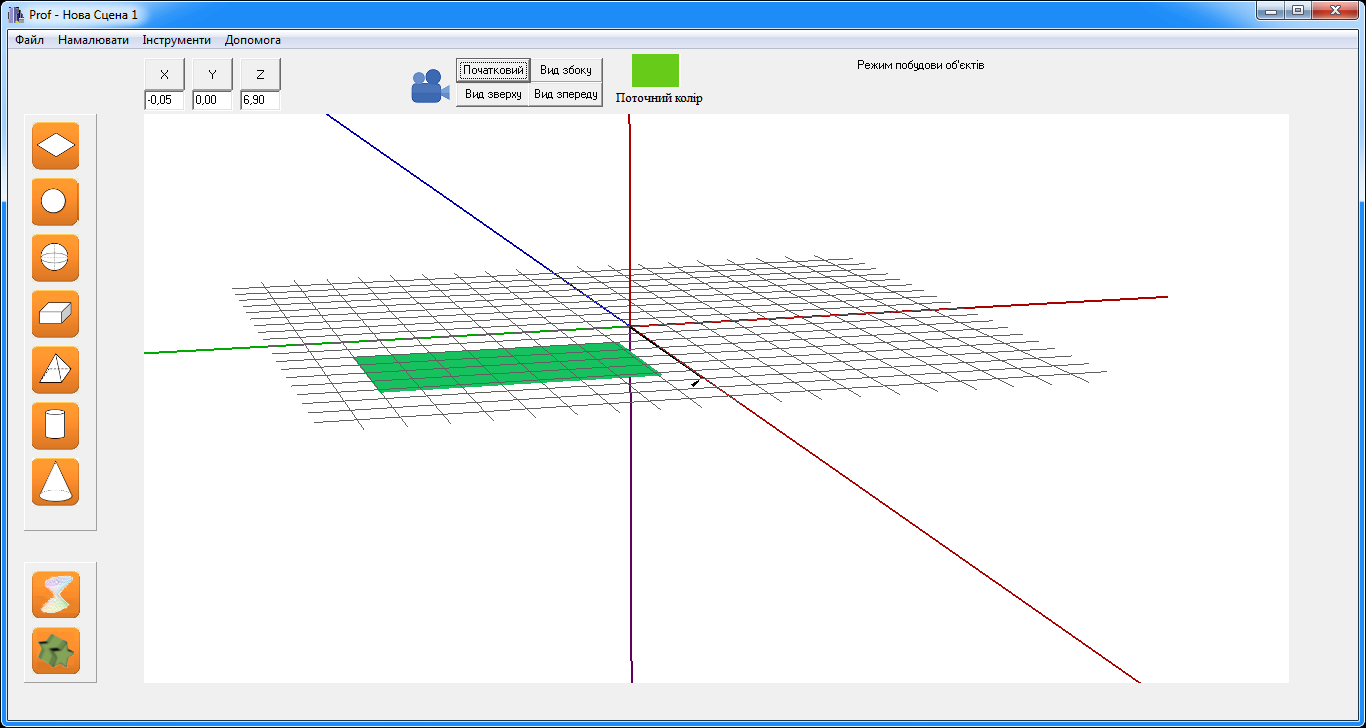
змн



**Побудова площини:**

При виборі інструменту « **Площина** » у змодельованому 3D просторі в центрі вікна, можна побудувати площину.

В програмі Prof площина – це прямокутник, який паралельний площині XZ, тому її можна визначити двома точками. Для побудови площини необхідно затиснути ліву клавішу миші для визначення першої точки і відпустити її в тому місці, де має бути точка діагонально протилежна першій. Після цього буде змодельована площина поточним кольором ( за замовчуванням він зелений ). Але поточний колір завжди можна змінити на інший, клікнувши лівою клавішею миші по індикатору відображення кольору у верхній частині вікна. Після цього, у палітрі кольорів, що з′явиться, вибрати бажаний колір.



**Побудова диска:**

« **Диск** » – інсртумент, призначений для побудови дисків. У програмі Prof диск може бути побудований у двох видах:

* Звичайний диск;
* Диск з отвором в центрі.

В першому випадку достатньо просто один раз натиснути ліву клавішу миші в точці, де має бути центр диска, а другий раз на відстані радіуса диска від його центра.

UA.IK.32070-01 01

7

Лист

Дата

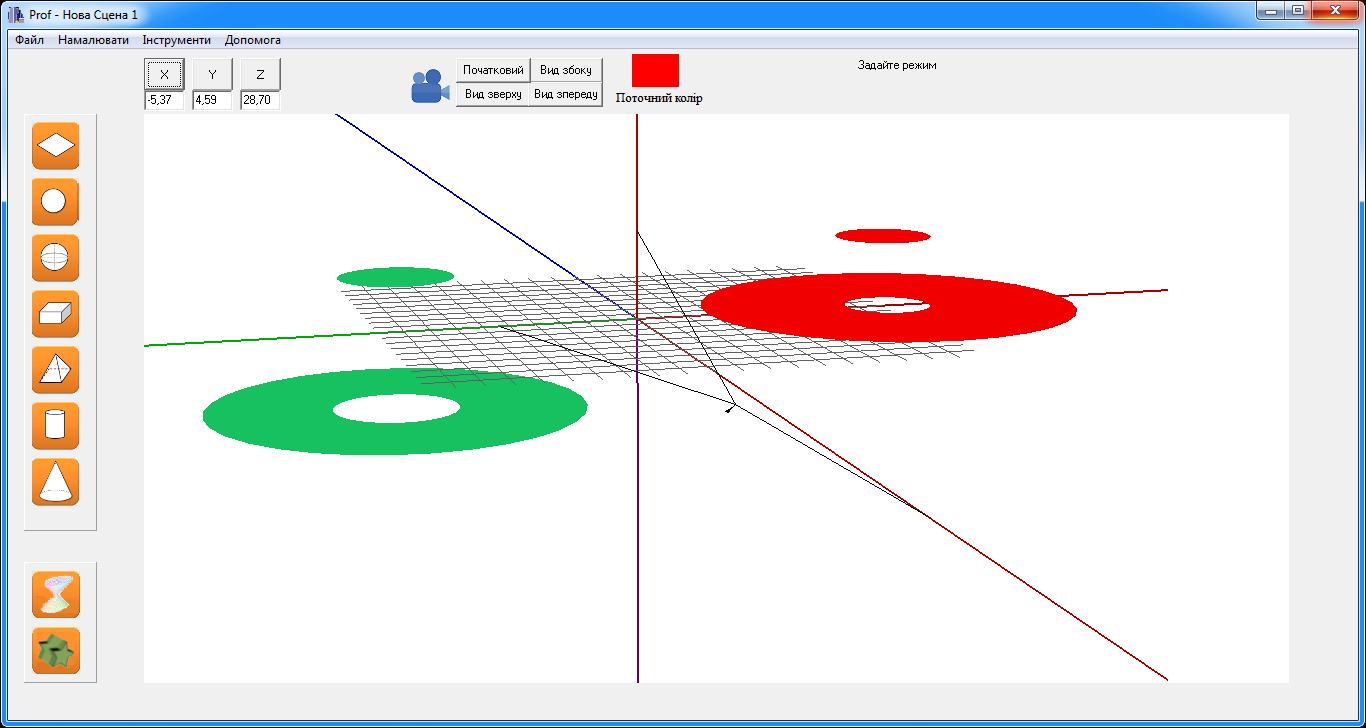
Підп.

№ докум.

Лист

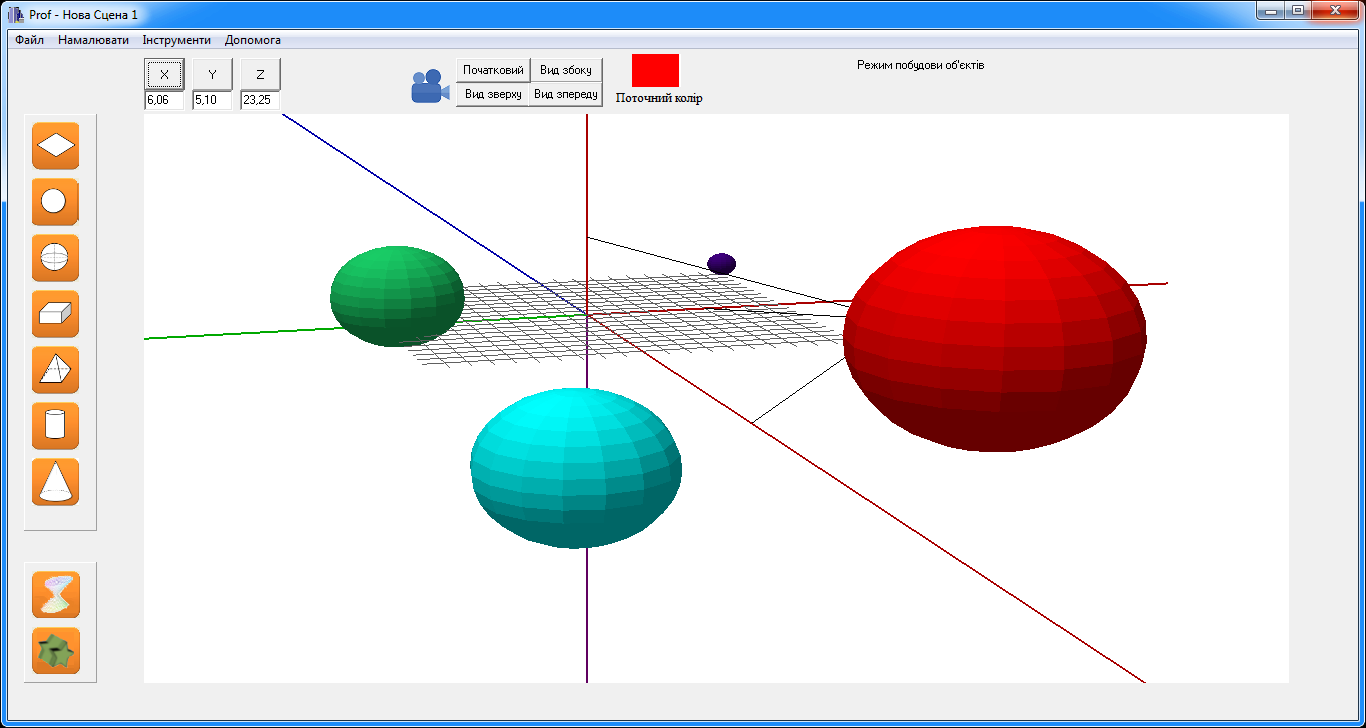
змн

В другому випадку потрібно затиснути ліву клавішу миші в центрі диска і відпустити на відстані радіуса отвору диска, потім ще раз натиснути на відстані радіуса самого диска.



**Побудова сфери:**

Для побудови сфери потрібно задати її центр та будь-яку точку на її поверхні, таким чином радіусом сфери буде відстань між цими двома точками:



**Побудова паралелепіпеда:**

Піктограма « **Паралелепіпед** » – призначена для побудови паралелепіпедів. Паралелепіпед будується аналогічно як і площина - через задання двох діагонально протилежних вершин.

UA.IK.32070-01 01

7

Лист

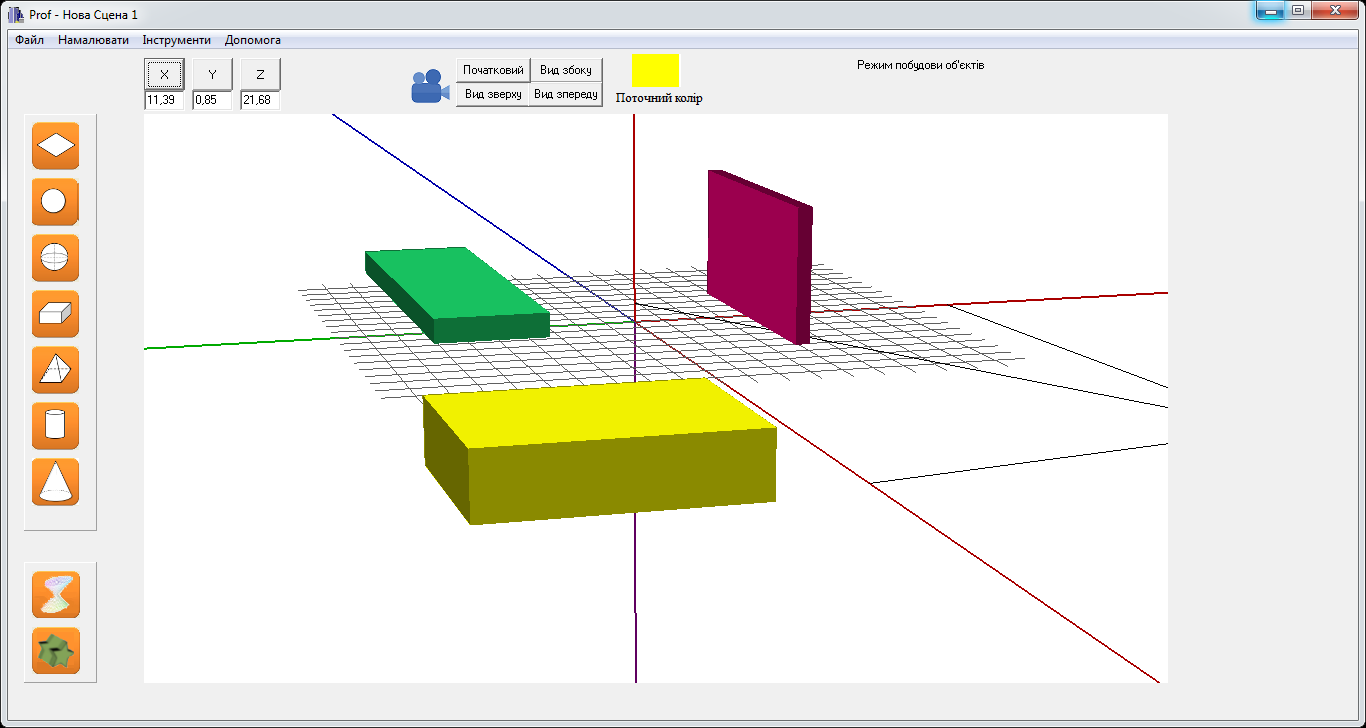
Дата

Підп.

№ докум.

Лист

змн

****

UA.IK.32070-01 01

7

Лист

Дата

Підп.

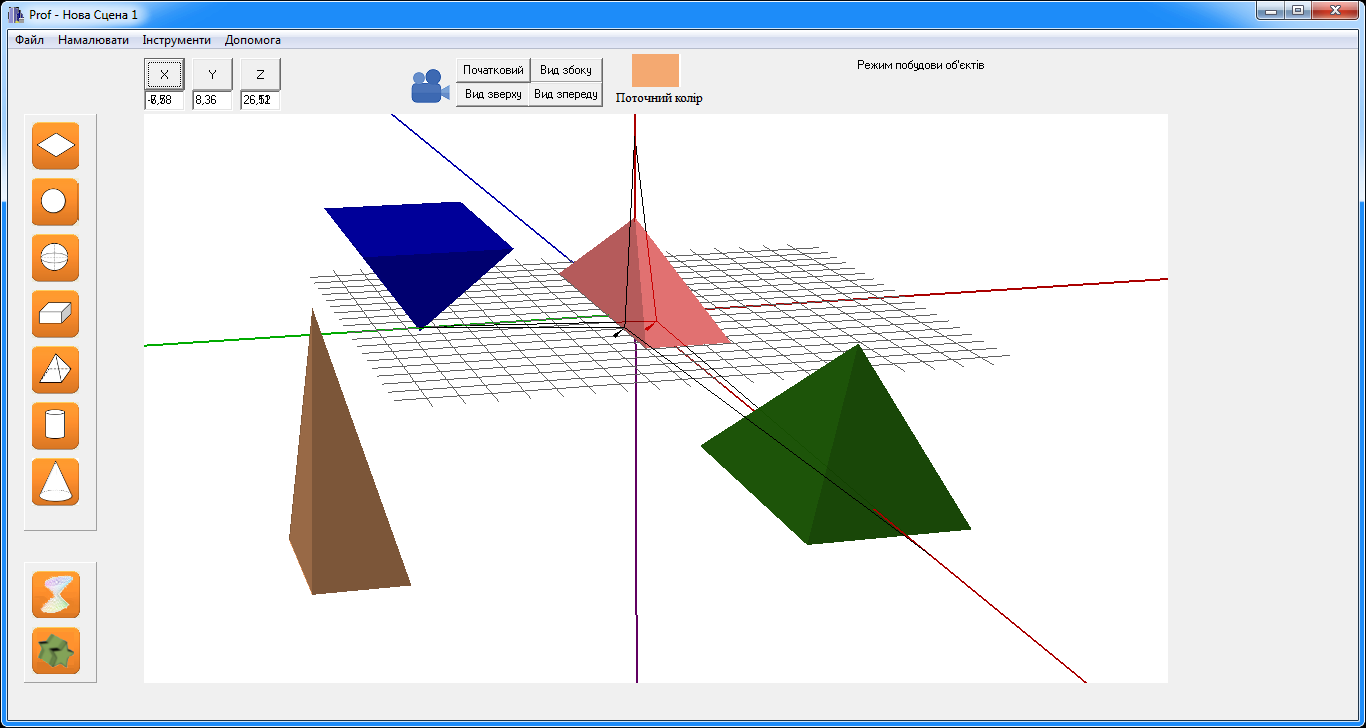
№ докум.

Лист

змн

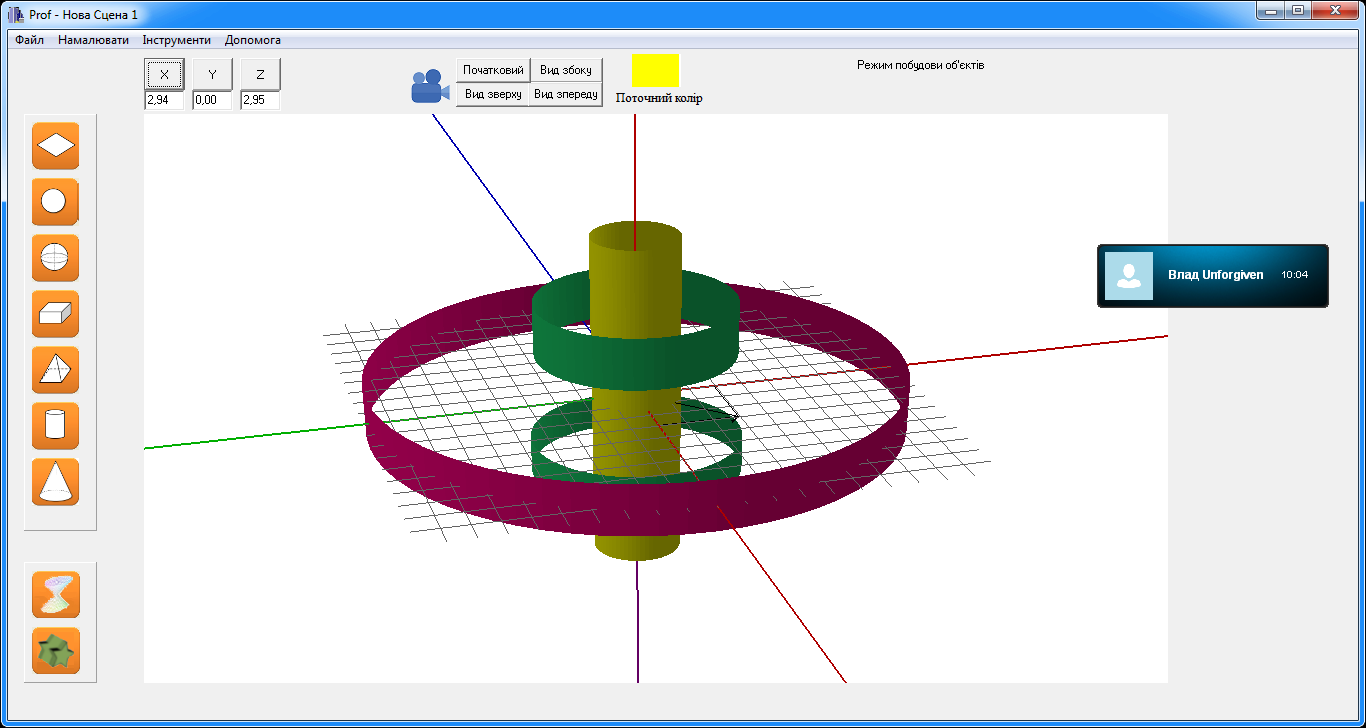
**Побудова піраміди:**

Для побудови піраміди використовується інструмент з піктограмою «**Піраміда**» Основою піраміди є площина. Тому для її побудови спочатку задається площина, а потім ще одна точка, яка є вершиною піраміди.



**Побудова циліндра:**

Для побудови циліндра використовується піктограма « **Циліндр** ». Щоб побудувати циліндр потрібно натиснути ліву клавішу миші там, де має бути центр його основи і відпустити на відстані радіуса основи. А потім вказати точку, яка буде на відстані висоти від центру основи.



**Побудова конуса:**

Для побудови конуса або зрізаного конуса використовується піктограма «**Конус/Зрізаний конус** ». Будуються ці фігури так: спочатку задається центр однієї основи і її радіус, аналогічно, як при побудові цилідра, а потім таким же чином задається центр і радіус іншої основи, причому центри основ мають лежати на осі паралельній Oy, інакше програма автоматично змінить центр другої основи так, щоб вона лежала на одній осі з першою, радіус другої основи при цьому не зміниться.

UA.IK.32070-01 01

7

Лист

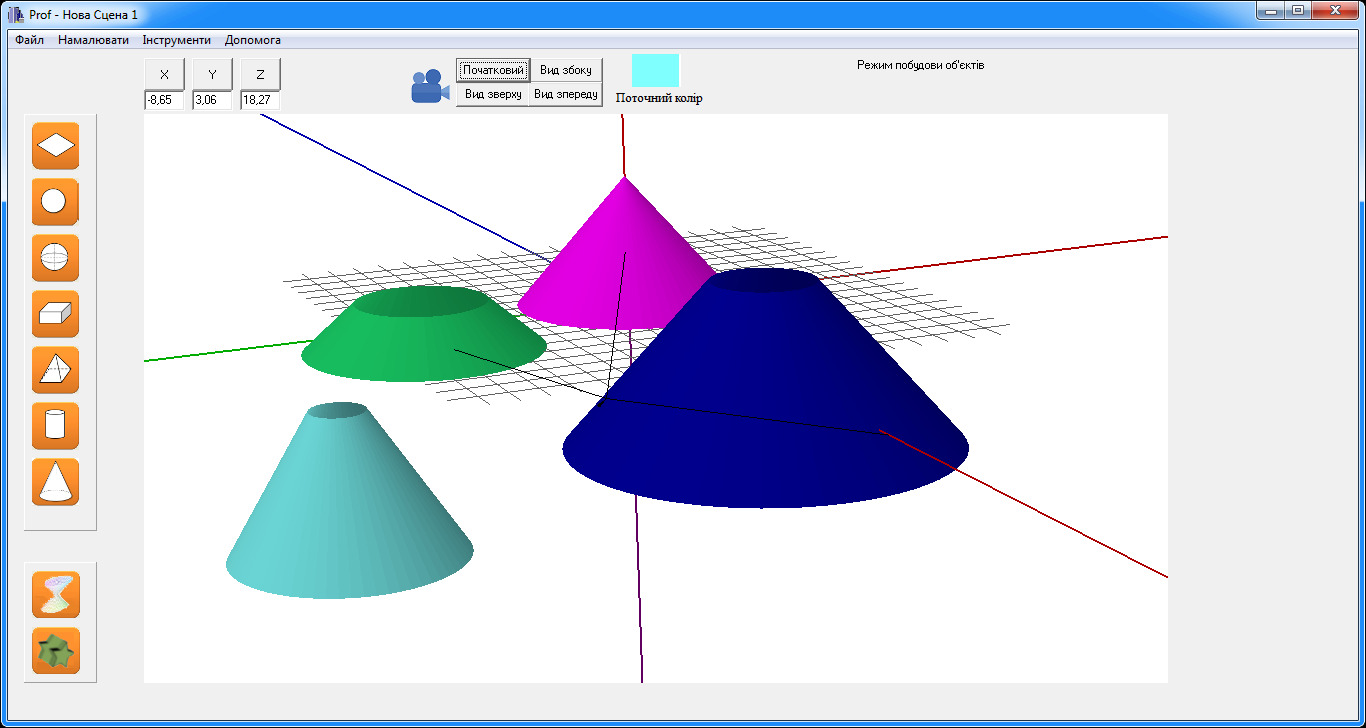
Дата

Підп.

№ докум.

Лист

змн



UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

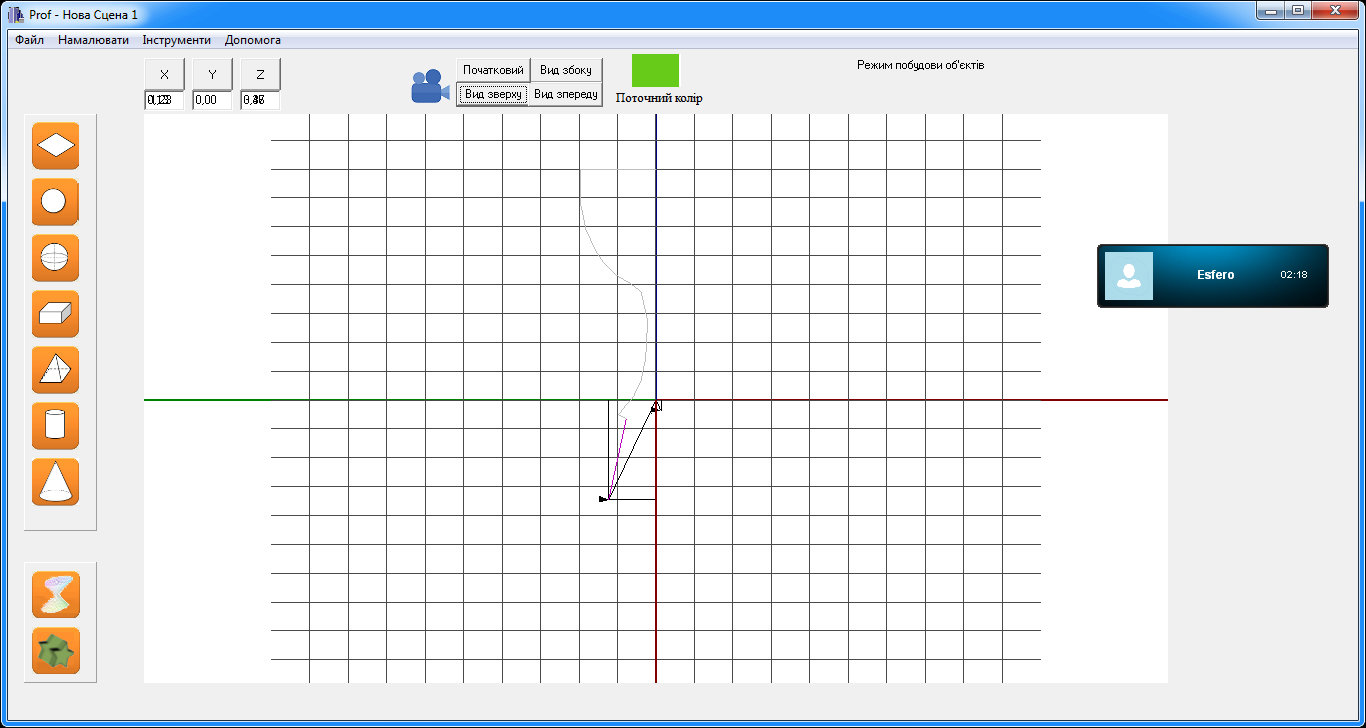
Лист

змн

**Побудова «тіл обертання»:**

« **Тіло обертання** » - поверхня, утворена шляхом обертання деякої твірної лінії або контуру навколо нерухомої осі.

У програмі ми спочатку задаємо контур, а поверхня буде утворена обертанням контуру навколо осі утвореної початковою та кінцевою точками цього контуру.



UA.IK.32070-01 01

7

Лист

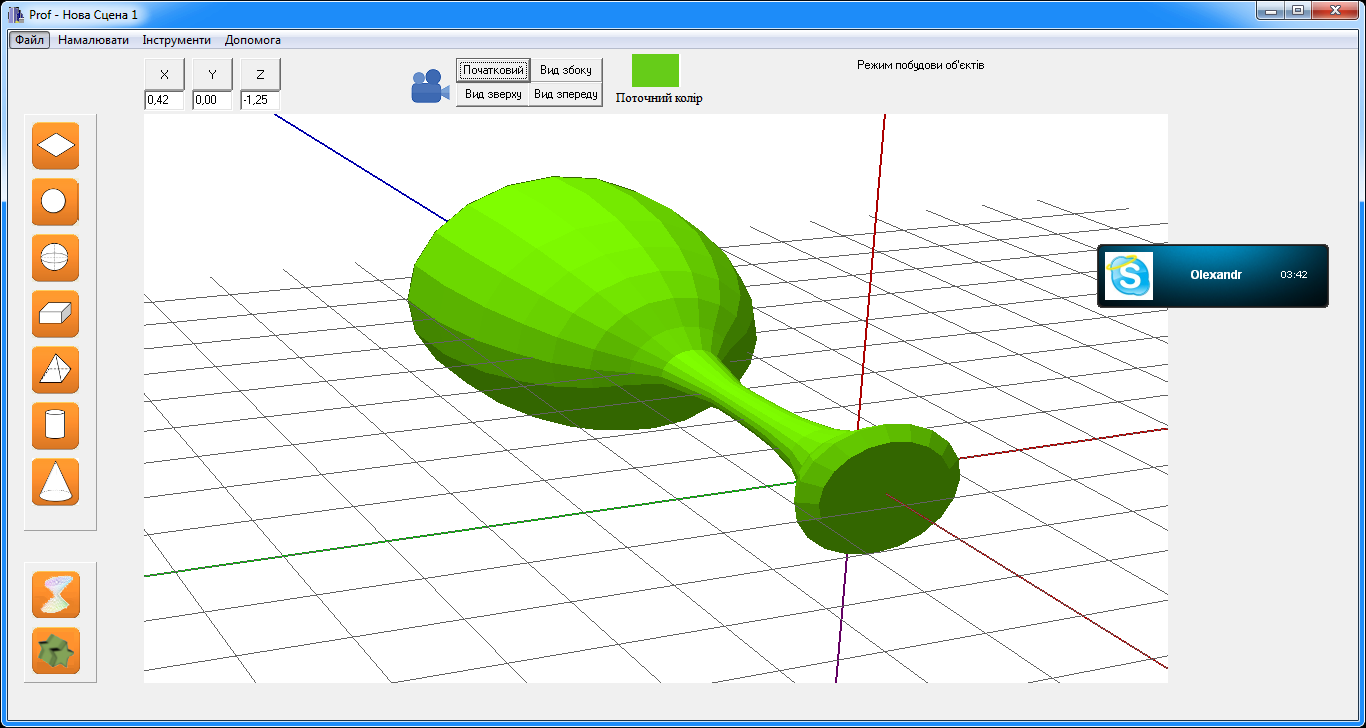
Дата

Підп.

№ докум.

Лист

змн



**Побудова тіл напряму:**

« **Тіло напряму** » - піктограма для побудови поверхонь другого порядку шляхом переміщення деякої твірної лінії або контуру у вказаному напрямі. Такі поверхні у математиці мають ще назву – **кінематичні поверхні**. Кінематичну поверхню ( тіло напряму ) визначають:

* твірна;
* напрямні елементи;
* умови зміни розташування та форми твірної під час руху, які задають її алгоритмічну частину.

Сукупність положень твірної складає лінійчатий каркас поверхні. Така поверхня неперервна.

Тіла напряму - це незакономірні поверхні, що називаються також **топографічними**.

1 контур

UA.IK.32070-01 01

7

Лист

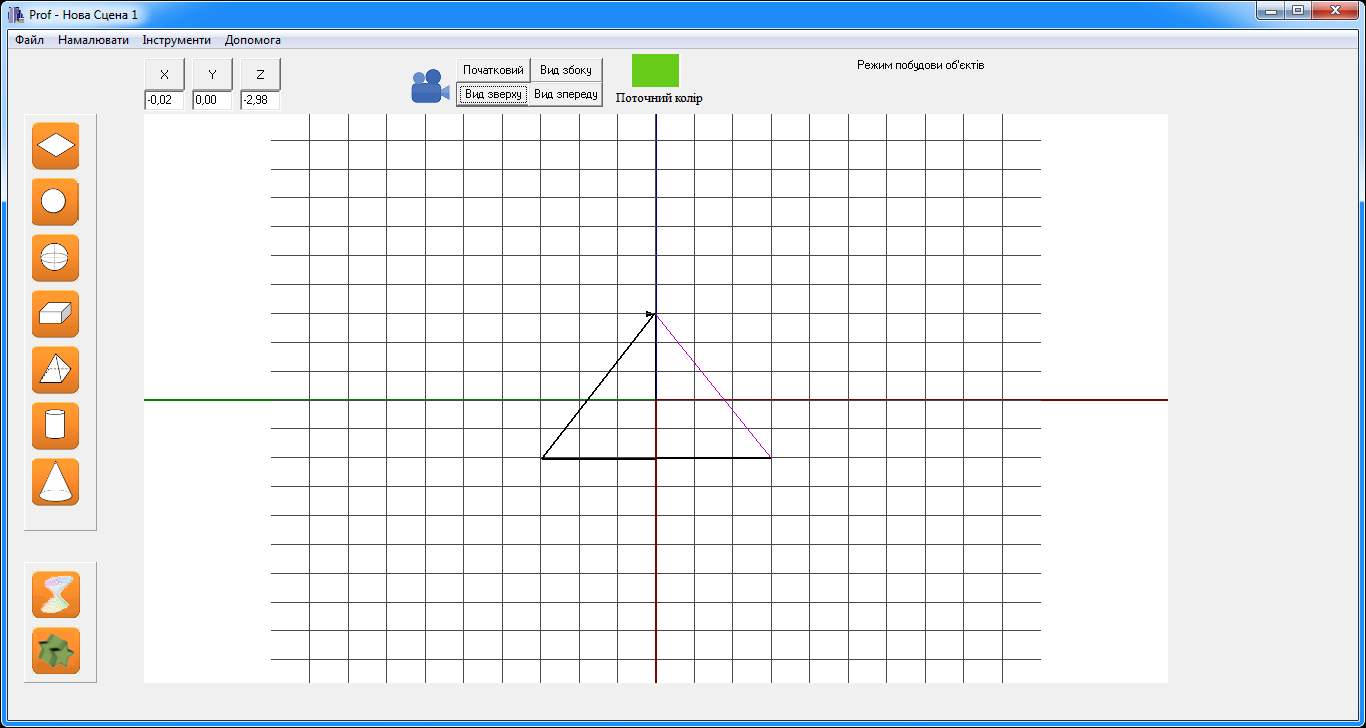
Дата

Підп.

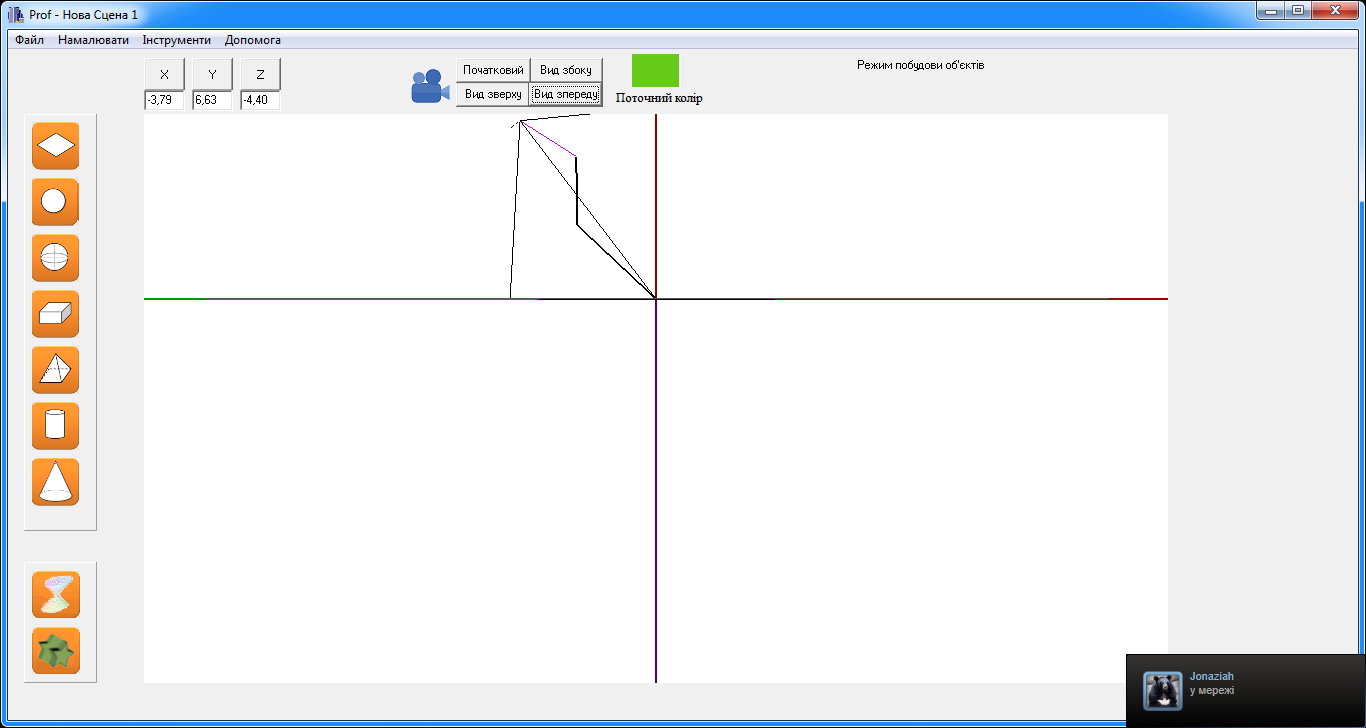
№ докум.

Лист

змн



2 контур:



Результат:

Результат:

UA.IK.32070-01 01

7

Лист

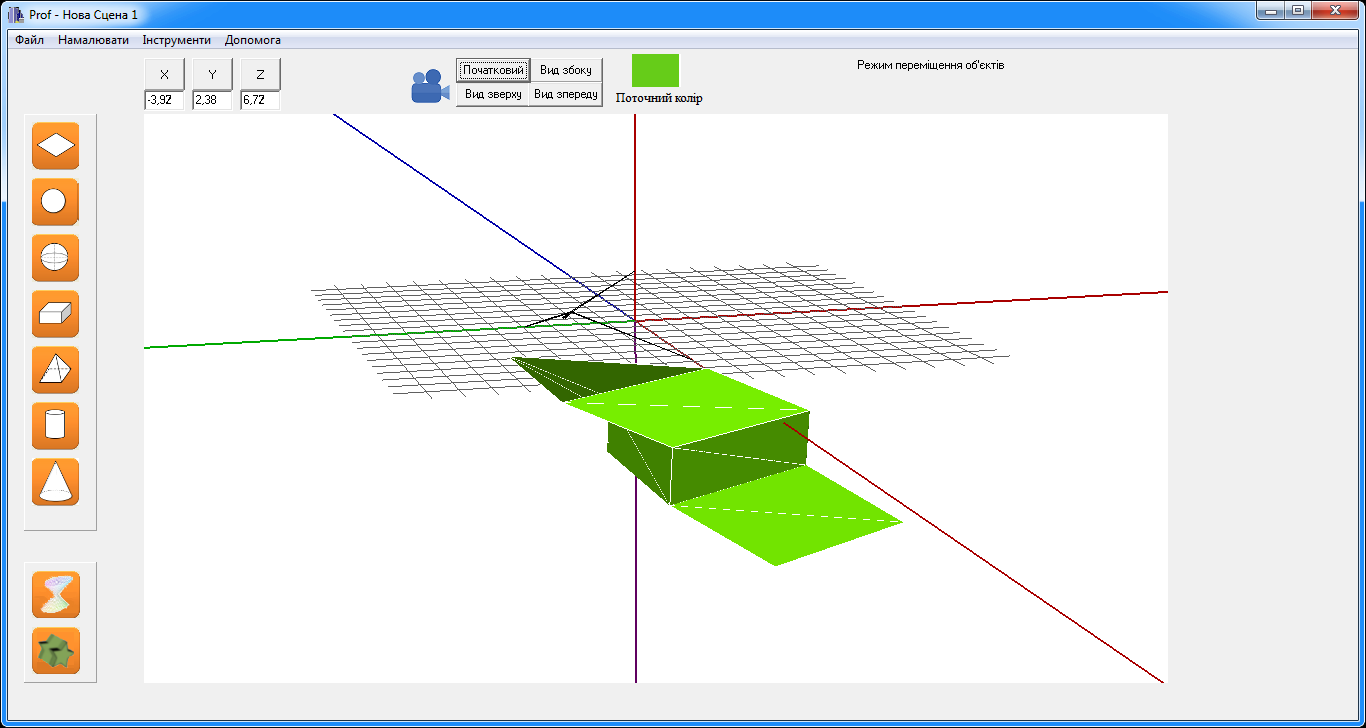
Дата

Підп.

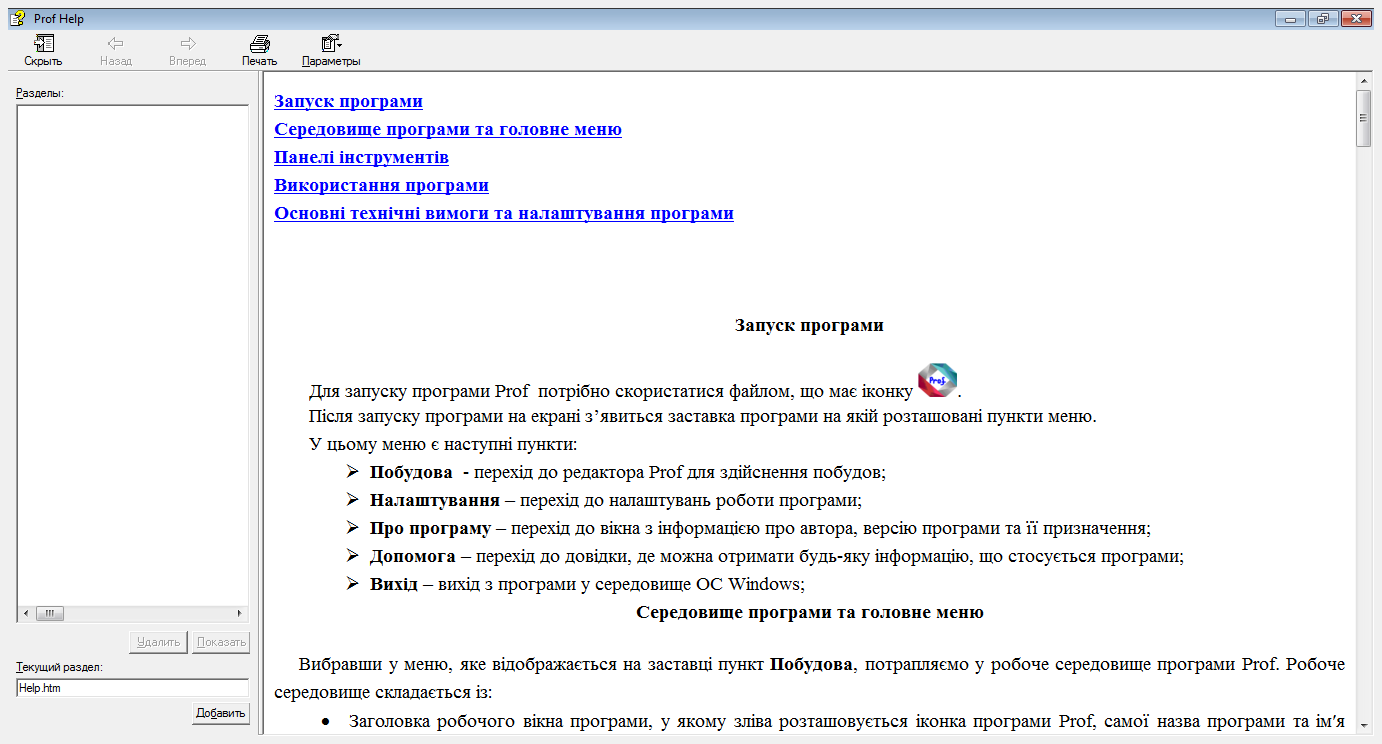
№ докум.

Лист

змн



Допомога:

****

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

**Висновок**

Створена програма – редактор тривимірної графіки, володіє основними базовими функціями необхідними для відкриття, збереження та редагування тривимірної сцени і має непоганий набір фігур, які можна побудувати. Вона має простий та доволі зручний інтерфейс, користувачу неважко її освоювати, оскільки є вбудована допомога крім того програма має невеликий розмір. Таким чином можна підсумувати, що програма добре підходить для користувачів, які лише починають освоювати тривимірну графіку для вивчення тривимірного середовища та основ побудови тривимірних об’єктів, а потім зможуть перейти на більш професійні пакети програм для роботи з тривимірною графікою.

**Список використаної літератури**

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

1. Райт, Липчак - OpenGL Суперкнига (2006)
2. Рост - OpenGL Трехмерная графика и язык программирования шейдеров (2005)
3. Фленов - DirectX и C++ Искусство программирования (2006).djvu
4. Херн, Бейкер - Компьютерная графика и стандарт OpenGL (2005).djvu
5. pmg.org.ru – Программирование магических игр
6. gamedev.ru – Разработка игр

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

**Додаток 1**

**Лістинги деяких функцій програми**

Основний цикл програми:

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

inline void \_\_fastcall TForm1::myfunc(TObject\*Sender,bool &don)

{

static PerfTimer Timer;

if(!hThreadHandle)

{

done=true;

if(!CreateContext())

{

MessageBox( NULL, "Помилка Ініціалізації",

"SHUTDOWN ERROR",MB\_OK | MB\_ICONINFORMATION );

Form1->Close();

}

ColorDialog1->Color=0x19CC66;

for(int i=0;i<Image1->Width;i++)

for(int j=0;j<Image1->Height;j++)

Image10->Canvas->Pixels[i][j]=0x19CC66;//0.098f,0.8f,0.4f;

hThreadHandle =(HANDLE) \_beginthread(keyboard,0,NULL);

Timer.Start();

}

if(Timer.GetRealTime()>16.6f)

{

Timer.Restart();

Vis();

DrawScene();

SwapBuffers( hDC );

}

}

Функція обробки клавіатури, миші та створення об’єктів:

inline void keyboard(void\*)

{

register PerfTimer kTimer;

Vector3 scale(1.0f,1.0f,1.0f),ang;

bool secDraw=false;

kTimer.Start();

while(Form1->done)

{

if(kTimer.GetRealTime()>16.6f)

{

kTimer.Restart();

//EnterCriticalSection(&Form1->CS);

mouse.Update();

//KeyBoard

HRESULT hr = g\_KDIDev->GetDeviceState(sizeof(buffer), &buffer);

if(FAILED(hr))g\_KDIDev->Acquire();

/\*MessageBox( NULL,"Помилка DirectX",

"RUNTIME ERROR", MB\_OK |MB\_ICONINFORMATION );

/\* HRESULT hr=0;

while (!hr) hr= g\_KDIDev->Acquire(); \*/

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

if((buffer[DIK\_M]&0x80)&&!pm)

{

pm=true;

m=!m;

if(m)

mouse.Set3DMode();

else

mouse.Set2DMode();

}

if(mouse.Is3D)

{

if(mode==GL\_DRAW)

{

switch(primMode)

{

case GL\_NULL\_MODE :break;

case GL\_PLANE :

{

if(mouse.CheckLeft())

{

Draw=true;

}

if(!mouse.KeepLeft()&&Draw)

{

gfPlane(mouse.LDownPos,mouse.Locate,2,2,&sc);

Draw=false;

}p

else if(Draw)

gfPlaneDraw(mouse.LDownPos,mouse.Locate,&sc);

break;

}

case GL\_SPHERE :

{

if(mouse.CheckLeft())

Draw=true;

if(!mouse.KeepLeft()&&Draw)

{

gfSphere(mouse.LDownPos,mouse.DeltaLPos().Abs(),18,18,&sc);

Draw=false;

}

break;

}

case GL\_BOX :

{

if(mouse.CheckLeft())

Draw=true;

if(!mouse.KeepLeft()&&Draw)

{

gfBox(mouse.LDownPos,mouse.Locate,&sc);

Draw=false;

}

break;

}

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

case GL\_PYRAMID :

{

static Vector3 Vert1,Vert2;

if(mouse.CheckLeft())

if(!secDraw)

Draw=true;

else

{

gfPyramid(Vert1,Vert2,mouse.Locate,&sc);

Draw=false;

secDraw=false;

Vert1=Vert2=nil3;

}

if(!mouse.KeepLeft()&&Draw)

{

Vert1=mouse.LDownPos;

Vert2=mouse.Locate;

secDraw=true;

Draw=false;

}

break;

}

case GL\_KONYS :

{

static Vector3 Vert1,Vert2,Vert3,Vert4;

static short TwoDraw=0;

if(mouse.CheckLeft())

if(!secDraw)

{

if(TwoDraw!=1)

Draw=true;

else

TwoDraw=2;

}

if(!mouse.KeepLeft()&&Draw)

{

Vert1=mouse.LDownPos;

Vert2=mouse.Locate;

Draw=false;

TwoDraw=1;

}

if(!mouse.KeepLeft()&&TwoDraw==2)

{

Vert3=mouse.LDownPos;

Vert4=mouse.Locate;

gfKonys(Vector3(Vert1.x,(Vert1.y+Vert3.y)/2,Vert1.z),

Vert1.ZXDistance(Vert2),Vert3.ZXDistance(Vert4),

64,360.0f,fabs((Vert3.y-Vert1.y)/2.0f),&sc);

Draw=false;

secDraw=false;

TwoDraw=0;

Vert1=Vert2=Vert3=Vert4=nil3;

}

break;

}

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

case GL\_CYLINDER :

{

static Vector3 Vert1,Vert2,Vert3,Vert4;

static short TwoDraw=0;

if(mouse.CheckLeft())

if(!secDraw)

{

if(TwoDraw!=1)

Draw=true;

else

TwoDraw=2;

}

if(!mouse.KeepLeft()&&Draw)

{

Vert1=mouse.LDownPos;

Vert2=mouse.Locate;

Draw=false;

TwoDraw=1;

}

if(!mouse.KeepLeft()&&TwoDraw==2)

{

Vert3=mouse.LDownPos;

Vert4=mouse.Locate;

gfCylinder(Vector3(Vert1.x,(Vert1.y+Vert3.y)/2,Vert1.z),

Vert1.ZXDistance(Vert2),64,360.0f,

fabs((Vert3.y-Vert1.y)/2.0f),&sc);

Draw=false;

secDraw=false;

TwoDraw=0;

Vert1=Vert2=Vert3=Vert4=nil3;

}

break;

}

case GL\_ROTATE\_OBJ :

{

if(mouse.CheckLeft())

if(!Draw)

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New=new Spline(1);

New->Indexs[0]=sc.VertCount-1;

sc.AddEnd(New);

Draw=true;

}

else

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New->AddIndex(sc.VertCount-1);

}

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

if(mouse.CheckRight())

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New->AddIndex(sc.VertCount-1);

New->CreateRotateShape(20);

Draw=false;

sc.RemoveShape(New);

New=NULL;

}

break;

}

case GL\_VECTOR\_OBJ :

{

static Shape\* Spl;

if(mouse.CheckLeft())

if(!Draw)

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New=new Spline(1);

New->Indexs[0]=sc.VertCount-1;

sc.AddEnd(New);

Draw=true;

}

else

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New->AddIndex(sc.VertCount-1);

}

if(mouse.CheckRight())

{

if(!secDraw)

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New->AddIndex(sc.VertCount-1);

Spl=New;

Draw=false;

secDraw=true;

}

else

{

v.v.Vert=mouse.Locate;

v.v.Color=Vector3(0.0f,0.0f,0.0f);

sc.AddVertex(&v);

New->AddIndex(sc.VertCount-1);

Draw=secDraw=false;

New->ShapeExtrude(Spl);

sc.RemoveShape(New);

New=NULL;

sc.RemoveShape(Spl);

Spl=NULL;

}

}

break;

}

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

case GL\_DISC :

{

static Vector3 Vert1,Vert2;

if(mouse.CheckLeft())

if(!secDraw)

Draw=true;

else

{

Vector3 v=Vert2-Vert1,v1=mouse.Locate-Vert1;

v.y=v1.y=0.0f;

gfDisc(Vert1,v1.Abs(),v.Abs(),64,360.0f,&sc);

Draw=false;

secDraw=false;

Vert1=Vert2=nil3;

}

if(!mouse.KeepLeft()&&Draw)

{

Vert1=mouse.LDownPos;

Vert2=mouse.Locate;

secDraw=true;

Draw=false;

}

break;

}

}

}

if (buffer[DIK\_S]&0x80)

{

cam.UDrotate(-0.8f);

}

if (buffer[DIK\_W]&0x80)

{

cam.UDrotate(0.8f);

}

if (buffer[DIK\_A]&0x80)

{

cam.LRrotate(-0.8f);

}

if (buffer[DIK\_D]&0x80)

{

cam.LRrotate(0.8f);

}

if (buffer[DIK\_Z]&0x80)

{

cam.CWrotate(1.0f);

}

if (buffer[DIK\_X]&0x80)

{

cam.CWrotate(-1.0f);

}

if (buffer[DIK\_R]&0x80)

{

cam.Go(0.25f);

}

if (buffer[DIK\_F]&0x80)

{

cam.Go(-0.25f);

}

if (buffer[DIK\_C]&0x80)

{

mouse.ObNull();

}

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

//Mouse

if(mouse.KeepLeft())

{

if(mode==GL\_CAMERA\_TRANSFORM)

{

cam.Translate(mouse.NDeltaLPos());

}

}

if(mouse.CheckLeft())

{

if(mode==GL\_VUD\_OBJ&&!vud)

{

vec1=mouse.Locate;

vud=true;

}

if(CheckMode()&&CheckRenderMode())

{

v.v.Vert=mouse.Locate;

sc.AddVertex(&v);

k++;

}

}

if(mouse.CheckRight())

{

if(mode==GL\_MOVE\_TO)

{

sc.MoveTo(mouse.Locate);

}

if((mode==GL\_ROTATE||mode==GL\_MOVE)&&!fone)

{

fone=true;

}

}

if(mouse.KeepRight())

{

if(fone)

{

if(mode==GL\_MOVE)

{

sc.Move(mouse.NDeltaRPos());

}

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

if(mode==GL\_ROTATE)

{

ang=mouse.NDeltaRPos()\*50.0f;

sc.Rotate(ang);

}

}

if(mode==GL\_CAMERA\_TRANSFORM)

{

cam.Rotate(mouse.NDeltaRPos()\*10.0f);

}

}

if(mode==GL\_SCALE&&d)

{

scale+=Vector3(d/5000.0f\*mouse.bAxis.x,

d/5000.0f\*mouse.bAxis.y,

d/5000.0f\*mouse.bAxis.z);

sc.Scale(scale);

scale=Vector3(1.0f,1.0f,1.0f);

}

if(mode==GL\_CAMERA\_TRANSFORM&&!mouse.KeepLeft()&&!mouse.KeepRight())

{

cam.Go((float)d/60.0f);

}

mouse.Locate.y+=mouse.bAxis.y\*(float)(d/700.0f);

d=0;

}

if(!(buffer[DIK\_M]&0x80)&&pm)pm=false;

}

}

\_endthread();

}

Класи контейнера фігур ( ShapeContainer ) та самої фігури:

class ShapeContainer

{

public:

int VertCount,FigureCount;

GLuint VB;

VERTEX \*Vlist;

VATTRIB \*Valist;

Vector3 ActiveColor;

//Vector3 MaxPos,MinPos;

TexObject \*TexCont,\*TexIter,\*ActiveTex;

char Nt;

Shape \*list,\*iterator,\*Last;

char ToReWrite;

GLuint Name;

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

ShapeContainer(Shape\*);

ShapeContainer(void);

inline Shape\* Find(GLuint);

Shape Copy(Shape);

bool Init(void);

inline void AddAfter(Shape\*,GLuint );

inline void AddAfter(Shape\*,Shape\*after=NULL);

inline void AddFront(Shape \*);

inline void AddEndWr(Shape \*);

inline void AddEnd(Shape \*);

inline void Pop(void);

inline void Push(void);

inline void Activate(GLuint,bool IsAdd=false);

inline void Deactivate(GLuint);

inline void RemoveShape(Shape\*);

inline void RemoveFront(void);

inline void RemoveEnd(void);

inline void RemoveAll(void);

inline void Remove(void);

inline void Move(Vector3,GLuint);

inline void MoveTo(Vector3,GLuint);

inline void Rotate(Vector3,GLuint);

inline void Scale(Vector3,GLuint);

inline bool WriteToFile(AnsiString);

inline bool ReadFromFile(AnsiString);

inline Shape\* GetFirst(void);

inline int Size();

void DrawFigures(void);

void DrawFiguresForSelect(void);

inline void AddVertex(VERT\*,int n=1);

inline void ReloadArray();

inline void SubReloadArray();

inline void CheckFigures(GLCamera\*,Vector3,Vector3);

inline void CheckVertexs(GLCamera\*,Vector3,Vector3,bool Add=false);

inline void CheckAll(void);

inline void Hide(void);

inline void HideAll(void);

inline void UnHide(void);

void SetBuffer(GLuint buffer);

void AddTexture(TexObject\*);

void FlipNormals(void);

bool CheckError(void);

~ShapeContainer();

};

class Shape

{

public:

int NumOfVerts;

INDEX \*Indexs;

bool IsActive,IsVisible;

char mode;

GLuint name;

Shape \*next;

static ShapeContainer \*s;

Shape(int);

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

Shape(GLuint,int);

Vector3 FindCentre(void);

void AddIndex(INDEX);

~Shape(void);

void Hide(void);

void UnHide(void);

virtual void Draw(void)=0;

virtual void DrawForSelect(void)=0;

virtual bool WriteToFile(ofstream &h);

virtual bool ReadFromFile(ifstream &h);

virtual void Active(void)=0;

virtual void CalculateNormal(void);

virtual bool Check(/\*Matrix4\*,Vector3,Vector3\*/);

virtual void CreateRotateShape(int);

virtual void ShapeExtrude(Shape\*);

virtual void InvertNormal();

virtual void SetTexInd(GLuint ti);

};

Клас обробки миші:

class GLMouse

{

LPDIRECTINPUTDEVICE g\_pMouse;

public:

TPoint Pos;

Vector3 Locate;

Vector3 normal;

Vector3 bAxis;

Vector3 LDownPos,RDownPos;

MouseState ms;

//int WheelPos;

bool LBDown,RBDown,

LBPress,RBPress,Is3D,CursVis;

inline bool Init(LPDIRECTINPUT,HWND);

inline void Show(void);

inline void Hide(void);

inline void SetPos(int X,int Y);

inline void Set3DMode(void);

inline void Set2DMode(void);

inline bool CheckLeft(void);

inline bool CheckRight(void);

inline bool KeepLeft();

inline bool KeepRight();

inline void CursorDraw(void);

inline void ObNull(void);

inline Vector3 DeltaLPos(void);

inline Vector3 DeltaRPos(void);

inline Vector3 NDeltaLPos(void);

inline Vector3 NDeltaRPos(void);

inline bool Update(void);

void ClampAxis(Vector3);

};

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

**Додаток 2**

**Графічний матеріал**

UA.IK.32070-01 01

7

Лист

Дата

Підп.

№ докум.

Лист

змн

**Додаток 3**

**CD-диск з програмою**