**Пример №2**

Задача: построение эскиза и выдавливание под углом (Усеченная пирамида).

**Этапное построение эскиза.**

1. Создаем функцию CreateSketch:

voidCreateSketch(RPArray<MbContour>&\_arrContours)

{

...

}

1. Затем описываем CreateSketch (Все функции, которые мы применили объяснены в комментариях).

// Создание массива точек квадрата, к которому в дальнейшем добавятся скругления.

// Размер массива - 8 точек для учета точек четырех сегментов скруглений.

SArray<MbCartPoint>arrPnts(8);

arrPnts.Add(MbCartPoint(0, 0));

arrPnts.Add(MbCartPoint(50, 0));

arrPnts.Add(MbCartPoint(50, 50));

arrPnts.Add(MbCartPoint(0, 50));

// Построение единой ломаной внешнего контура по точкам

MbPolyline\* pPolyline = newMbPolyline(arrPnts, true);

MbContour\* pContourPolyline = NULL;

// ЗаданиескруглениясиспользованиемфункцииFilletPolyContour

::FilletPolyContour(pPolyline, 5, false, arrPnts[4], pContourPolyline);

// Задание индексов точек, в которых будет задаваться скругление с учетом

// добавления новой точки при скруглении с использованием функции FilletTwoSegments

ptrdiff\_t idxSideRight1 = 0;

ptrdiff\_t idxSideRight2 = 2;

ptrdiff\_t idxSideRight3 = 4;

// Добавлениескруглений

pContourPolyline->FilletTwoSegments(idxSideRight1, 5);

pContourPolyline->FilletTwoSegments(idxSideRight2, 5);

pContourPolyline->FilletTwoSegments(idxSideRight3, 5);

\_arrContours.push\_back(pContourPolyline);

1. Переходим к описанию основной функции.

void MakeUserCommand1()

{

...

}

1. Первое что мы делам, создаем локальную систему координат.

// Локальная СК (по умолчанию совпадает с мировой СК)

MbPlacement3Dpl;

1. Далее, мы вызываем функцию, отвечающую за создание эскиза.

// Создание образующей для тела выдавливания

RPArray<MbContour>arrContours;

CreateSketch(arrContours);

1. После, мы отображаем данный эскиз.

// Отображение образующей (в плоскости XY глобальной СК)

for (inti = 0; i<arrContours.size(); i++)

viewManager->AddObject(Style(1, LIGHTRED), arrContours[i], &pl);

1. Ниже будет представлен код, отвечающий за построение твердого тела с помощью выдавливания.

// ПОСТРОЕНИЕ ТЕЛА ВЫДАВЛИВАНИЯ

// Образующая размещается на плоскости XY глобальной СК.

// Важное замечание: объект-плоскость должен создаваться динамически,

// поскольку он продолжает использоваться в объекте-твердом теле после

// выхода из данной функции.

MbPlane\* pPlaneXY = newMbPlane(MbCartPoint3D(0, 0, 0),

MbCartPoint3D(1, 0, 0),

MbCartPoint3D(0, 1, 0));

// Объект, хранящий параметры образующей

MbSweptDatasweptData(\*pPlaneXY, arrContours);

// Направляющий вектор для операции выдавливания

MbVector3Ddir(0, 0, -1);

// Параметры операции выдавливания, задающие свойства тела для построения в прямом

// и в обратном направлении вдоль (глубина выдавливания и уклон).

constdouble HEIGHT\_FORWARD = 60.0, HEIGHT\_BACKWARD = 0.0;

constdouble ANGLE\_FORWARD\_DEGREE = 15.0;

ExtrusionValuesextrusionParams(HEIGHT\_FORWARD, HEIGHT\_BACKWARD);

// Указание уклона для выдавливания в прямом направлении

extrusionParams.side1.rake = ANGLE\_FORWARD\_DEGREE \* M\_PI / 180.0;

// Именователи элементов модели твердого тела и контуров образующей

MbSNameMakeroperNames(1, MbSNameMaker::i\_SideNone, 0);

PArray<MbSNameMaker>cNames(0, 1, false);

// Вызов функции-утилиты для построения твердого тела выдавливания

MbSolid\* pSolid = NULL;

MbResultType res = ::ExtrusionSolid(sweptData, dir, NULL, NULL, false,

extrusionParams, operNames, cNames, pSolid);

// Отображениепостроенноготела

if (res == rt\_Success)

{

// Смещение тела по оси Y, чтобы при отображении оно не накладывалось на образующую

pSolid->Move(MbVector3D(0, 0, 0));

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

}

// Уменьшение счетчиков ссылок динамически созданных объектов ядра

::DeleteItem(pSolid);

**Кодпрограммы**

voidCreateSketch(RPArray<MbContour>&\_arrContours)

{

// Создание массива точек квадрата, к которому в дальнейшем добавятся скругления.

// Размер массива - 8 точек для учета точек четырех сегментов скруглений.

SArray<MbCartPoint>arrPnts(8);

arrPnts.Add(MbCartPoint(0, 0));

arrPnts.Add(MbCartPoint(50, 0));

arrPnts.Add(MbCartPoint(50, 50));

arrPnts.Add(MbCartPoint(0, 50));

// Построение единой ломаной внешнего контура по точкам

MbPolyline\* pPolyline = newMbPolyline(arrPnts, true);

MbContour\* pContourPolyline = NULL;

// ЗаданиескруглениясиспользованиемфункцииFilletPolyContour

::FilletPolyContour(pPolyline, 5, false, arrPnts[4], pContourPolyline);

// Задание индексов точек, в которых будет задаваться скругление с учетом

// добавления новой точки при скруглении с использованием функции FilletTwoSegments

ptrdiff\_t idxSideRight1 = 0;

ptrdiff\_t idxSideRight2 = 2;

ptrdiff\_t idxSideRight3 = 4;

// Добавлениескруглений

pContourPolyline->FilletTwoSegments(idxSideRight1, 5);

pContourPolyline->FilletTwoSegments(idxSideRight2, 5);

pContourPolyline->FilletTwoSegments(idxSideRight3, 5);

\_arrContours.push\_back(pContourPolyline);

}

void MakeUserCommand1()

{

// Локальная СК (по умолчанию совпадает с мировой СК)

MbPlacement3Dpl;

// Создание образующей для тела выдавливания

RPArray<MbContour>arrContours;

CreateSketch(arrContours);

// Отображение образующей (в плоскости XY глобальной СК)

for (inti = 0; i<arrContours.size(); i++)

viewManager->AddObject(Style(1, LIGHTRED), arrContours[i], &pl);

// ПОСТРОЕНИЕ ТЕЛА ВЫДАВЛИВАНИЯ

// Образующая размещается на плоскости XY глобальной СК.

// Важное замечание: объект-плоскость должен создаваться динамически,

// поскольку он продолжает использоваться в объекте-твердом теле после

// выхода из данной функции.

MbPlane\* pPlaneXY = newMbPlane(MbCartPoint3D(0, 0, 0),

MbCartPoint3D(1, 0, 0),

MbCartPoint3D(0, 1, 0));

// Объект, хранящий параметры образующей

MbSweptDatasweptData(\*pPlaneXY, arrContours);

// Направляющий вектор для операции выдавливания

MbVector3Ddir(0, 0, -1);

// Параметры операции выдавливания, задающие свойства тела для построения в прямом

// и в обратном направлении вдоль (глубина выдавливания и уклон).

constdouble HEIGHT\_FORWARD = 60.0, HEIGHT\_BACKWARD = 0.0;

constdouble ANGLE\_FORWARD\_DEGREE = 15.0;

ExtrusionValuesextrusionParams(HEIGHT\_FORWARD, HEIGHT\_BACKWARD);

// Указание уклона для выдавливания в прямом направлении

extrusionParams.side1.rake = ANGLE\_FORWARD\_DEGREE \* M\_PI / 180.0;

// Именователи элементов модели твердого тела и контуров образующей

MbSNameMakeroperNames(1, MbSNameMaker::i\_SideNone, 0);

PArray<MbSNameMaker>cNames(0, 1, false);

// Вызов функции-утилиты для построения твердого тела выдавливания

MbSolid\* pSolid = NULL;

MbResultType res = ::ExtrusionSolid(sweptData, dir, NULL, NULL, false,

extrusionParams, operNames, cNames, pSolid);

// Отображениепостроенноготела

if (res == rt\_Success)

{

// Смещение тела по оси Y, чтобы при отображении оно не накладывалось на образующую

pSolid->Move(MbVector3D(0, 0, 0));

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

}

// Уменьшение счетчиков ссылок динамически созданных объектов ядра

::DeleteItem(pSolid);

}

**Список используемых функций**