**Пример №5**

Задача: построение простых тел.

**Этапное построение эскиза.**

1. Создаем с помощью функции MbSphereSurface сферу, для этого мы задаем координаты центра сферы в пространстве и радиус.

MbSphereSurface\* pSurf = new MbSphereSurface(MbCartPoint3D(50, 0, 0), 20);

1. После делаем отображение сферы в пространстве.

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf);

1. Создаем цилиндрическую поверхность. Для этого необходимо задать центр локальной оси, после задать два вектора по оси Z и Y.

// Цилиндрическая поверхность - построение по трем точкам

const MbCartPoint3D p0(50, 0, 0); // Центр локальной СК цилиндра (центр основания)

const MbCartPoint3D p1(0, 20, 0); // Вектор p0-p1 задает ось Z локальной СК

// и высоту цилиндра

const MbCartPoint3D p2(10, 0, 0); // Вектор p0-p2 задает ось X локальной СК и

// радиус основания

1. Следующий этап вызов функции MbCylinderSurface для построения поверхности цилиндра и отображаем его в пространстве.

MbCylinderSurface\* pSurf1 = new MbCylinderSurface(p0, p1, p2);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf1);

1. Следующее строим коническую поверхность. На понадобится локальная СК, радиус основания конуса, высота конуса и Угол между осью Z локальной СК и боковой образующей.

// Коническая поверхность - конструктор по локальной системе координат,

// радиусу, углу и высоте.

const MbPlacement3D pl1; // Локальная СК совпадает с мировой

const double radius = 0; // Радиус в плоскости XY локальной СК

// (если 0 - строится коническая поверхность конуса,

// если >0 - строится поверхность усеченного конуса)

// Угол между осью Z локальной СК и боковой образующей

const double angle = 22.5 \* M\_PI / 180.0;

const double height = 10; // Высота конуса (вдоль оси Z)

1. И также вызываем функцию MbConeSurfaсе и делаем конус видимым в пространстве.

MbConeSurface\* pSurf2 = new MbConeSurface(pl1, radius, angle, height);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf2);

1. Делаем тородиальную поверхность.

// Тороидальная поверхность: конструктор по локальной СК и двум радиусам

// Ось Z локальной СК является осью вращательной симметрии торической поверхности

const MbPlacement3D pl2; // Локальная СК совпадает с мировой.

const double radius1 = 10; // Радиус центров трубки тора

const double radius2 = 3; // Радиус трубки тора

MbTorusSurface\* pSurf3 = new MbTorusSurface(pl2, radius1, radius2);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf3);

**Код программы**

void MakeUserCommand6()

{

// Сферическая поверхность - построение по центру и радиусу

MbSphereSurface\* pSurf = new MbSphereSurface(MbCartPoint3D(50, 0, 0), 20);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf);

// Уменьшение счетчика ссылок динамически созданных объектов ядра

::DeleteItem(pSurf);

// Цилиндрическая поверхность - построение по трем точкам

const MbCartPoint3D p0(50, 0, 0); // Центр локальной СК цилиндра (центр основания)

const MbCartPoint3D p1(0, 20, 0); // Вектор p0-p1 задает ось Z локальной СК

// и высоту цилиндра

const MbCartPoint3D p2(10, 0, 0); // Вектор p0-p2 задает ось X локальной СК и

// радиус основания

MbCylinderSurface\* pSurf1 = new MbCylinderSurface(p0, p1, p2);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf1);

// Уменьшение счетчика ссылок динамически созданных объектов ядра

::DeleteItem(pSurf1);

// Коническая поверхность - конструктор по локальной системе координат,

// радиусу, углу и высоте.

const MbPlacement3D pl1; // Локальная СК совпадает с мировой

const double radius = 0; // Радиус в плоскости XY локальной СК

// (если 0 - строится коническая поверхность конуса,

// если >0 - строится поверхность усеченного конуса)

// Угол между осью Z локальной СК и боковой образующей

const double angle = 22.5 \* M\_PI / 180.0;

const double height = 10; // Высота конуса (вдоль оси Z)

MbConeSurface\* pSurf2 = new MbConeSurface(pl1, radius, angle, height);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf2);

// Уменьшение счетчика ссылок динамически созданных объектов ядра

::DeleteItem(pSurf2);

// Тороидальная поверхность: конструктор по локальной СК и двум радиусам

// Ось Z локальной СК является осью вращательной симметрии торической поверхности

const MbPlacement3D pl2; // Локальная СК совпадает с мировой.

const double radius1 = 10; // Радиус центров трубки тора

const double radius2 = 3; // Радиус трубки тора

MbTorusSurface\* pSurf3 = new MbTorusSurface(pl2, radius1, radius2);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf3);

// Уменьшение счетчика ссылок динамически созданных объектов ядра

::DeleteItem(pSurf3);

}

**Список используемых функций**

CreateSketch – пользовательская функция для создание эскиза.

MbResultType::ExtrusionSOlid – функция создание модели выдавливания.

**Список используемых классов**

MbPolyline – класс для создания замкнутой кривой.

MbContour – класс для создания контура.

MbArc – класс дуги.

MbPlane – класс плоскости.

MbCartPoint3D – класс точки в трехмерном пространстве.

MbVector3D – класс вектора в трехмерном пространстве.

MbSweptData – класс данных образующей.

MbSNameMaker – класс для генерирования имен.

MbSolid – класс твердого тела.

MbResultType – класс для описания конечного результата ( в данном случае твердое тело выдавливания)