**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ** **ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Московский государственный политехнический университет»**

КАФЕДРА «СМАРТ ТЕХНОЛОГИИ»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине

**ПРОГРАММИРОВАНИЕ В САПР**

Задание № 12

**РАЗРАБОТКА ИНТЕРФЕЙСА ПАРАМЕТРИЧЕСКОГО 3D-МОДЕЛИРОВАНИЯ АППАРАТА С ЭЛЛЕПТИЧЕСКИМ ДНИЩЕМ И СЪЕМНОЙ КРЫШКОЙ**

Студент: Константинова К.А.

Группа: 201-324

Преподаватель: Джунковский А.В.

Москва ‒ 2022

**ЗАДАНИЕ**

Разработка программного обеспечения для параметрической массо-габаритной 3D-модели плунжерного смесителя типа Пж-250 на языке высокого уровня C++ в среде Visual Studio 2019 под Autodesk Inventor 2022. Сборочный чертеж представлен на рисунке 1.

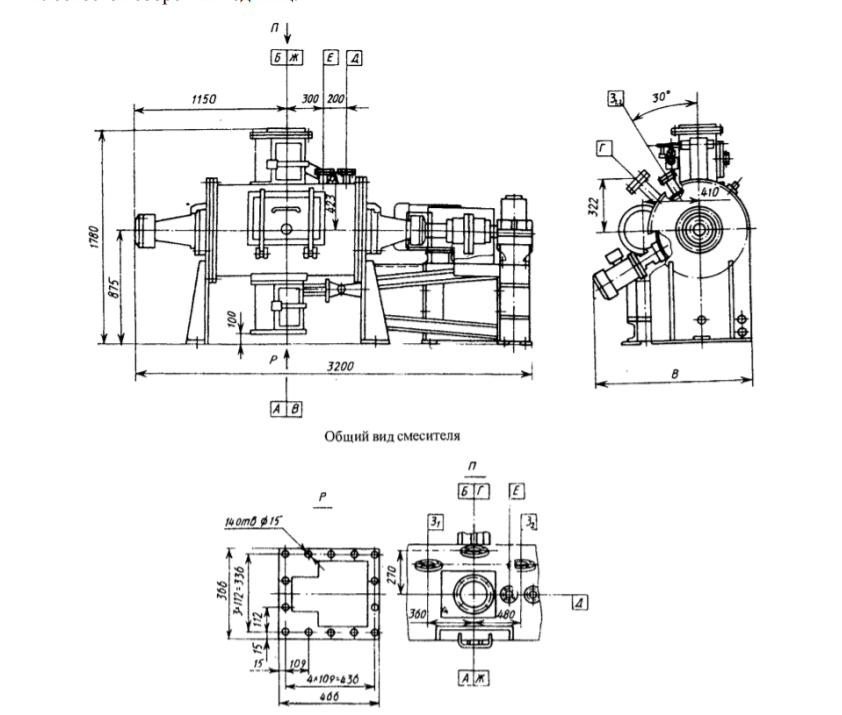


Рисунок 1 – Сборочный чертеж Плунжерного смесителя типа ПЖ-250

**СОДЕРЖАНИЕ**

**Введение**……………………………………………………………………………..4

1. **Проектирование интерфейса**………………………………………….…...5
   1. Начальный экран.……………………………………………………….……5
   2. Составляющие интерфейса……………………………………..……….…..5
   3. Описание диалогового окна…………………………………………………6
2. **Работа программы**…………………………………………………………..8
   1. Разделение экрана на две части……………………………………………..8
   2. Создание второго вида…………………………………………………….…9
   3. Заполнение древовидного списка………………………………….……....10
   4. Связь древовидного списка и документа………….………………………10
   5. Добавление ресурсов и их реализация ………..…………………………..12
   6. Создание диалогового окна…………………………………………….…..13
   7. Построение сборки…………………………………………………………16
      1. Подключение к API Inventor……………………………….……...…16
      2. Объявление переменных и глобальных указателей………………...17
      3. Проверка вводимых в диалоговом окне значений………………….17
      4. Запуск Inventor………………………………………………………...18
      5. Построение деталей и сборки…………………………………….….18
3. **Результат программы**…………………………………………...…………19

**Заключение**………………………………………………………...………………21

1. **Список использованной литературы**………………..…………..………22

**Приложение (Листинг программного кода)**…………………………………...23

**ВВЕДЕНИЕ**

Целью проекта является разработка подсистемы параметрического 3D-моделирования Плунжерного смесителя типа ПЖ-250на языке С++ в среде Visual Studio 2019 под AutoDesk Inventor 2022.

Смесители широко используются при строительстве, и нередки случаи, когда перепроектировать смеситель для данного конкретного производства, чтобы его габариты соответствовали требованиям заказчика. Данная программа значительно упрощает и ускоряет процесс получения примерной модели смесителя, позволяя пользователю выбрать желаемые параметры.

В теоретической части уточняются выбранные для работы инструменты, в практической части описывается подробная реализация интерфейса и сам процесс проектирования. В заключении можно видеть результат работы программы.

В приложении доступна часть исходного кода, которого будет достаточно, чтобы восстановить работающую программу в случае утери или повреждения оригинала.

1. **ПРОЕКТИРОВАНИЕ ИНТЕРФЕЙСА**
   1. Начальный экран

Для работы над заданным проектом была выбрана среда Visual Studio 2019. В качестве языка программирования используется язык высокого уровня C++. Вся графическая часть выполнялась в приложении на основе одного документа SDI application. На рисунке 1.1 представлен начальный экран, который выводится после запуска программы.

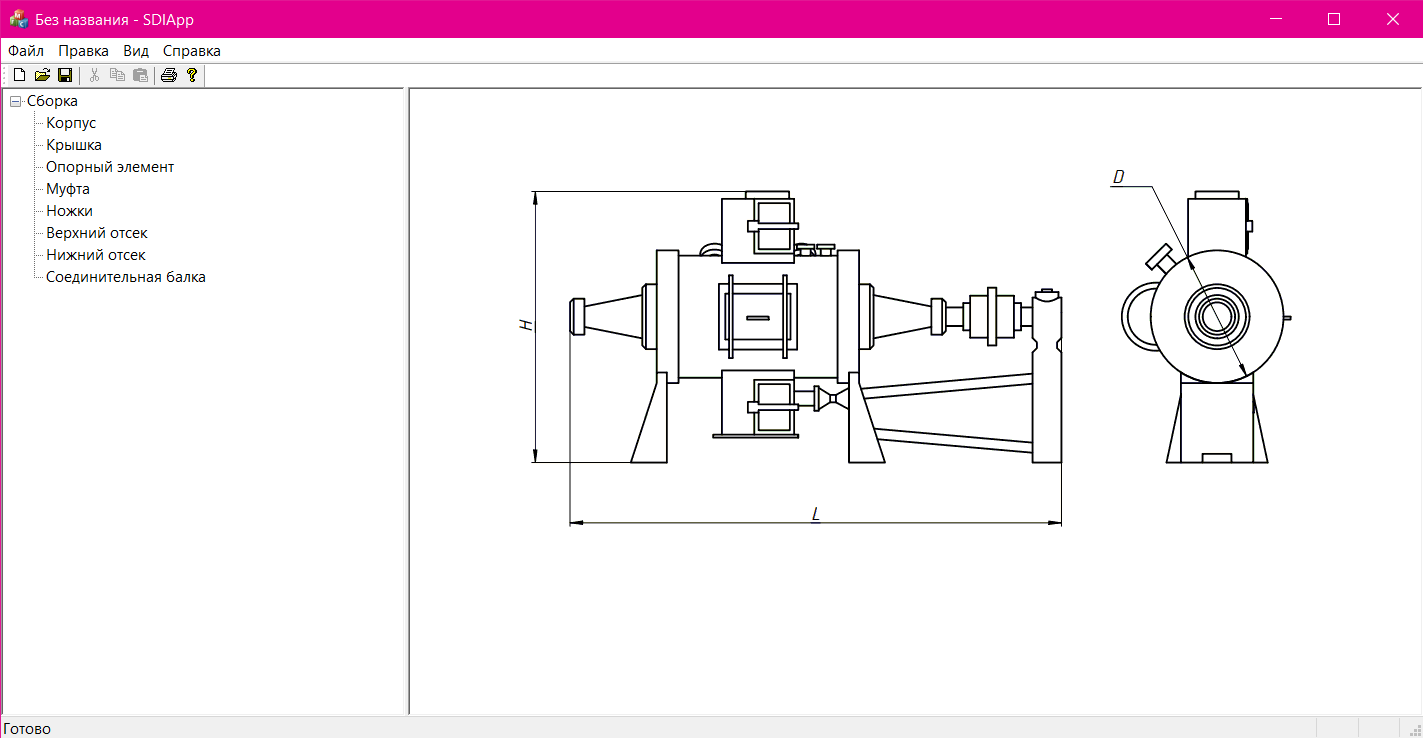


Рис.1.1. Начальный экран

* 1. Составляющие интерфейса

Интерфейс состоит из древовидного списка с наименованиями деталей, который расположен в левой части экрана, и чертежами деталей и сборки, в правой части экрана. При однократном нажатии на строку древовидного списка откроется чертеж детали или сборки (рис. 1.2), при двукратном нажатии на корневой элемент древовидного списка откроется диалоговое окно, представленное на рисунке 1.3, которое можно перетаскивать для удобства.

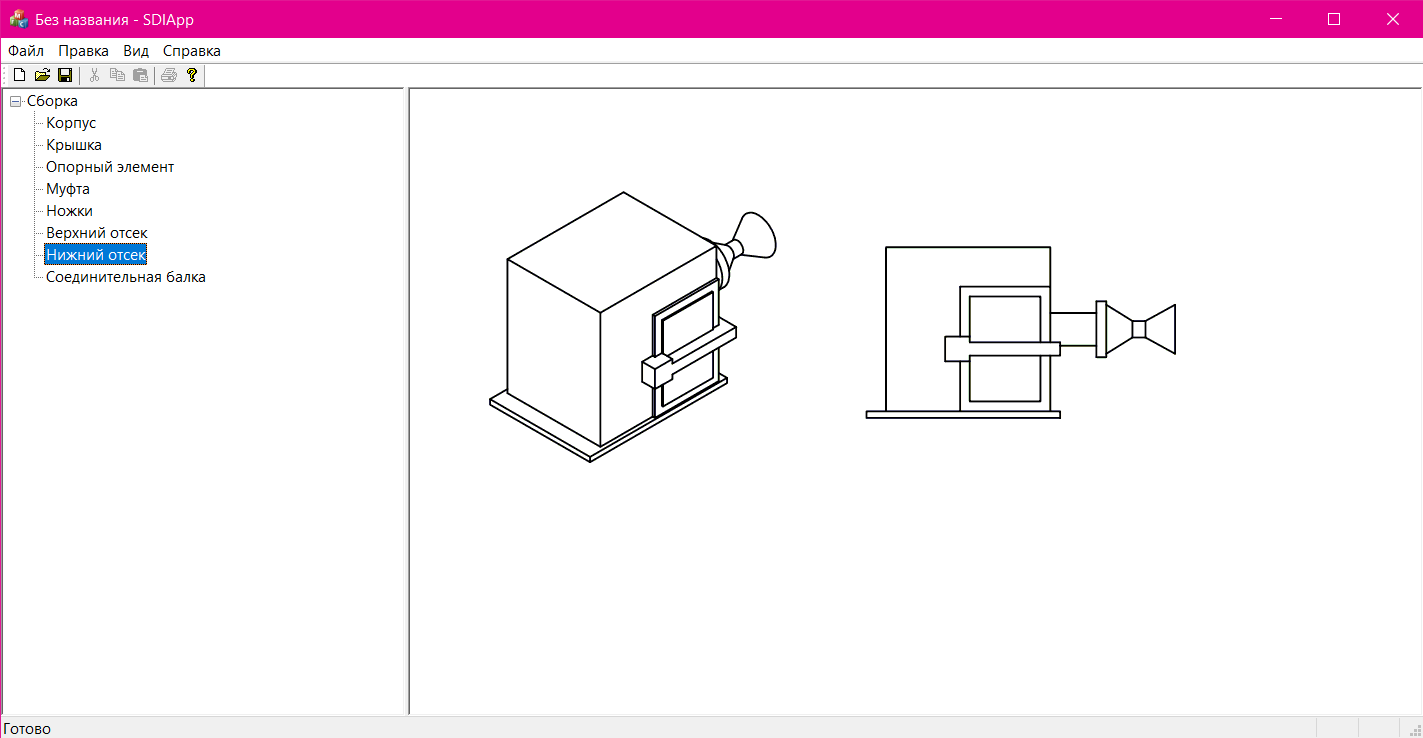


Рис. 1.2. Работа древовидного списка

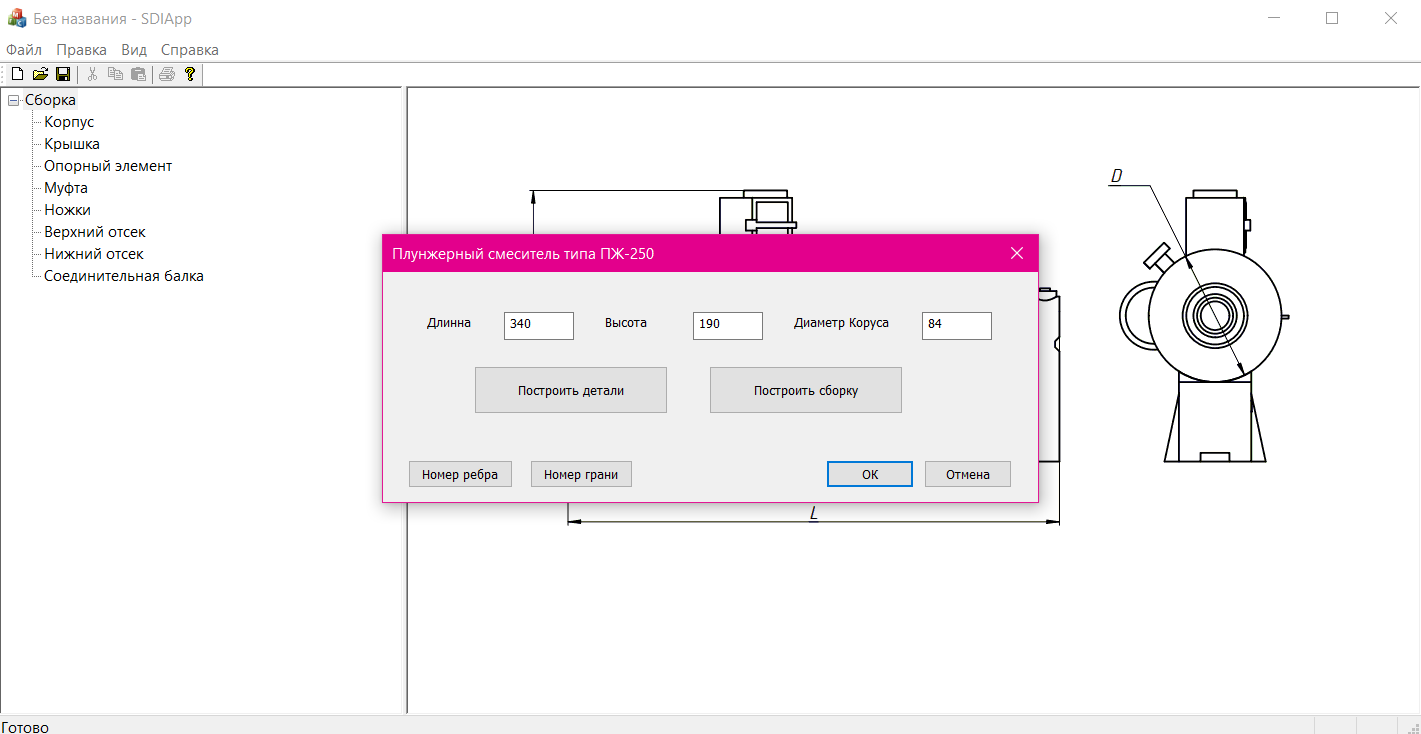


Рис. 1.3 Диалоговое окно

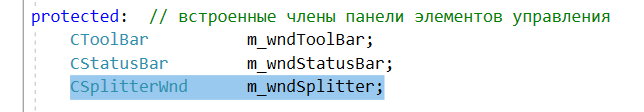
* 1. Описание диалогового окна

В диалоговом окне находятся поля ввода двух габаритных размеров и одного параметра корпуса для параметризации. Также в диалоговом окне находятся две кнопки: «Номер ребра» и «Номер грани». Они созданы для упрощения поиска номеров граней и ребер у деталей при их моделировании и последующей сборке.

После нажатия на кнопку «Построить детали» произойдет подключение к Inventor, построение всех деталей и их сохранение на устройство. После нажатия на кнопку «Построить сборку» программа вновь подключится к Inventor и соберет сборку из только что смоделированных деталей или последних сохраненных в папке.

1. **РАБОТА ПРОГРАММЫ**
   1. Разделение экрана на две части

В «Обозревателе решений» перейдем в файл «MainFrm.h». В области «protected», класса CMainFrame, добавим экземпляр MFC класса «разделяемое окно» (Листинг. 2.1).



Листинг. 2.1. Объявление «сплиттера»

Чтобы происходил процесс разделения клиентской области, реализуем функцию «OnCreateClient» для класса CMainFrame. Перейдем в «Мастер классов». Выберем класс «CMainFrame» и в разделе «Виртуальные функции» найдем функцию «OnCreateClient» (рис. 2.1).

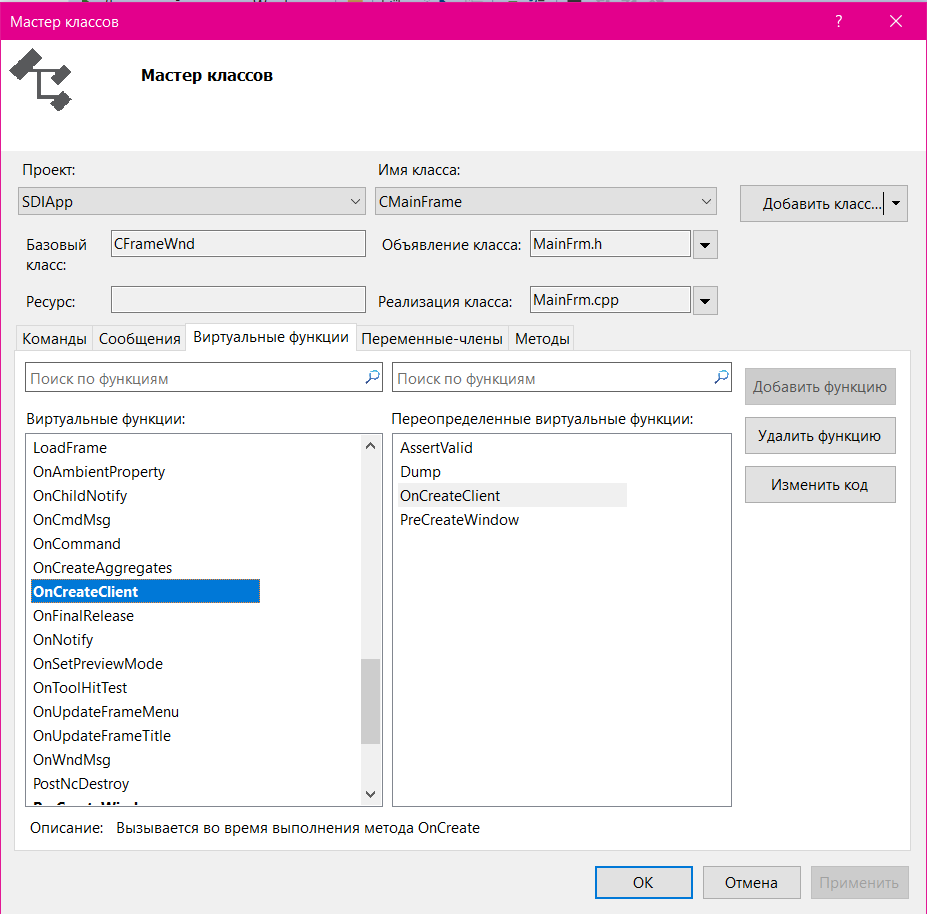
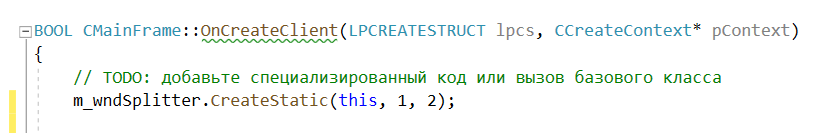


Рис. 2.1. Мастер классов

Добавим в созданную функцию код создания «сплиттера». В нем можно заметить, что мы делим нашу область на две колонки и один ряд (Листинг. 2.2).



Листинг. 2.2. Программный код создания «сплиттера»

* 1. Создание второго вида

Теперь создадим древовидный список, который будет отображаться в левой части экрана. Воспользуемся «Мастером классов». В выпадающем списке «Добавить класс» выберем «Класс MFC». В появившемся окне в поле «Имя класса» напишем «CMyView» и нажмем «ОК» (рис. 2.2) .

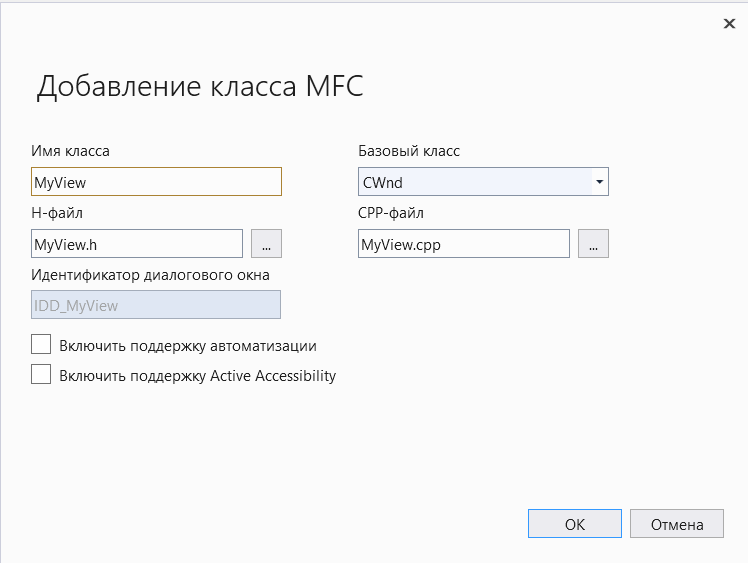
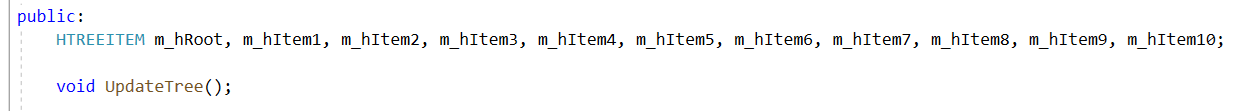


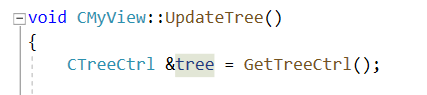
Рис. 2.2. Добавление класса «CMyView»

Наш древовидный список уже существует, но в нем пока нет элементов. Добавим метод «UpdateTree» в наш созданный класс. Также, каждый элемент древовидного списка описывается типом HTREEITEM. В заголовочном файле класса «MyView» добавим код (Листинг. 2.3).



Листинг. 2.3. Объявление элементов и метода

В файле реализации класса «MyView» разместим код метода «UpdateTree». Теперь получим доступ к контролу, который зашит в класс вида. В данном случае это объект типа «CTreeCtrl» (Листинг. 2.4).



Листинг. 2.4. Доступ к контролу

* 1. Заполнение древовидного списка

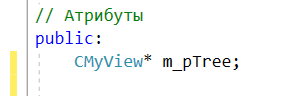
Теперь, в методе «UpdateTree» добавим корневой элемент «Сборка» и дочерние элементы с названиями деталей. (Листинг. 2.5).



Листинг. 2.5. Добавление корневого элемента

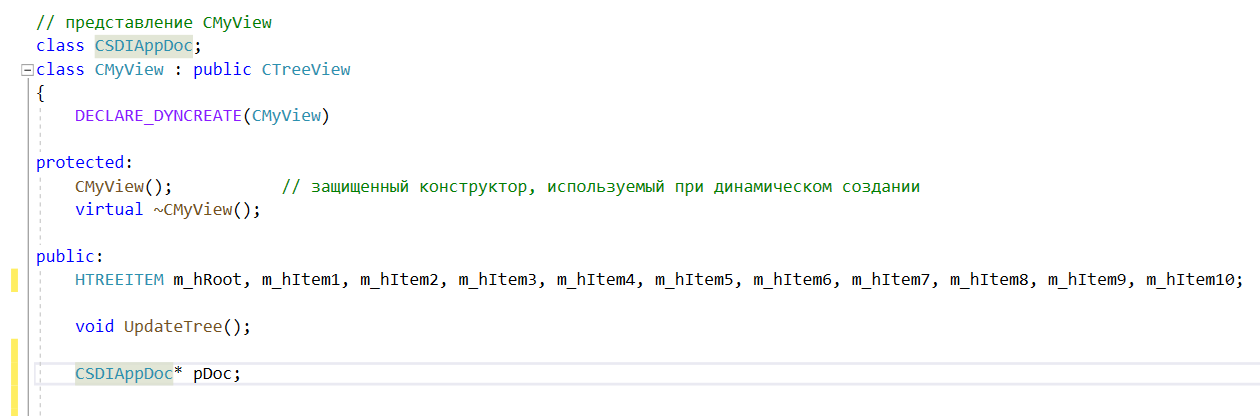
* 1. Связь древовидного списка и документа.

На данный момент ни документ, ни основной его вид понятия не имеют, что у нас в программе появился дополнительный вид документа. Очевидно, что нужно установить связи между этими объектами, дабы они могли управлять друг другом и получать нужную информацию друг от друга. Наиболее удобным методом «установления связей» является передача указателей на соответствующие объекты. Итак, добавим в заголовочный файл класса документа указатель на дерево (Листинг. 2.6).



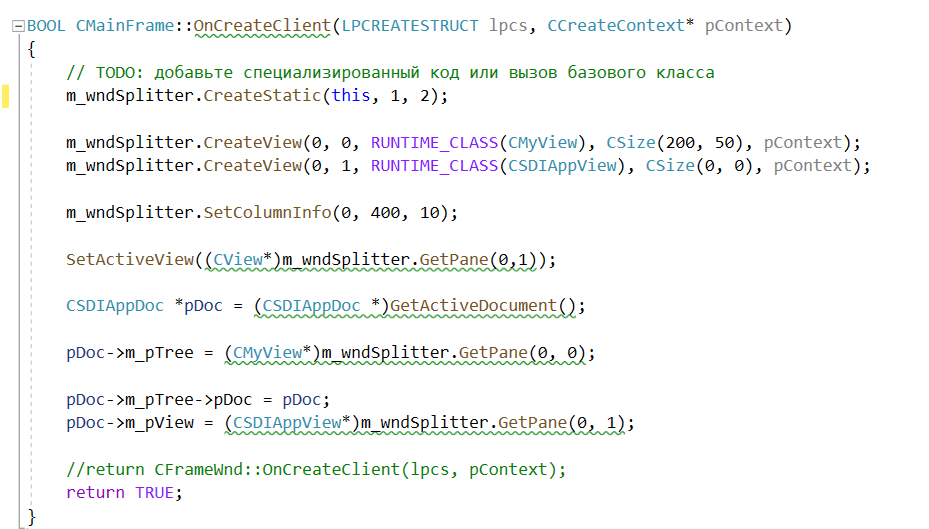
Листинг. 2.6. Добавление указателя на дерево

В заголовочном файле документа применим *forward declaration* к классу *CMyTreeView*. Аналогично сделаем указатель на документ и *forward declaration* его класса для класса (Листинг.2.7).



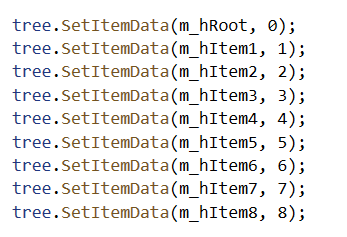
Листинг. 2.7. Указатель на документ

Осталось инициализировать заведенные указатели. Вернемся в метод «*OnCreateClient»* класса «*CMainFrame»* и попробуем инициализировать указатели с использованием информации, хранящейся в сплиттере (Листинг. 2.8).



Листинг. 2.8. Инициализация заведенных указателей

В методе «UpdateTree» функции «CMyView» пропишем код для добавления id-номера для каждой строки древовидного списка (Листинг 2.9).



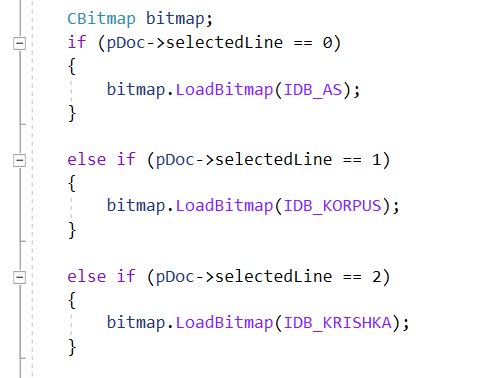
Листинг. 2.9. Добавление id-номера

* 1. Добавление ресурсов и их реализация

Теперь добавим чертежи в правую часть экрана по нажатию на строку дерева. Зайдем в окно ресурсов, нажмем правой кнопкой мыши по названию нашей программы в списке и выберем «Добавить ресурс». Чертежи нужно сохранять на устройстве в формате .bmp.

Нажимаем на вкладку «Bitmap» после на «Импорт», выберем все нужные изображения. После добавления изображений в «Окно ресурсов» должна была добавиться папка «Bitmap», в которой хранятся все чертежи.

Добавим код, который присвоит каждому элементу древовидного списка свой чертеж (Листинг. 2.10).



Листинг. 2.10. Присваивание элементу списка изображение

* 1. Создание диалогового окна

Далее создадим диалоговое окно. Для этого в окне ресурсов нажимаем правой кнопкой на название проекта и на «Добавить Ресурс» – «Dialog» – «Создать». С помощью панели элементов добавим в окно такие элементы как Button, StaticText, EditControl. (рис. 2.3)

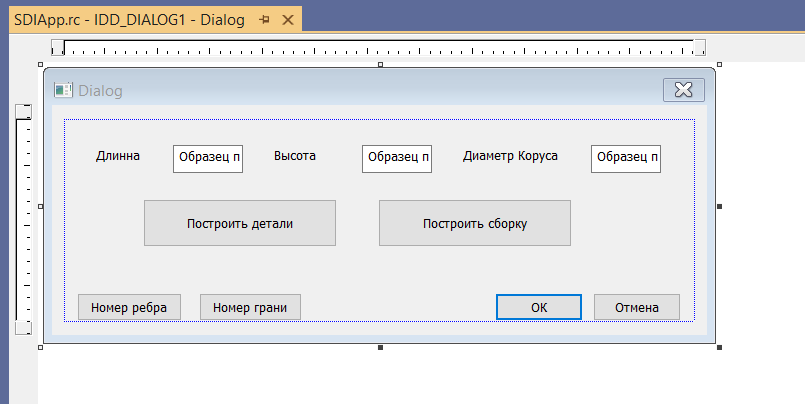
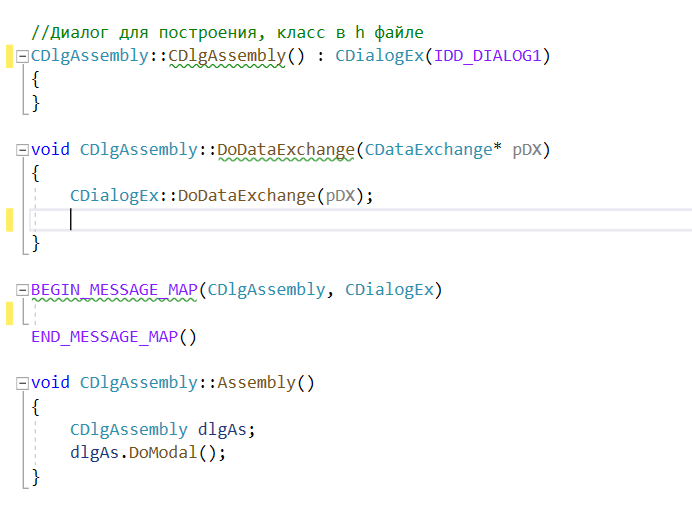
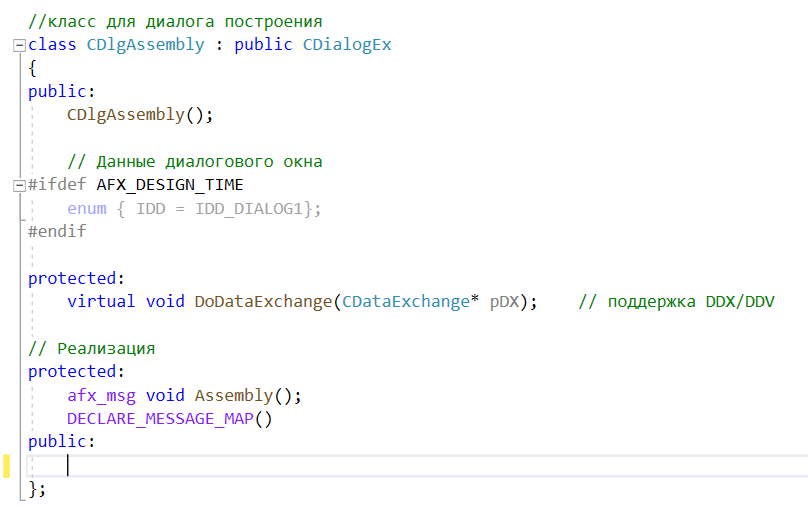


Рис. 2.3. Заполненное диалоговое окно

Для того чтобы окно работало должным образом, мы должны его объявить в SDIApp.cpp, как на листинге 2.11 и в SDIApp.h, как на листинге 2.11.



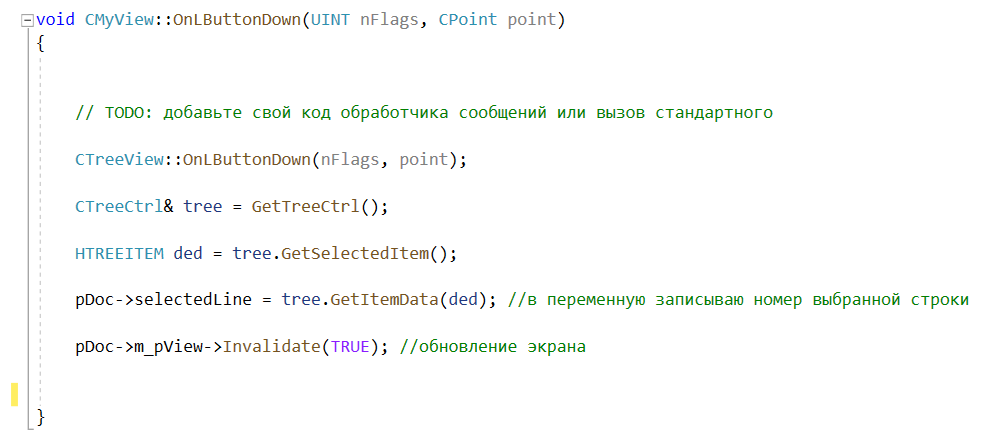
Листинг. 2.11. Объявление в .cpp файле



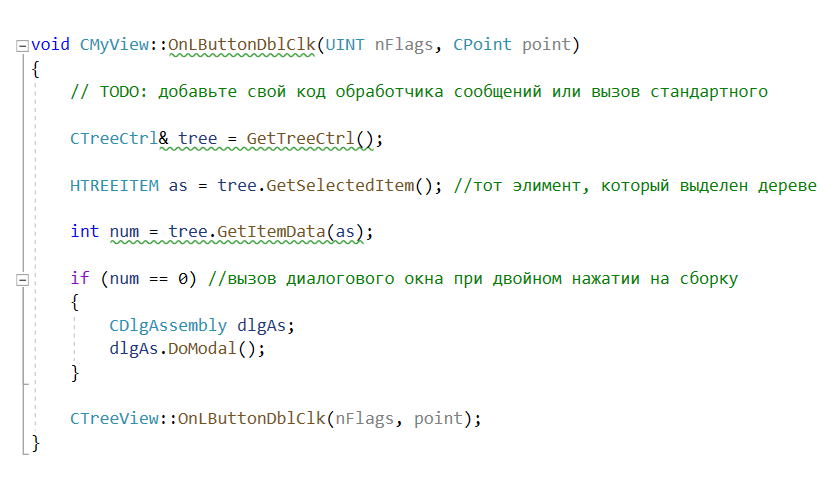
Листинг. 2.12. Объявление в заголовочном файле

Добавим две функции обработчика нажатий ЛКМ для древовидного списка.

Первая функция будет обрабатывать один клик мышки, после которого будет выводить чертеж детали. Вторая функция будет обрабатывать двойное нажатие на корневой элемент древовидного списка, после которых будет выводиться диалоговое окно для параметризации сборки (Листинг. 2.13 и Листинг. 2.14).

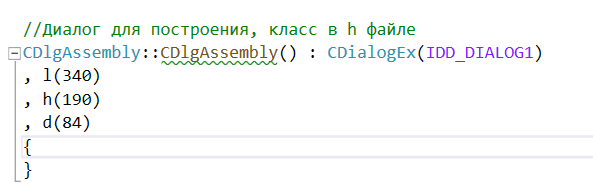


Листинг. 2.13. Функция обработки одного нажатия ЛКМ



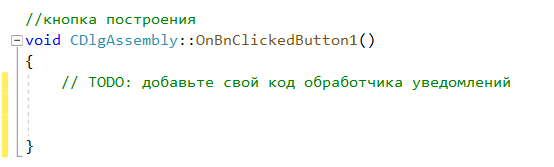
Листинг. 2.14. Функция обработки двойного нажатия ЛКМ

Добавим переменные в «EditControl», для считывания данных с экрана. Для этого нажмем ПКМ на «EditControl» – «Добавить переменную». Выбираем категорию, имя и тип. Категория всех переменных в нашем диалоговом окне будет «Значение», а тип «double». Так же зададим начальные параметры для всех «EditControl». Для этого переходим в SDIApp.cpp и проставляем значения переменных (Листинг. 2.15).



Листинг. 2.15. Добавление значений по умолчанию

Создадим обработчики события нажатия ЛКМ для кнопок в диалоговом окне. Для этого переходим в «Окно ресурсов» нажимаем на выпадающий список «Dialog», выбираем созданное диалоговое окно. Два раза нажимаем на кнопку и программа автоматически создает функцию обработки события. Создадим такие функции для кнопок «Построить детали», «Построить сборку», «Номер ребра» и «Номер грани» (Листинг. 2.16.).



Листинг. 2.16. Обработчик события нажатия на кнопку в диалоге

* 1. Построение сборки.
     1. Подключение к API Inventor и объявление переменных.

В начале, подключим библиотеку RxInventor.tlb, принадлежащую данной САПР с помощью директивы *import*. Уберём «пересечения» имен функций API Windows и API Inventor, для этого переименуем их с помощью rename (Листинг. 2.17.).



Листинг. 2.17. Подключение библиотеки API Inventor

* + 1. Объявление переменных и глобальных указателей.

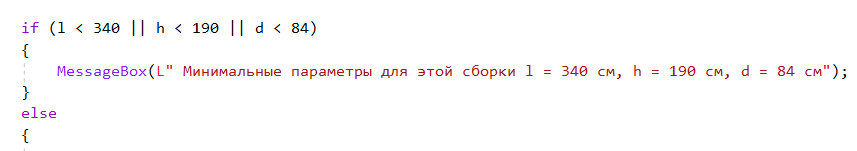
Объявим переменные деталей/сборки и перечислим необходимые глобальные указатели (Листинг. 2.18).



Листинг. 2.19. Подключение библиотеки API Inventor

* + 1. Проверка вводимых в диалоговом окне значений

Обозначим минимальные значения для данной сборки (Листинг. 2.20).



Листинг. 2.20. Проверка вводимых значений

* + 1. Запуск Inventor

Затем запустим Autodesk Inventor с помощью представленного ниже кода (Листинг. 2.21).



Листинг. 2.21. Запуск Autodesk Inventor

* + 1. Построение деталей и сборки.

Полный код построения сборки можно посмотреть в листинге программы в приложении.

1. **РЕЗУЛЬТАТЫ РАБОТЫ**

Запущенная программа после компиляции представленная на рисунке 3.1.

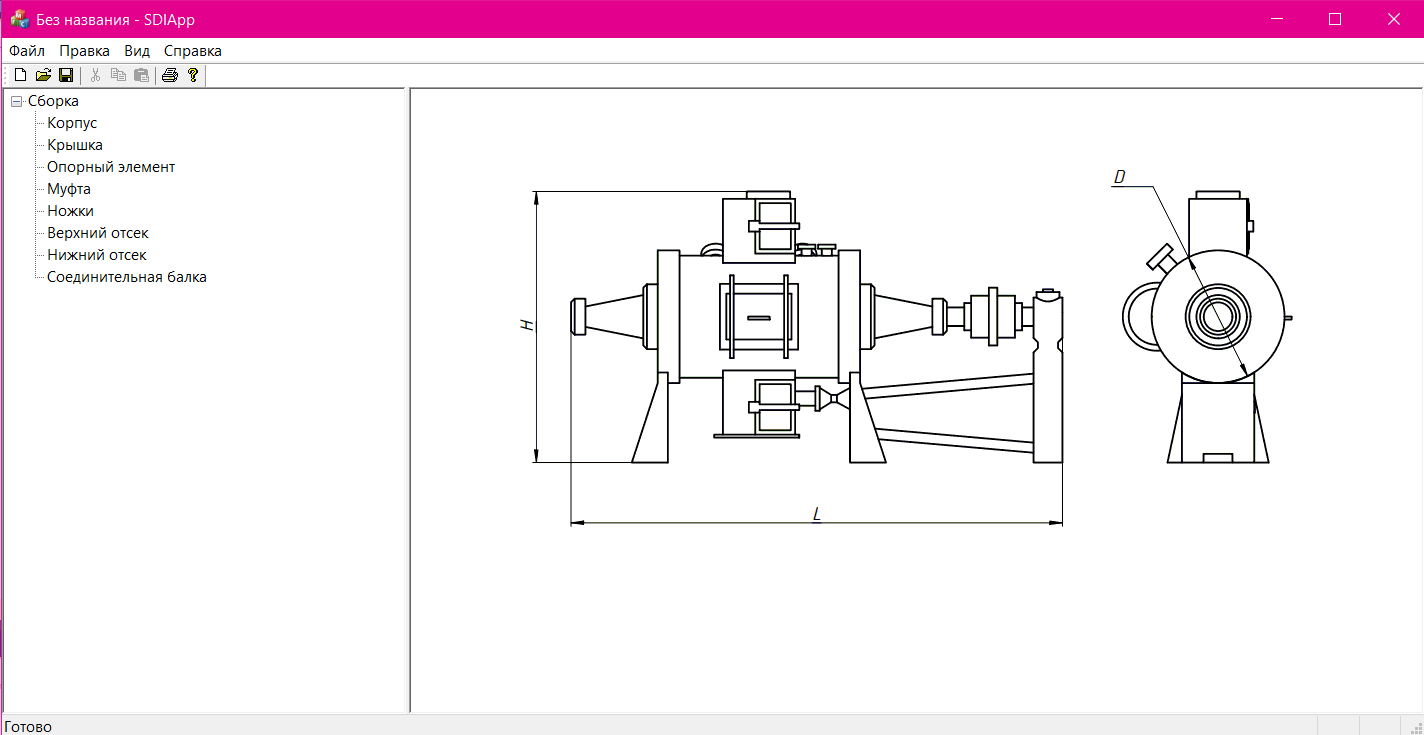


Рис. 3.1. Приложение после запуска

После двойного нажатия на корневой элемент древовидного списка, открывается диалоговое окно (рис. 3.2.).

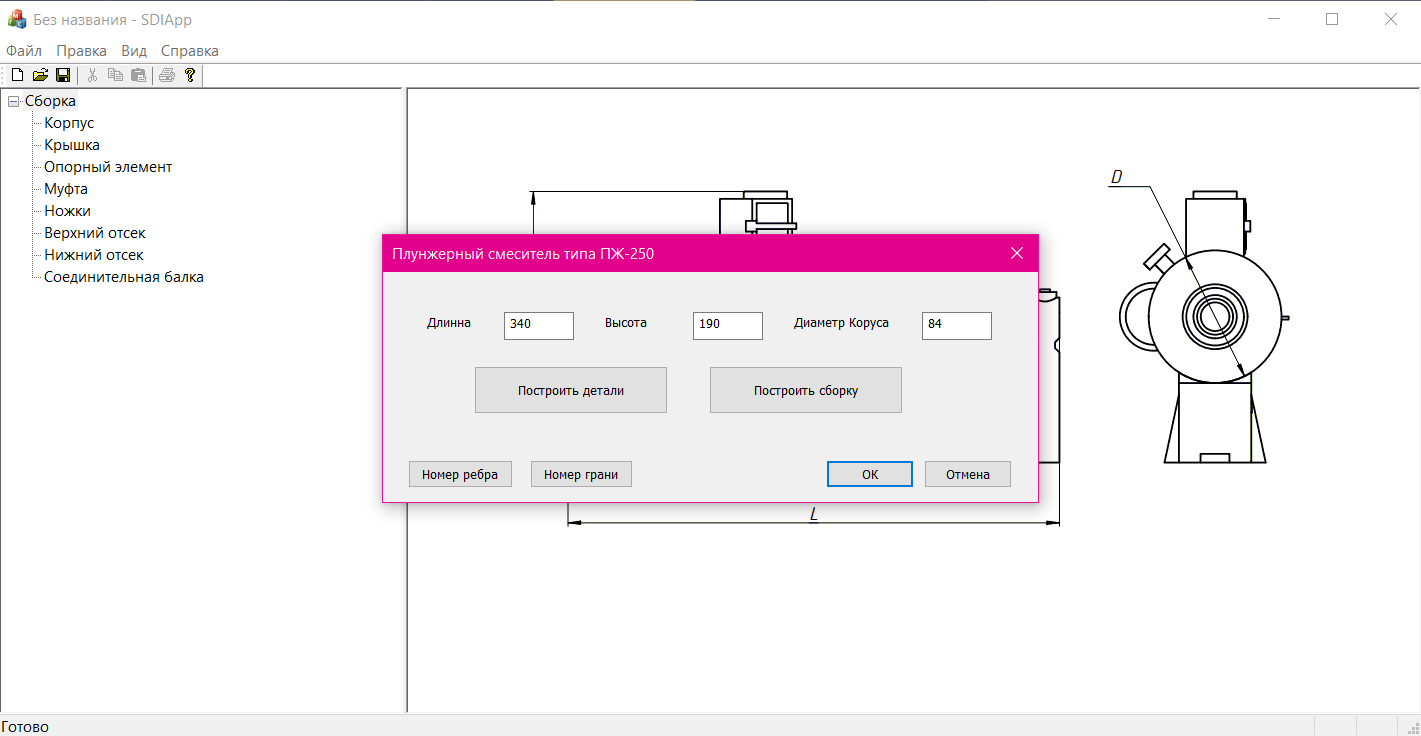


Рис. 3.2. Диалоговое окно

Нажмем на кнопку «Построить детали». Программа запускает Inventor и строит все детали сборки и сохраняет по заранее указанному пути (рис. 3.3).

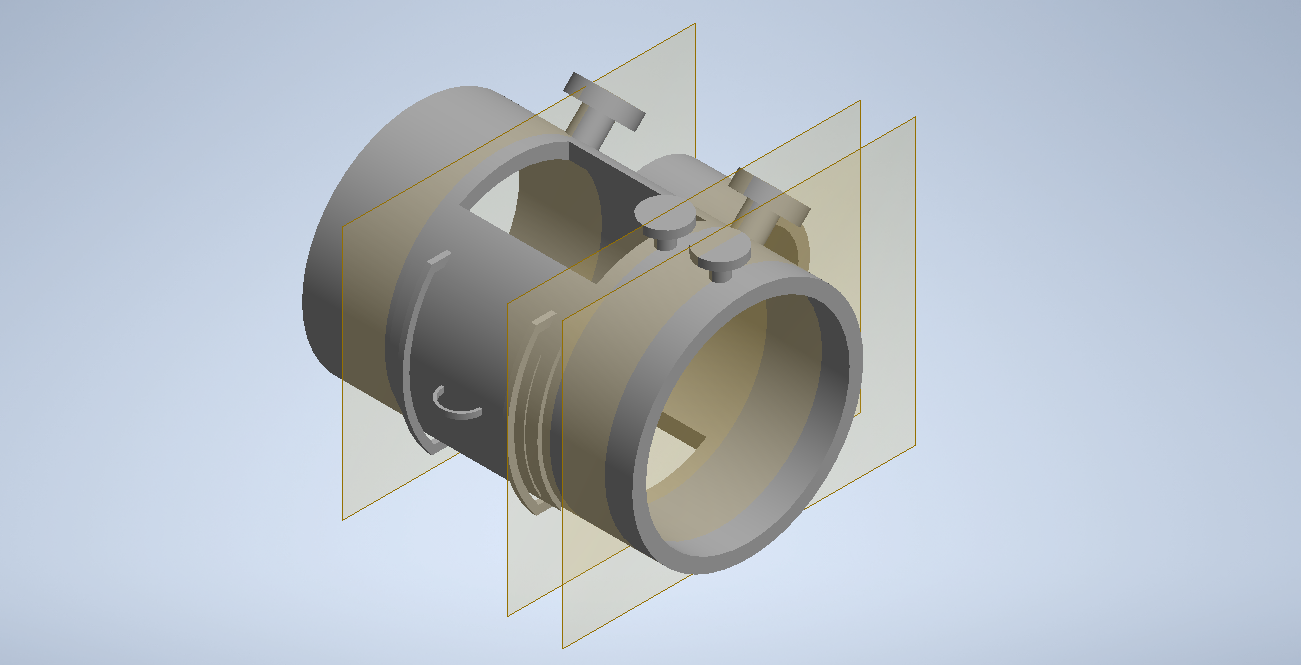


Рис. 3.2. Построение корпуса

После нажатия на кнопку «Построить сборку» строится Плунжерный смеситель (рис. 3.4).

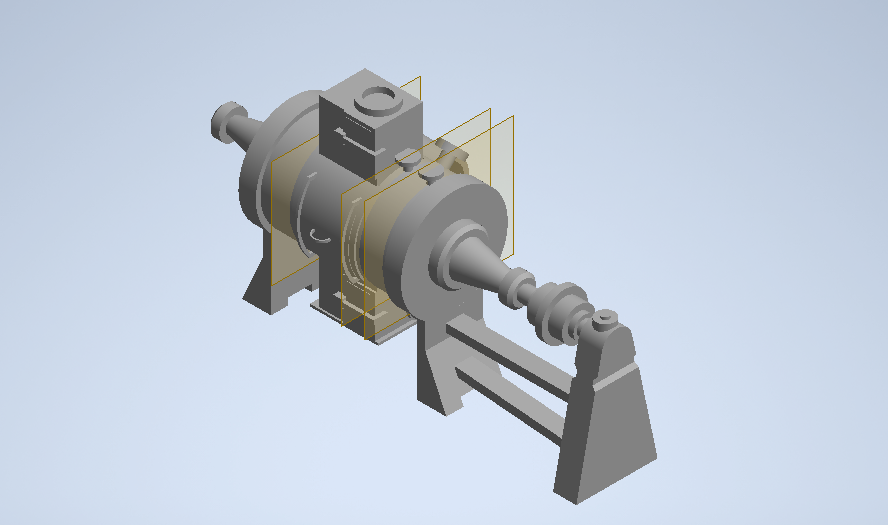


Рис. 3.3. Плунжерный смеситель типа ПЖ-250

**ЗАКЛЮЧЕНИЕ**

Была разработана подсистемы параметрического 3D-моделирования Плунжерного смесителя типа ПЖ-250на языке С++ в среде Visual Studio 2019 под Autodesk Inventor 2022.

Программа включает в себя окно, разделенное на две части. В части расположен древовидный список с названиями деталей сборки. По нажатию на каждую из строк списка в правой части окна появляется чертеж детали. По двойному нажатию на корневой элемент списка, открывается диалоговое окно с полями для ввода данных для параметризации, с кнопками для построения деталей и сборки и с кнопками для нахождения номеров ребер и граней конкретной детали.

1. **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. А.С. Тимонин. Основы конструирования и расчета химико-технологического оборудования и природоохранного оборудования. Справочник в 2 т. Т 2. — Калуга: Издательство Н. Бочкаревой, 2002. — с. 221

2. А.С. Тимонин. Основы конструирования и расчета химико-технологического оборудования и природоохранного оборудования. Справочник в 2 т. Т 1. — Калуга: Издательство Н. Бочкаревой, 2002. — с. 541 ÷ 569

1. **ПРИЛОЖЕНИЕ (ЛИСТИНГ ПРОГРАММНОГО КОДА)**

void CDlgAssembly::OnBnClickedButton1()

{ // TODO: добавьте свой код обработчика уведомлений

UpdateData(TRUE);

if (l < 340 || h < 190 || d < 84){

MessageBox(L" Минимальные параметры для этой сборки l = 340 см, h = 190 см, d = 84 см");

}

Else {

BeginWaitCursor();

// Get hold of the program id of Inventor.

CLSID InvAppClsid;

HRESULT hRes = CLSIDFromProgID(L"Inventor.Application", &InvAppClsid);

if (FAILED(hRes)) {

pInvApp = nullptr;

return;// ReturnAndShowCOMError(hRes, L"ConnectToInventor,CLSIDFromProgID failed");

}

// See if Inventor is already running...

CComPtr<IUnknown> pInvAppUnk = nullptr;

hRes = ::GetActiveObject(InvAppClsid, NULL, &pInvAppUnk);

if (FAILED(hRes)) {

// Inventor is not already running, so try to start it...

TRACE(L"Could not get hold of an active Inventor, will start a new session\n");

hRes = CoCreateInstance(InvAppClsid, NULL, CLSCTX\_LOCAL\_SERVER, \_\_uuidof(IUnknown), (void\*\*)&pInvAppUnk);

if (FAILED(hRes)) {

pInvApp = nullptr;

return;// ReturnAndShowCOMError(hRes, L"ConnectToInventor,CoCreateInstance failed");

}

}

// Get the pointer to the Inventor application...

if (pInvApp == NULL) {

hRes = pInvAppUnk->QueryInterface(\_\_uuidof(Application), (void\*\*)&pInvApp);

}

if (FAILED(hRes)) {

return;// ReturnAndShowCOMError(hRes, L"ConnectToInventor,QueryInterface failed");

}

pInvApp->put\_Visible(TRUE);

//КОРПУС

//1 эскиз

pPartDoc8 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc8->DisplayName = \_T("Корпус");

pPartDoc8->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc8);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches); pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp); pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax); pPartComDef->get\_WorkPoints(&wpt);

//5 эскиз

PlanarSketch\* pSketch31;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch31);

SketchPointPtr point31[9]; SketchLinePtr lines31[6]; SketchArcs\* skArcs31;

SketchPoints\* skPoints31; SketchLines\* skLines31;

Profiles\* skProfiles31;

SketchArc\* arc31[2];

pSketch31->get\_SketchPoints(&skPoints31); pSketch31->get\_SketchLines(&skLines31);

pSketch31->get\_Profiles(&skProfiles31); pSketch31->get\_SketchArcs(&skArcs31);

double y3 = sqrt(d \* d / 4 + 3 \* d + 3 \* 3 - 27 \* 27);

double y4 = sqrt(d \* d / 4 + 5.5 \* d + 5.5 \* 5.5 - 30 \* 30);

point31[1] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(27, y3), false);

point31[2] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(27, 25), false);

point31[3] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(30, 25), false);

point31[4] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(30, y4), false);

point31[5] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(-30, y4), false);

point31[6] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(-30, 25), false);

point31[7] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(-27, 25), false);

point31[8] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(-27, y3), false);

point31[0] = skPoints31->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

lines31[0] = skLines31->MethodAddByTwoPoints(point31[1], point31[2]);

lines31[1] = skLines31->MethodAddByTwoPoints(point31[2], point31[3]);

lines31[2] = skLines31->MethodAddByTwoPoints(point31[3], point31[4]);

lines31[3] = skLines31->MethodAddByTwoPoints(point31[5], point31[6]);

lines31[4] = skLines31->MethodAddByTwoPoints(point31[6], point31[7]);

lines31[5] = skLines31->MethodAddByTwoPoints(point31[7], point31[8]);

arc31[0] = skArcs31->MethodAddByCenterStartEndPoint(point31[0], point31[1], point31[8], true);

arc31[1] = skArcs31->MethodAddByCenterStartEndPoint(point31[0], point31[4], point31[5], true);

ProfilePtr pProfile31;

Try { pProfile31 = skProfiles31->MethodAddForSolid(true); }

catch (...)

{ AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude31;

ft->get\_ExtrudeFeatures(&ftExtrude31);

ExtrudeFeaturePtr extrude31 = ftExtrude31->MethodAddByDistanceExtent(pProfile31, 40.0f, kSymmetricExtentDirection, kJoinOperation);

//6 эскиз

PlanarSketch\* pSketch32;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch32);

SketchPointPtr point32[9]; SketchLinePtr lines32[6]; SketchArcs\* skArcs32;

SketchPoints\* skPoints32; SketchLines\* skLines32;

Profiles\* skProfiles32;

SketchArc\* arc32[2];

pSketch32->get\_SketchPoints(&skPoints32); pSketch32->get\_SketchLines(&skLines32);

pSketch32->get\_Profiles(&skProfiles32); pSketch32->get\_SketchArcs(&skArcs32);

point32[1] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(27, y3), false);

point32[2] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(27, 25), false);

point32[3] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(30, 25), false);

point32[4] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(30, y4), false);

point32[5] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(-30, y4), false);

point32[6] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(-30, 25), false);

point32[7] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(-27, 25), false);

point32[8] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(-27, y3), false);

point32[0] = skPoints32->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

lines32[0] = skLines32->MethodAddByTwoPoints(point32[1], point32[2]);

lines32[1] = skLines32->MethodAddByTwoPoints(point32[2], point32[3]);

lines32[2] = skLines32->MethodAddByTwoPoints(point32[3], point32[4]);

lines32[3] = skLines32->MethodAddByTwoPoints(point32[5], point32[6]);

lines32[4] = skLines32->MethodAddByTwoPoints(point32[6], point32[7]);

lines32[5] = skLines32->MethodAddByTwoPoints(point32[7], point32[8]);

arc32[0] = skArcs32->MethodAddByCenterStartEndPoint(point32[0], point32[1], point32[8], true);

arc32[1] = skArcs32->MethodAddByCenterStartEndPoint(point32[0], point32[4], point32[5], true);

ProfilePtr pProfile32;

try{ pProfile32 = skProfiles32->MethodAddForSolid(true);}

catch (...)

{ AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude32;

ft->get\_ExtrudeFeatures(&ftExtrude32);

ExtrudeFeaturePtr extrude32 = ftExtrude32->MethodAddByDistanceExtent(pProfile32, 36.0f, kSymmetricExtentDirection, kCutOperation);

//1 эскиз

PlanarSketch\* pSketch27;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch27);

SketchPointPtr point27; SketchCirclePtr circ27;

Profiles\* skProfiles27;

SketchCirclesPtr skCircles27; SketchPoints\* skPoints27;

pSketch27->get\_Profiles(&skProfiles27); pSketch27->get\_SketchCircles(&skCircles27);

pSketch27->get\_SketchPoints(&skPoints27);

point27 = skPoints27->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

circ27 = skCircles27->MethodAddByCenterRadius(point27, d / 2);

ProfilePtr pProfile27;

Try { pProfile27 = skProfiles27->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude27;

ft->get\_ExtrudeFeatures(&ftExtrude27);

ExtrudeFeaturePtr extrude27 = ftExtrude27->MethodAddByDistanceExtent(pProfile27, l - 220, kSymmetricExtentDirection, kJoinOperation);

//3 эскиз

PlanarSketch\* pSketch29;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch29); //номер грани

SketchPointPtr point29[3]; SketchLinePtr lines29; SketchArcsPtr skArcs29;

SketchPoints\* skPoints29; SketchLines\* skLines29;

Profiles\* skProfiles29; SketchArc\* arc29;

pSketch29->get\_SketchPoints(&skPoints29);

pSketch29->get\_SketchLines(&skLines29);

pSketch29->get\_Profiles(&skProfiles29);

pSketch29->get\_SketchArcs(&skArcs29);

double y = sqrt(d \* d / 4 + 1.5 \* d + 1.5 \* 1.5 - 26 \* 26);

point29[0] = skPoints29->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point29[1] = skPoints29->MethodAdd(pTransGeom->MethodCreatePoint2d(26, y), false);

point29[2] = skPoints29->MethodAdd(pTransGeom->MethodCreatePoint2d(-26, y), false);

lines29 = skLines29->MethodAddByTwoPoints(point29[1], point29[2]);

arc29 = skArcs29->MethodAddByCenterStartEndPoint(point29[0], point29[2], point29[1], false);

ProfilePtr pProfile29;

Try { pProfile29 = skProfiles29->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude29;

ft->get\_ExtrudeFeatures(&ftExtrude29);

ExtrudeFeaturePtr extrude29 = ftExtrude29->MethodAddByDistanceExtent(pProfile29, 54.0f, kSymmetricExtentDirection, kJoinOperation);

//4 эскиз

PlanarSketch\* pSketch30;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch30); //номер грани

SketchPointPtr point30[3]; SketchLinePtr lines30; SketchArcsPtr skArcs30;

SketchPoints\* skPoints30; SketchLines\* skLines30;

Profiles\* skProfiles30; SketchArc\* arc30;

pSketch30->get\_SketchPoints(&skPoints30);

pSketch30->get\_SketchLines(&skLines30);

pSketch30->get\_Profiles(&skProfiles30);

pSketch30->get\_SketchArcs(&skArcs30);

double y1 = sqrt(d \* d / 4 + 2 \* d + 4 - 23 \* 23);

point30[0] = skPoints30->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point30[1] = skPoints30->MethodAdd(pTransGeom->MethodCreatePoint2d(23, y1), false);

point30[2] = skPoints30->MethodAdd(pTransGeom->MethodCreatePoint2d(-23, y1), false);

lines30 = skLines30->MethodAddByTwoPoints(point30[1], point30[2]);

arc30 = skArcs30->MethodAddByCenterStartEndPoint(point30[0], point30[2], point30[1], false);

ProfilePtr pProfile30;

Try { pProfile30 = skProfiles30->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude30;

ft->get\_ExtrudeFeatures(&ftExtrude30);

ExtrudeFeaturePtr extrude30 = ftExtrude30->MethodAddByDistanceExtent(pProfile30, 45.0f, kSymmetricExtentDirection, kJoinOperation);

//7 эскиз

PlanarSketch\* pSketch33;

sketches->raw\_Add(wp->GetItem(2), false, &pSketch33);

SketchPointPtr point33[5]; SketchLinePtr lines33[2]; SketchArcs\* skArcs33;

SketchPoints\* skPoints33; SketchLines\* skLines33;

Profiles\* skProfiles33; SketchArc\* arc33[2];

pSketch33->get\_SketchPoints(&skPoints33);

pSketch33->get\_SketchLines(&skLines33);

pSketch33->get\_Profiles(&skProfiles33);

pSketch33->get\_SketchArcs(&skArcs33); double yy = d / 2 + 1;

point33[0] = skPoints33->MethodAdd(pTransGeom->MethodCreatePoint2d(0, yy + 1), false);

point33[1] = skPoints33->MethodAdd(pTransGeom->MethodCreatePoint2d(-7.4, yy), false);

point33[2] = skPoints33->MethodAdd(pTransGeom->MethodCreatePoint2d(-6.4, yy), false);

point33[3] = skPoints33->MethodAdd(pTransGeom->MethodCreatePoint2d(6.4, yy), false);

point33[4] = skPoints33->MethodAdd(pTransGeom->MethodCreatePoint2d(7.4, yy), false);

lines33[0] = skLines33->MethodAddByTwoPoints(point33[1], point33[2]);

lines33[1] = skLines33->MethodAddByTwoPoints(point33[4], point33[3]);

arc33[0] = skArcs33->MethodAddByCenterStartEndPoint(point33[0], point33[1], point33[4], false);

arc33[1] = skArcs33->MethodAddByCenterStartEndPoint(point33[0], point33[2], point33[3], false);

ProfilePtr pProfile33;

try{ pProfile33 = skProfiles33->MethodAddForSolid(true); }

catch (...){ AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude33;

ft->get\_ExtrudeFeatures(&ftExtrude33);

ExtrudeFeaturePtr extrude33 = ftExtrude33->MethodAddByDistanceExtent(pProfile33, 2.0f, kSymmetricExtentDirection, kJoinOperation);

//8 эскиз

PlanarSketch\* pSketch34;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch34);

SketchPointPtr point34[5]; SketchLinePtr lines34[2];

SketchArcs\* skArcs34; SketchPoints\* skPoints34;

SketchLines\* skLines34; Profiles\* skProfiles34; SketchArc\* arc34[2];

pSketch34->get\_SketchPoints(&skPoints34);

pSketch34->get\_SketchLines(&skLines34);

pSketch34->get\_Profiles(&skProfiles34);

pSketch34->get\_SketchArcs(&skArcs34);

double yy1 = d / 2 - 9; double yy3 = d / 2;

point34[0] = skPoints34->MethodAdd(pTransGeom->MethodCreatePoint2d(0, -yy3), false);

point34[1] = skPoints34->MethodAdd(pTransGeom->MethodCreatePoint2d(21.3, -yy1), false);

point34[2] = skPoints34->MethodAdd(pTransGeom->MethodCreatePoint2d(16.7, -yy1), false);

point34[3] = skPoints34->MethodAdd(pTransGeom->MethodCreatePoint2d(-16.7, -yy1), false);

point34[4] = skPoints34->MethodAdd(pTransGeom->MethodCreatePoint2d(-21.3, -yy1), false);

lines34[0] = skLines34->MethodAddByTwoPoints(point34[1], point34[2]);

lines34[1] = skLines34->MethodAddByTwoPoints(point34[4], point34[3]);

arc34[0] = skArcs34->MethodAddByCenterStartEndPoint(point34[0], point34[1], point34[4], false);

arc34[1] = skArcs34->MethodAddByCenterStartEndPoint(point34[0], point34[2], point34[3], false);

ProfilePtr pProfile34;

Try { pProfile34 = skProfiles34->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude34;

ft->get\_ExtrudeFeatures(&ftExtrude34);

ExtrudeFeaturePtr extrude34 = ftExtrude34->MethodAddByDistanceExtent(pProfile34, 35.0f, kSymmetricExtentDirection, kJoinOperation);

// 9 эскиз

wp->MethodAddByPlaneAndOffset(wp->GetItem(1), -30.0f, false);

PlanarSketch\* pSketch35;

sketches->raw\_Add(wp->GetItem(4), false, &pSketch35);

SketchPointPtr point35[6]; SketchLinePtr lines35[6]; SketchPoints\* skPoints35;

SketchLines\* skLines35; Profiles\* skProfiles35;

pSketch35->get\_SketchPoints(&skPoints35);

pSketch35->get\_SketchLines(&skLines35);

pSketch35->get\_Profiles(&skProfiles35);

double yp1 = d / 2 + 5.5; double yp2 = d / 2 - 2; double yp3 = d / 2 - 2.5; double yp4 = d / 2 + 1.5;

point35[0] = skPoints35->MethodAdd(pTransGeom->MethodCreatePoint2d(yp2, -40), false);

point35[1] = skPoints35->MethodAdd(pTransGeom->MethodCreatePoint2d(yp1, -32.3), false);

point35[2] = skPoints35->MethodAdd(pTransGeom->MethodCreatePoint2d(yp4, -28.3), false);

point35[3] = skPoints35->MethodAdd(pTransGeom->MethodCreatePoint2d(yp3, -32.5), false);

point35[4] = skPoints35->MethodAdd(pTransGeom->MethodCreatePoint2d(7, 0), false);

point35[5] = skPoints35->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

lines35[0] = skLines35->MethodAddByTwoPoints(point35[0], point35[1]);

lines35[1] = skLines35->MethodAddByTwoPoints(point35[1], point35[2]);

lines35[2] = skLines35->MethodAddByTwoPoints(point35[2], point35[3]);

lines35[3] = skLines35->MethodAddByTwoPoints(point35[3], point35[4]);

lines35[4] = skLines35->MethodAddByTwoPoints(point35[4], point35[5]);

lines35[5] = skLines35->MethodAddByTwoPoints(point35[5], point35[0]);

ProfilePtr pProfile35;

Try { pProfile35 = skProfiles35->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return;

RevolveFeaturesPtr ftRefolve35;

ft->get\_RevolveFeatures(&ftRefolve35);

ExtrudeFeaturePtr extrude35 = ftRefolve35->MethodAddFull(pProfile35, lines35[5], kJoinOperation);

// 10 эскиз

wp->MethodAddByPlaneAndOffset(wp->GetItem(1), 30.0f, false);

PlanarSketch\* pSketch36;

sketches->raw\_Add(wp->GetItem(5), false, &pSketch36);

SketchPointPtr point36[6]; SketchLinePtr lines36[6]; SketchPoints\* skPoints36;

SketchLines\* skLines36; Profiles\* skProfiles36;

pSketch36->get\_SketchPoints(&skPoints36);

pSketch36->get\_SketchLines(&skLines36);

pSketch36->get\_Profiles(&skProfiles36);

point36[0] = skPoints36->MethodAdd(pTransGeom->MethodCreatePoint2d(yp2, -40), false);

point36[1] = skPoints36->MethodAdd(pTransGeom->MethodCreatePoint2d(yp1, -32.3), false);

point36[2] = skPoints36->MethodAdd(pTransGeom->MethodCreatePoint2d(yp4, -28.3), false);

point36[3] = skPoints36->MethodAdd(pTransGeom->MethodCreatePoint2d(yp3, -32.5), false);

point36[4] = skPoints36->MethodAdd(pTransGeom->MethodCreatePoint2d(7, 0), false);

point36[5] = skPoints36->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

lines36[0] = skLines36->MethodAddByTwoPoints(point36[0], point36[1]);

lines36[1] = skLines36->MethodAddByTwoPoints(point36[1], point36[2]);

lines36[2] = skLines36->MethodAddByTwoPoints(point36[2], point36[3]);

lines36[3] = skLines36->MethodAddByTwoPoints(point36[3], point36[4]);

lines36[4] = skLines36->MethodAddByTwoPoints(point36[4], point36[5]);

lines36[5] = skLines36->MethodAddByTwoPoints(point36[5], point36[0]);

ProfilePtr pProfile36;

Try { pProfile36 = skProfiles36->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

RevolveFeaturesPtr ftRefolve36;

ft->get\_RevolveFeatures(&ftRefolve36);

ExtrudeFeaturePtr extrude36 = ftRefolve36->MethodAddFull(pProfile36, lines36[5], kJoinOperation);

//13 эскиз

PlanarSketch\* pSketch40;

sketches->raw\_Add(wp->GetItem(2), false, &pSketch40);

SketchPointPtr point40[4]; SketchLinePtr lines40[4]; SketchPoints\* skPoints40;

SketchLines\* skLines40; Profiles\* skProfiles40;

pSketch40->get\_SketchPoints(&skPoints40);

pSketch40->get\_SketchLines(&skLines40);

pSketch40->get\_Profiles(&skProfiles40);

point40[0] = skPoints40->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 20), false);

point40[1] = skPoints40->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, 20), false);

point40[2] = skPoints40->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, -20), false);

point40[3] = skPoints40->MethodAdd(pTransGeom->MethodCreatePoint2d(25, -20), false);

lines40[0] = skLines40->MethodAddByTwoPoints(point40[0], point40[1]);

lines40[1] = skLines40->MethodAddByTwoPoints(point40[1], point40[2]);

lines40[2] = skLines40->MethodAddByTwoPoints(point40[2], point40[3]);

lines40[3] = skLines40->MethodAddByTwoPoints(point40[3], point40[0]);

ProfilePtr pProfile40;

Try { pProfile40 = skProfiles40->MethodAddForSolid(true);}

catch (...) {AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude40;

ft->get\_ExtrudeFeatures(&ftExtrude40);

ExtrudeFeaturePtr extrude40 = ftExtrude40->MethodAddByDistanceExtent(pProfile40, 150.0f, kSymmetricExtentDirection, kCutOperation);

// 11 эскиз

PlanarSketch\* pSketch37;

sketches->raw\_Add(wp->GetItem(5), false, &pSketch37);

SketchPointPtr point37[6]; SketchLinePtr lines37[6]; SketchPoints\* skPoints37;

SketchLines\* skLines37; Profiles\* skProfiles37;

pSketch37->get\_SketchPoints(&skPoints37);

pSketch37->get\_SketchLines(&skLines37);

pSketch37->get\_Profiles(&skProfiles37);

double yp5 = d / 2 + 10; double yp6 = d / 2 + 7;

point37[0] = skPoints37->MethodAdd(pTransGeom->MethodCreatePoint2d(yp5, 0), false);

point37[1] = skPoints37->MethodAdd(pTransGeom->MethodCreatePoint2d(40, 0), false);

point37[2] = skPoints37->MethodAdd(pTransGeom->MethodCreatePoint2d(40, 3), false);

point37[3] = skPoints37->MethodAdd(pTransGeom->MethodCreatePoint2d(yp6, 3), false);

point37[4] = skPoints37->MethodAdd(pTransGeom->MethodCreatePoint2d(yp6, 8), false);

point37[5] = skPoints37->MethodAdd(pTransGeom->MethodCreatePoint2d(yp5, 8), false);

lines37[0] = skLines37->MethodAddByTwoPoints(point37[0], point37[1]);

lines37[1] = skLines37->MethodAddByTwoPoints(point37[1], point37[2]);

lines37[2] = skLines37->MethodAddByTwoPoints(point37[2], point37[3]);

lines37[3] = skLines37->MethodAddByTwoPoints(point37[3], point37[4]);

lines37[4] = skLines37->MethodAddByTwoPoints(point37[4], point37[5]);

lines37[5] = skLines37->MethodAddByTwoPoints(point37[5], point37[0]);

ProfilePtr pProfile37;

Try { pProfile37 = skProfiles37->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

RevolveFeaturesPtr ftRefolve37;

ft->get\_RevolveFeatures(&ftRefolve37);

ExtrudeFeaturePtr extrude37 = ftRefolve37->MethodAddFull(pProfile37, lines37[0], kJoinOperation);

// 12 эскиз

wp->MethodAddByPlaneAndOffset(wp->GetItem(1), 50.0f, false);

PlanarSketch\* pSketch38;

sketches->raw\_Add(wp->GetItem(6), false, &pSketch38);

SketchPointPtr point38[6]; SketchLinePtr lines38[6]; SketchPoints\* skPoints38;

SketchLines\* skLines38; Profiles\* skProfiles38;

pSketch38->get\_SketchPoints(&skPoints38);

pSketch38->get\_SketchLines(&skLines38);

pSketch38->get\_Profiles(&skProfiles38);

point38[0] = skPoints38->MethodAdd(pTransGeom->MethodCreatePoint2d(yp5, 0), false);

point38[1] = skPoints38->MethodAdd(pTransGeom->MethodCreatePoint2d(40, 0), false);

point38[2] = skPoints38->MethodAdd(pTransGeom->MethodCreatePoint2d(40, 3), false);

point38[3] = skPoints38->MethodAdd(pTransGeom->MethodCreatePoint2d(yp6, 3), false);

point38[4] = skPoints38->MethodAdd(pTransGeom->MethodCreatePoint2d(yp6, 8), false);

point38[5] = skPoints38->MethodAdd(pTransGeom->MethodCreatePoint2d(yp5, 8), false);

lines38[0] = skLines38->MethodAddByTwoPoints(point38[0], point38[1]);

lines38[1] = skLines38->MethodAddByTwoPoints(point38[1], point38[2]);

lines38[2] = skLines38->MethodAddByTwoPoints(point38[2], point38[3]);

lines38[3] = skLines38->MethodAddByTwoPoints(point38[3], point38[4]);

lines38[4] = skLines38->MethodAddByTwoPoints(point38[4], point38[5]);

lines38[5] = skLines38->MethodAddByTwoPoints(point38[5], point38[0]);

ProfilePtr pProfile38;

Try { pProfile38 = skProfiles38->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

RevolveFeaturesPtr ftRefolve38;

ft->get\_RevolveFeatures(&ftRefolve38);

ExtrudeFeaturePtr extrude38 = ftRefolve38->MethodAddFull(pProfile38, lines38[0], kJoinOperation);

//12 эскиз

PlanarSketch\* pSketch39;

sketches->raw\_Add(wp->GetItem(1), false, &pSketch39);

SketchPointPtr point39; SketchCirclePtr circ39;

Profiles\* skProfiles39; SketchCirclesPtr skCircles39; SketchPoints\* skPoints39;

pSketch39->get\_Profiles(&skProfiles39);

pSketch39->get\_SketchCircles(&skCircles39);

pSketch39->get\_SketchPoints(&skPoints39);

point39 = skPoints39->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

circ39 = skCircles39->MethodAddByCenterRadius(point39, (d - 9) / 2);

ProfilePtr pProfile39;

Try { pProfile39 = skProfiles39->MethodAddForSolid(true); }

catch (...){ AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude39;

ft->get\_ExtrudeFeatures(&ftExtrude39);

ExtrudeFeaturePtr extrude39 = ftExtrude39->MethodAddByDistanceExtent(pProfile39, l - 220, kSymmetricExtentDirection, kCutOperation);

pPartDoc8->MethodSaveAs("D://Прога//Детали//Корпус.ipt", true);

//КРЫШКА

pPartDoc5 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc5->DisplayName = \_T("Крышка");

pPartDoc5->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc5);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches); pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp); pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax); pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch8; sketches->raw\_Add(wp->GetItem(3), false, &pSketch8);

SketchPointPtr point8[16]; SketchLinePtr lines8[16];

SketchPoints\* skPoints8; SketchLines\* skLines8; Profiles\* skProfiles8;

pSketch8->get\_SketchPoints(&skPoints8);

pSketch8->get\_SketchLines(&skLines8);

pSketch8->get\_Profiles(&skProfiles8);

double rk1 = d / 2 - 0.5; double rk2 = d / 2 + 4;

point8[0] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(5, 0), false);

point8[1] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(5, rk1), false);

point8[2] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(0, rk1), false);

point8[3] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(0, rk2), false);

point8[4] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(15, rk2), false);

point8[5] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(15, 22.5), false);

point8[6] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 22.5), false);

point8[7] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 15), false);

point8[8] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(65, 7), false);

point8[9] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(65, 12.5), false);

point8[10] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(75, 12.5), false);

point8[11] = skPoints8->MethodAdd(pTransGeom->MethodCreatePoint2d(75, 0), false);

lines8[0] = skLines8->MethodAddByTwoPoints(point8[0], point8[1]);

lines8[1] = skLines8->MethodAddByTwoPoints(point8[1], point8[2]);

lines8[2] = skLines8->MethodAddByTwoPoints(point8[2], point8[3]);

lines8[3] = skLines8->MethodAddByTwoPoints(point8[3], point8[4]);

lines8[4] = skLines8->MethodAddByTwoPoints(point8[4], point8[5]);

lines8[5] = skLines8->MethodAddByTwoPoints(point8[5], point8[6]);

lines8[6] = skLines8->MethodAddByTwoPoints(point8[6], point8[7]);

lines8[7] = skLines8->MethodAddByTwoPoints(point8[7], point8[8]);

lines8[8] = skLines8->MethodAddByTwoPoints(point8[8], point8[9]);

lines8[9] = skLines8->MethodAddByTwoPoints(point8[9], point8[10]);

lines8[10] = skLines8->MethodAddByTwoPoints(point8[10], point8[11]);

lines8[11] = skLines8->MethodAddByTwoPoints(point8[11], point8[0]);

ProfilePtr pProfile8;

Try { pProfile8 = skProfiles8->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

RevolveFeaturesPtr ftRefolve8;

ft->get\_RevolveFeatures(&ftRefolve8);

ExtrudeFeaturePtr extrude8 = ftRefolve8->MethodAddFull(pProfile8, wax->GetItem(1), kJoinOperation);

//фаски

ChamferFeaturesPtr pChamferFt;

ft->get\_ChamferFeatures(&pChamferFt);

EdgeCollectionPtr edgeColl;

pInvApp->TransientObjects->raw\_CreateEdgeCollection(vtMissing, &edgeColl);

SurfaceBodyPtr SurfBody; SurfaceBodiesPtr SurfBodies;

pPartComDef->get\_SurfaceBodies(&SurfBodies); edgeColl->MethodClear();

pPartComDef->get\_SurfaceBodies(&SurfBodies); EdgesPtr edges;

SurfBodies->get\_Item(1, &SurfBody);

SurfBody->get\_Edges(&edges);

EdgePtr ed; edges->get\_Item(5, &ed); edgeColl->MethodAdd(ed);

edges->get\_Item(1, &ed); edgeColl->MethodAdd(ed);

ChamferFeaturePtr chamFeature = pChamferFt->MethodAddUsingDistance(edgeColl, 2.5f, false, false, false);

pPartDoc5->MethodSaveAs("D://Прога//Детали//Крышка.ipt", true);

//БАЛКИ

pPartDoc1 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc1->DisplayName = \_T("Балки");

pPartDoc1->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc1);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches); pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp); pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax); pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch; sketches->raw\_Add(wp->GetItem(3), false, &pSketch);

SketchPointPtr point[8]; SketchLinePtr lines[8]; SketchPoints\* skPoints;

SketchLines\* skLines; Profiles\* skProfiles;

pSketch->get\_SketchPoints(&skPoints); pSketch->get\_SketchLines(&skLines);

pSketch->get\_Profiles(&skProfiles);

point[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 18), false);

point[1] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 25), false);

point[2] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(130, 25), false);

point[3] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(130, 18), false);

point[4] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 45), false);

point[5] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 52), false);

point[6] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(130, 63.4), false);

point[7] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(130, 56.4), false);

lines[0] = skLines->MethodAddByTwoPoints(point[0], point[1]);

lines[1] = skLines->MethodAddByTwoPoints(point[1], point[2]);

lines[2] = skLines->MethodAddByTwoPoints(point[2], point[3]);

lines[3] = skLines->MethodAddByTwoPoints(point[3], point[0]);

lines[4] = skLines->MethodAddByTwoPoints(point[4], point[5]);

lines[5] = skLines->MethodAddByTwoPoints(point[5], point[6]);

lines[6] = skLines->MethodAddByTwoPoints(point[6], point[7]);

lines[7] = skLines->MethodAddByTwoPoints(point[7], point[4]);

ProfilePtr pProfile;

Try { pProfile = skProfiles->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude;

ft->get\_ExtrudeFeatures(&ftExtrude);

ExtrudeFeaturePtr extrude = ftExtrude->MethodAddByDistanceExtent(pProfile, 15.0f, kSymmetricExtentDirection, kJoinOperation);

pPartDoc1->MethodSaveAs("D://Прога//Детали//Балки.ipt", true);

//МУФТА

pPartDoc2 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc2->DisplayName = \_T("Муфта");

pPartDoc2->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc2);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches); pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp); pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax); pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch1;

sketches->raw\_Add(wp->GetItem(3), false, &pSketch1);

SketchPointPtr point1[16]; SketchLinePtr lines1[16]; SketchPoints\* skPoints1;

SketchLines\* skLines1;Profiles\* skProfiles1;

pSketch1->get\_SketchPoints(&skPoints1); pSketch1->get\_SketchLines(&skLines1);

pSketch1->get\_Profiles(&skProfiles1);

point1[0] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point1[1] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 6.5), false);

point1[2] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(12, 6.5), false);

point1[3] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(12, 9.5), false);

point1[4] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(16.8, 9.5), false);

point1[5] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(16.8, 14.5), false);

point1[6] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(29, 14.5), false);

point1[7] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(29, 20), false);

point1[8] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(35, 20), false);

point1[9] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(35, 14.5), false);

point1[10] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(47.2, 14.5), false);

point1[11] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(47.2, 9.5), false);

point1[12] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(52, 9.5), false);

point1[13] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(52, 6.5), false);

point1[14] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(60, 6.5), false);

point1[15] = skPoints1->MethodAdd(pTransGeom->MethodCreatePoint2d(60, 0), false);

lines1[0] = skLines1->MethodAddByTwoPoints(point1[0], point1[1]);

lines1[1] = skLines1->MethodAddByTwoPoints(point1[1], point1[2]);

lines1[2] = skLines1->MethodAddByTwoPoints(point1[2], point1[3]);

lines1[3] = skLines1->MethodAddByTwoPoints(point1[3], point1[4]);

lines1[4] = skLines1->MethodAddByTwoPoints(point1[4], point1[5]);

lines1[5] = skLines1->MethodAddByTwoPoints(point1[5], point1[6]);

lines1[6] = skLines1->MethodAddByTwoPoints(point1[6], point1[7]);

lines1[7] = skLines1->MethodAddByTwoPoints(point1[7], point1[8]);

lines1[8] = skLines1->MethodAddByTwoPoints(point1[8], point1[9]);

lines1[9] = skLines1->MethodAddByTwoPoints(point1[9], point1[10]);

lines1[10] = skLines1->MethodAddByTwoPoints(point1[10], point1[11]);

lines1[11] = skLines1->MethodAddByTwoPoints(point1[11], point1[12]);

lines1[12] = skLines1->MethodAddByTwoPoints(point1[12], point1[13]);

lines1[13] = skLines1->MethodAddByTwoPoints(point1[13], point1[14]);

lines1[14] = skLines1->MethodAddByTwoPoints(point1[14], point1[15]);

lines1[15] = skLines1->MethodAddByTwoPoints(point1[15], point1[0]);

ProfilePtr pProfile1;

Try { pProfile1 = skProfiles1->MethodAddForSolid(true); }

catch (...){ AfxMessageBox(L"Ошибочный контур!"); return; }

RevolveFeaturesPtr ftRefolve1;

ft->get\_RevolveFeatures(&ftRefolve1);

ExtrudeFeaturePtr extrude1 = ftRefolve1->MethodAddFull(pProfile1, wax->GetItem(1), kJoinOperation);

pPartDoc2->MethodSaveAs("D://Прога//Детали//Муфта.ipt", true);

//ОПОРА

//1 эскиз

pPartDoc3 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc3->DisplayName = \_T("Опорный Элемент");

pPartDoc3->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc3);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches); pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp); pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax); pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch2; sketches->raw\_Add(wp->GetItem(1), false, &pSketch2);

SketchPointPtr point2[5]; SketchLinePtr lines2[4];

SketchCirclePtr circ2; SketchPoints\* skPoints2;

SketchLines\* skLines2; Profiles\* skProfiles2;

SketchCirclesPtr skCircles2; pSketch2->get\_SketchPoints(&skPoints2);

pSketch2->get\_SketchLines(&skLines2); pSketch2->get\_Profiles(&skProfiles2);

pSketch2->get\_SketchCircles(&skCircles2);double rzhh = h - 190;

point2[0] = skPoints2->MethodAdd(pTransGeom->MethodCreatePoint2d(101+ rzhh, 0), false);

point2[1] = skPoints2->MethodAdd(pTransGeom->MethodCreatePoint2d(0, -35), false);

point2[2] = skPoints2->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 35), false);

point2[3] = skPoints2->MethodAdd(pTransGeom->MethodCreatePoint2d( 103.75 + rzhh, -12.91), false);

point2[4] = skPoints2->MethodAdd(pTransGeom->MethodCreatePoint2d(103.75 + rzhh, 12.91), false);

lines2[0] = skLines2->MethodAddByTwoPoints(point2[1], point2[2]);

lines2[1] = skLines2->MethodAddByTwoPoints(point2[2], point2[4]);

lines2[2] = skLines2->MethodAddByTwoPoints(point2[4], point2[3]);

lines2[3] = skLines2->MethodAddByTwoPoints(point2[3], point2[1]);

circ2 = skCircles2->MethodAddByCenterRadius(point2[0], 13.2);

ProfilePtr pProfile2;

Try { pProfile2 = skProfiles2->MethodAddForSolid(true);}

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude2;

ft->get\_ExtrudeFeatures(&ftExtrude2);

ExtrudeFeaturePtr extrude2 = ftExtrude2->MethodAddByDistanceExtent(pProfile2, 20.0f, kPositiveExtentDirection, kJoinOperation);

//2 и 3 эскиз

PlanarSketch\* pSketch4;

sketches->raw\_Add(wp->GetItem(2), false, &pSketch4);

SketchPointPtr point4; SketchCirclePtr circ4; Profiles\* skProfiles4;

SketchCirclesPtr skCircles4; SketchPoints\* skPoints4;

pSketch4->get\_Profiles(&skProfiles4); pSketch4->get\_SketchCircles(&skCircles4);

pSketch4->get\_SketchPoints(&skPoints4);

point4 = skPoints4->MethodAdd(pTransGeom->MethodCreatePoint2d(-10, 0), false);

circ4 = skCircles4->MethodAddByCenterRadius(point4, 8);

ProfilePtr pProfile4;

Try { pProfile4 = skProfiles4->MethodAddForSolid(true);}

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude4;

ft->get\_ExtrudeFeatures(&ftExtrude4);

ExtrudeFeaturePtr extrude4 = ftExtrude4->MethodAddByDistanceExtent(pProfile4, 118.0f + rzhh, kPositiveExtentDirection, kJoinOperation);

PlanarSketch\* pSketch5;

sketches->raw\_Add(wp->GetItem(2), false, &pSketch5);

SketchPointPtr point5; SketchCirclePtr circ5; Profiles\* skProfiles5;

SketchCirclesPtr skCircles5;SketchPoints\* skPoints5;

pSketch5->get\_Profiles(&skProfiles5); pSketch5->get\_SketchCircles(&skCircles5);

pSketch5->get\_SketchPoints(&skPoints5);

point5 = skPoints5->MethodAdd(pTransGeom->MethodCreatePoint2d(-10, 0), false);

circ5 = skCircles5->MethodAddByCenterRadius(point5, 3.5);

ProfilePtr pProfile5;

Try { pProfile5 = skProfiles5->MethodAddForSolid(true); }

catch (...){ AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude5;

ft->get\_ExtrudeFeatures(&ftExtrude5);

ExtrudeFeaturePtr extrude5 = ftExtrude5->MethodAddByDistanceExtent(pProfile5, 120.0f + rzhh, kPositiveExtentDirection, kJoinOperation);

//4 эскиз

PlanarSketch\* pSketch3;

sketches->raw\_Add(wp->GetItem(3), false, &pSketch3);

SketchPointPtr point3[8]; SketchLinePtr lines3[8]; SketchPoints\* skPoints3;

SketchLines\* skLines3; Profiles\* skProfiles3;

pSketch3->get\_SketchPoints(&skPoints3); pSketch3->get\_SketchLines(&skLines3);

pSketch3->get\_Profiles(&skProfiles3);

point3[0] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d(20, 76.2 + rzhh), false);

point3[1] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d(17,79.2 + rzhh), false);

point3[2] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d(17,83.2 + rzhh), false);

point3[3] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d(20,86.2 + rzhh), false);

point3[4] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d( 0,76.2 + rzhh), false);

point3[5] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d(3,79.2 + rzhh), false);

point3[6] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d(3,83.2 + rzhh), false);

point3[7] = skPoints3->MethodAdd(pTransGeom->MethodCreatePoint2d( 0,86.2 + rzhh), false);

lines3[0] = skLines3->MethodAddByTwoPoints(point3[0], point3[1]);

lines3[1] = skLines3->MethodAddByTwoPoints(point3[1], point3[2]);

lines3[2] = skLines3->MethodAddByTwoPoints(point3[2], point3[3]);

lines3[3] = skLines3->MethodAddByTwoPoints(point3[3], point3[0]);

lines3[4] = skLines3->MethodAddByTwoPoints(point3[4], point3[5]);

lines3[5] = skLines3->MethodAddByTwoPoints(point3[5], point3[6]);

lines3[6] = skLines3->MethodAddByTwoPoints(point3[6], point3[7]);

lines3[7] = skLines3->MethodAddByTwoPoints(point3[7], point3[4]);

ProfilePtr pProfile3;

Try { pProfile3 = skProfiles3->MethodAddForSolid(true);}

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude3;

ft->get\_ExtrudeFeatures(&ftExtrude3);

ExtrudeFeaturePtr extrude3 = ftExtrude3->MethodAddByDistanceExtent(pProfile3, 40.0f, kSymmetricExtentDirection, kCutOperation);

pPartDoc3->MethodSaveAs("D://Прога//Детали//Опорный элемент.ipt", true);

//НОЖКА

//1 эскиз

pPartDoc4 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc4->DisplayName = \_T("Ножка");

pPartDoc4->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager(); Doc = CComQIPtr <Document>(pPartDoc4);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches); pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp); pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax); pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch6; sketches->raw\_Add(wp->GetItem(3), false, &pSketch6);

SketchPointPtr point6[5]; SketchLinePtr lines6[5];

SketchPoints\* skPoints6; SketchLines\* skLines6; Profiles\* skProfiles6;

pSketch6->get\_SketchPoints(&skPoints6); pSketch6->get\_SketchLines(&skLines6);

pSketch6->get\_Profiles(&skProfiles6); double rzh = h - d - 81; double rzz = h - 190;

point6[0] = skPoints6->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, 0), false);

point6[1] = skPoints6->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point6[2] = skPoints6->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 49 + rzh), false);

point6[3] = skPoints6->MethodAdd(pTransGeom->MethodCreatePoint2d(-7, 49 + rzh), false);

point6[4] = skPoints6->MethodAdd(pTransGeom->MethodCreatePoint2d(-7, 40 + rzz), false);

lines6[0] = skLines6->MethodAddByTwoPoints(point6[0], point6[1]);

lines6[1] = skLines6->MethodAddByTwoPoints(point6[1], point6[2]);

lines6[2] = skLines6->MethodAddByTwoPoints(point6[2], point6[3]);

lines6[3] = skLines6->MethodAddByTwoPoints(point6[3], point6[4]);

lines6[4] = skLines6->MethodAddByTwoPoints(point6[4], point6[0]);

ProfilePtr pProfile6;

Try { pProfile6 = skProfiles6->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude6;

ft->get\_ExtrudeFeatures(&ftExtrude6);

ExtrudeFeaturePtr extrude6 = ftExtrude6->MethodAddByDistanceExtent(pProfile6, 50.0f, kSymmetricExtentDirection, kJoinOperation);

//2 эскиз

PlanarSketch\* pSketch7; sketches->raw\_Add(wp->GetItem(1), false, &pSketch7);

SketchPointPtr point7[5]; SketchCirclePtr circ7; SketchLinePtr lines7[4];

Profiles\* skProfiles7; SketchCirclesPtr skCircles7;

SketchPoints\* skPoints7; SketchLines\* skLines7;

pSketch7->get\_Profiles(&skProfiles7); pSketch7->get\_SketchCircles(&skCircles7);

pSketch7->get\_SketchPoints(&skPoints7); pSketch7->get\_SketchLines(&skLines7);

double rr = h - 190;

point7[0] = skPoints7->MethodAdd(pTransGeom->MethodCreatePoint2d(101+rr, 0), false);

circ7 = skCircles7->MethodAddByCenterRadius(point7[0], d/2);

point7[1] = skPoints7->MethodAdd(pTransGeom->MethodCreatePoint2d(6, -10), false);

point7[2] = skPoints7->MethodAdd(pTransGeom->MethodCreatePoint2d(6, 10), false);

point7[3] = skPoints7->MethodAdd(pTransGeom->MethodCreatePoint2d(-2, 10), false);

point7[4] = skPoints7->MethodAdd(pTransGeom->MethodCreatePoint2d(-2, -10), false);

lines7[0] = skLines7->MethodAddByTwoPoints(point7[1], point7[2]);

lines7[1] = skLines7->MethodAddByTwoPoints(point7[2], point7[3]);

lines7[2] = skLines7->MethodAddByTwoPoints(point7[3], point7[4]);

lines7[3] = skLines7->MethodAddByTwoPoints(point7[4], point7[1]);

ProfilePtr pProfile7;

Try { pProfile7 = skProfiles7->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude7;

ft->get\_ExtrudeFeatures(&ftExtrude7);

ExtrudeFeaturePtr extrude7 = ftExtrude7->MethodAddByDistanceExtent(pProfile7, 30.0f, kNegativeExtentDirection, kCutOperation);

pPartDoc4->MethodSaveAs("D://Прога//Детали//Ножка.ipt", true);

//ВЕРХНИЙ ОТСЕК

//1 эскиз

pPartDoc6 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc6->DisplayName = \_T("Верхний отсек ");

pPartDoc6->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc6);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches);

pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp);

pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax);

pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch9; sketches->raw\_Add(wp->GetItem(2), false, &pSketch9);

SketchPointPtr point9[5]; SketchLinePtr lines9[5]; SketchPoints\* skPoints9;

SketchLines\* skLines9; Profiles\* skProfiles9;

pSketch9->get\_SketchPoints(&skPoints9); pSketch9->get\_SketchLines(&skLines9);

pSketch9->get\_Profiles(&skProfiles9);

point9[0] = skPoints9->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, 20), false);

point9[1] = skPoints9->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, -20), false);

point9[2] = skPoints9->MethodAdd(pTransGeom->MethodCreatePoint2d(25, -20), false);

point9[3] = skPoints9->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 20), false);

lines9[0] = skLines9->MethodAddByTwoPoints(point9[0], point9[1]);

lines9[1] = skLines9->MethodAddByTwoPoints(point9[1], point9[2]);

lines9[2] = skLines9->MethodAddByTwoPoints(point9[2], point9[3]);

lines9[3] = skLines9->MethodAddByTwoPoints(point9[3], point9[0]);

ProfilePtr pProfile9;

Try { pProfile9 = skProfiles9->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude9;

ft->get\_ExtrudeFeatures(&ftExtrude9);

ExtrudeFeaturePtr extrude9 = ftExtrude9->MethodAddByDistanceExtent(pProfile9, 50.0f, kPositiveExtentDirection, kJoinOperation);

//2 эскиз

PlanarSketch\* pSketch10; sketches->raw\_Add(wp->GetItem(2), false, &pSketch10);

SketchPointPtr point10; SketchCirclePtr circ10;

Profiles\* skProfiles10; SketchCirclesPtr skCircles10;

SketchPoints\* skPoints10; pSketch10->get\_Profiles(&skProfiles10);

pSketch10->get\_SketchCircles(&skCircles10); pSketch10->get\_SketchPoints(&skPoints10);

point10 = skPoints10->MethodAdd(pTransGeom->MethodCreatePoint2d(-6.5, 0), false);

circ10 = skCircles10->MethodAddByCenterRadius(point10, 15);

ProfilePtr pProfile10;

Try { pProfile10 = skProfiles10->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude10;

ft->get\_ExtrudeFeatures(&ftExtrude10);

ExtrudeFeaturePtr extrude10 = ftExtrude10->MethodAddByDistanceExtent(pProfile10, 55.0f, kPositiveExtentDirection, kJoinOperation);

//3 эскиз на грани детали

SurfaceBodyPtr SurfBody1; SurfaceBodiesPtr SurfBodies1;

pPartComDef->get\_SurfaceBodies(&SurfBodies1); SurfBodies1->get\_Item(1, &SurfBody1);

PlanarSketch\* pSketch11;

sketches->raw\_Add(SurfBody1->Faces->GetItem(2), false, &pSketch11); //номер грани

SketchPointPtr point11; SketchCirclePtr circ11;

Profiles\* skProfiles11; SketchCirclesPtr skCircles11;

SketchPoints\* skPoints11; pSketch11->get\_Profiles(&skProfiles11);

pSketch11->get\_SketchCircles(&skCircles11); pSketch11->get\_SketchPoints(&skPoints11);

point11 = skPoints11->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

circ11 = skCircles11->MethodAddByCenterRadius(point11, 10);

ProfilePtr pProfile11;

Try { pProfile11 = skProfiles11->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return;}

ExtrudeFeaturesPtr ftExtrude11;

ft->get\_ExtrudeFeatures(&ftExtrude11);

ExtrudeFeaturePtr extrude11 = ftExtrude11->MethodAddByDistanceExtent(pProfile11, 5.0f, kNegativeExtentDirection, kCutOperation);

//4 эскиз

PlanarSketch\* pSketch12;

sketches->raw\_Add(wp->GetItem(3), false, &pSketch12);

SketchPointPtr point12[5]; SketchLinePtr lines12[5]; SketchPoints\* skPoints12;

SketchLines\* skLines12; Profiles\* skProfiles12;

pSketch12->get\_SketchPoints(&skPoints12); pSketch12->get\_SketchLines(&skLines12);

pSketch12->get\_Profiles(&skProfiles12);

point12[0] = skPoints12->MethodAdd(pTransGeom->MethodCreatePoint2d(-2.5, 50), false);

point12[1] = skPoints12->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 50), false);

point12[2] = skPoints12->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 12), false);

point12[3] = skPoints12->MethodAdd(pTransGeom->MethodCreatePoint2d(-2.5, 12), false);

lines12[0] = skLines12->MethodAddByTwoPoints(point12[0], point12[1]);

lines12[1] = skLines12->MethodAddByTwoPoints(point12[1], point12[2]);

lines12[2] = skLines12->MethodAddByTwoPoints(point12[2], point12[3]);

lines12[3] = skLines12->MethodAddByTwoPoints(point12[3], point12[0]);

ProfilePtr pProfile12;

Try { pProfile12 = skProfiles12->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude12;

ft->get\_ExtrudeFeatures(&ftExtrude12);

ExtrudeFeaturePtr extrude12 = ftExtrude12->MethodAddByDistanceExtent(pProfile12, 21.5f, kPositiveExtentDirection, kJoinOperation);

//5 эскиз

PlanarSketch\* pSketch13;

sketches->raw\_Add(wp->GetItem(3), false, &pSketch13);

SketchPointPtr point13[5]; SketchLinePtr lines13[5];

SketchPoints\* skPoints13; SketchLines\* skLines13; Profiles\* skProfiles13;

pSketch13->get\_SketchPoints(&skPoints13); pSketch13->get\_SketchLines(&skLines13);

pSketch13->get\_Profiles(&skProfiles13);

point13[0] = skPoints13->MethodAdd(pTransGeom->MethodCreatePoint2d(0.5, 47), false);

point13[1] = skPoints13->MethodAdd(pTransGeom->MethodCreatePoint2d(0.5, 15), false);

point13[2] = skPoints13->MethodAdd(pTransGeom->MethodCreatePoint2d(22, 15), false);

point13[3] = skPoints13->MethodAdd(pTransGeom->MethodCreatePoint2d(22, 47), false);

lines13[0] = skLines13->MethodAddByTwoPoints(point13[0], point13[1]);

lines13[1] = skLines13->MethodAddByTwoPoints(point13[1], point13[2]);

lines13[2] = skLines13->MethodAddByTwoPoints(point13[2], point13[3]);

lines13[3] = skLines13->MethodAddByTwoPoints(point13[3], point13[0]);

ProfilePtr pProfile13;

Try { pProfile13 = skProfiles13->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude13;

ft->get\_ExtrudeFeatures(&ftExtrude13);

ExtrudeFeaturePtr extrude13 = ftExtrude13->MethodAddByDistanceExtent(pProfile13, 22.0f, kPositiveExtentDirection, kJoinOperation);

//6 эскиз

SurfaceBodyPtr SurfBody2; SurfaceBodiesPtr SurfBodies2;

pPartComDef->get\_SurfaceBodies(&SurfBodies2); SurfBodies2->get\_Item(1, &SurfBody2);

PlanarSketch\* pSketch14;

sketches->raw\_Add(SurfBody2->Faces->GetItem(9), false, &pSketch14); //номер грани

SketchPointPtr point14[5]; SketchLinePtr lines14[5]; SketchPoints\* skPoints14;

SketchLines\* skLines14; Profiles\* skProfiles14;

pSketch14->get\_SketchPoints(&skPoints14); pSketch14->get\_SketchLines(&skLines14);

pSketch14->get\_Profiles(&skProfiles14);

point14[0] = skPoints14->MethodAdd(pTransGeom->MethodCreatePoint2d(1.5, 29), false);

point14[1] = skPoints14->MethodAdd(pTransGeom->MethodCreatePoint2d(1.5, 33), false);

point14[2] = skPoints14->MethodAdd(pTransGeom->MethodCreatePoint2d(5.5, 33), false);

point14[3] = skPoints14->MethodAdd(pTransGeom->MethodCreatePoint2d(5.5, 29), false);

lines14[0] = skLines14->MethodAddByTwoPoints(point14[0], point14[1]);

lines14[1] = skLines14->MethodAddByTwoPoints(point14[1], point14[2]);

lines14[2] = skLines14->MethodAddByTwoPoints(point14[2], point14[3]);

lines14[3] = skLines14->MethodAddByTwoPoints(point14[3], point14[0]);

ProfilePtr pProfile14;

Try { pProfile14 = skProfiles14->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude14;

ft->get\_ExtrudeFeatures(&ftExtrude14);

ExtrudeFeaturePtr extrude14 = ftExtrude14->MethodAddByDistanceExtent(pProfile14, 3.0f, kPositiveExtentDirection, kJoinOperation);

//7 эскиз

SurfaceBodyPtr SurfBody3; SurfaceBodiesPtr SurfBodies3;

pPartComDef->get\_SurfaceBodies(&SurfBodies3);

SurfBodies3->get\_Item(1, &SurfBody3); PlanarSketch\* pSketch15;

sketches->raw\_Add(SurfBody3->Faces->GetItem(4), false, &pSketch15); //номер грани

SketchPointPtr point15[5]; SketchLinePtr lines15[5];

SketchPoints\* skPoints15; SketchLines\* skLines15;

Profiles\* skProfiles15; pSketch15->get\_SketchPoints(&skPoints15);

pSketch15->get\_SketchLines(&skLines15); pSketch15->get\_Profiles(&skProfiles15);

point15[0] = skPoints15->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point15[1] = skPoints15->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 4), false);

point15[2] = skPoints15->MethodAdd(pTransGeom->MethodCreatePoint2d(3, 4), false);

point15[3] = skPoints15->MethodAdd(pTransGeom->MethodCreatePoint2d(3, 0), false);

lines15[0] = skLines15->MethodAddByTwoPoints(point15[0], point15[1]);

lines15[1] = skLines15->MethodAddByTwoPoints(point15[1], point15[2]);

lines15[2] = skLines15->MethodAddByTwoPoints(point15[2], point15[3]);

lines15[3] = skLines15->MethodAddByTwoPoints(point15[3], point15[0]);

ProfilePtr pProfile15;

Try { pProfile15 = skProfiles15->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude15;

ft->get\_ExtrudeFeatures(&ftExtrude15);

ExtrudeFeaturePtr extrude15 = ftExtrude15->MethodAddByDistanceExtent(pProfile15, 6.0f, kPositiveExtentDirection, kJoinOperation);

//8 эскиз

SurfaceBodyPtr SurfBody4;SurfaceBodiesPtr SurfBodies4;

pPartComDef->get\_SurfaceBodies(&SurfBodies4); SurfBodies4->get\_Item(1, &SurfBody4);

PlanarSketch\* pSketch16;

sketches->raw\_Add(SurfBody4->Faces->GetItem(1), false, &pSketch16); //номер грани

SketchPointPtr point16[5]; SketchLinePtr lines16[5];

SketchPoints\* skPoints16; SketchLines\* skLines16;

Profiles\* skProfiles16; pSketch16->get\_SketchPoints(&skPoints16);

pSketch16->get\_SketchLines(&skLines16); pSketch16->get\_Profiles(&skProfiles16);

point16[0] = skPoints16->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point16[1] = skPoints16->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 4), false);

point16[2] = skPoints16->MethodAdd(pTransGeom->MethodCreatePoint2d(3, 4), false);

point16[3] = skPoints16->MethodAdd(pTransGeom->MethodCreatePoint2d(3, 0), false);

lines16[0] = skLines16->MethodAddByTwoPoints(point16[0], point16[1]);

lines16[1] = skLines16->MethodAddByTwoPoints(point16[1], point16[2]);

lines16[2] = skLines16->MethodAddByTwoPoints(point16[2], point16[3]);

lines16[3] = skLines16->MethodAddByTwoPoints(point16[3], point16[0]);

ProfilePtr pProfile16;

Try { pProfile16 = skProfiles16->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude16;

ft->get\_ExtrudeFeatures(&ftExtrude16);

ExtrudeFeaturePtr extrude16 = ftExtrude16->MethodAddByDistanceExtent(pProfile16, 30.0f, kPositiveExtentDirection, kJoinOperation);

//9 эскиз

SurfaceBodyPtr SurfBody5;SurfaceBodiesPtr SurfBodies5;

pPartComDef->get\_SurfaceBodies(&SurfBodies5);

SurfBodies5->get\_Item(1, &SurfBody5);PlanarSketch\* pSketch17;

sketches->raw\_Add(SurfBody5->Faces->GetItem(8), false, &pSketch17); //номер грани

SketchPointPtr point17[5]; SketchLinePtr lines17[5];

SketchPoints\* skPoints17; SketchLines\* skLines17; Profiles\* skProfiles17;

pSketch17->get\_SketchPoints(&skPoints17); pSketch17->get\_SketchLines(&skLines17);

pSketch17->get\_Profiles(&skProfiles17);

point17[0] = skPoints17->MethodAdd(pTransGeom->MethodCreatePoint2d(-24.5, 5.8), false);

point17[1] = skPoints17->MethodAdd(pTransGeom->MethodCreatePoint2d(-32, 5.8), false);

point17[2] = skPoints17->MethodAdd(pTransGeom->MethodCreatePoint2d(-32, -1.8), false);

point17[3] = skPoints17->MethodAdd(pTransGeom->MethodCreatePoint2d(-24.5, -1.8), false);

lines17[0] = skLines17->MethodAddByTwoPoints(point17[0], point17[1]);

lines17[1] = skLines17->MethodAddByTwoPoints(point17[1], point17[2]);

lines17[2] = skLines17->MethodAddByTwoPoints(point17[2], point17[3]);

lines17[3] = skLines17->MethodAddByTwoPoints(point17[3], point17[0]);

ProfilePtr pProfile17;

Try { pProfile17 = skProfiles17->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude17;

ft->get\_ExtrudeFeatures(&ftExtrude17);

ExtrudeFeaturePtr extrude17 = ftExtrude17->MethodAddByDistanceExtent(pProfile17, 5.0f, kNegativeExtentDirection, kJoinOperation);

pPartDoc6->MethodSaveAs("D://Прога//Детали//Верхний отсек.ipt", true);

//НИЖНИЙ ОТСЕК

//1 эскиз

pPartDoc7 = pInvApp->Documents->MethodAdd(kPartDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kPartDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pPartDoc7->DisplayName = \_T("Нижний отсек ");

pPartDoc7->get\_ComponentDefinition(&pPartComDef);

pInvApp->get\_TransientGeometry(&pTransGeom);

pTransManager = pInvApp->GetTransactionManager();

Doc = CComQIPtr <Document>(pPartDoc7);

pTransManager->raw\_StartTransaction(Doc, \_T(""), &pTrans);

pPartComDef->get\_Sketches(&sketches);

pPartComDef->get\_Sketches3D(&sketches3D);

pPartComDef->get\_WorkPlanes(&wp);

pPartComDef->get\_Features(&ft);

pPartComDef->get\_WorkAxes(&wax);

pPartComDef->get\_WorkPoints(&wpt);

PlanarSketch\* pSketch18;

sketches->raw\_Add(wp->GetItem(2), false, &pSketch18);

SketchPointPtr point18[5]; SketchLinePtr lines18[5];

SketchPoints\* skPoints18; SketchLines\* skLines18; Profiles\* skProfiles18;

pSketch18->get\_SketchPoints(&skPoints18); pSketch18->get\_SketchLines(&skLines18);

pSketch18->get\_Profiles(&skProfiles18);

point18[0] = skPoints18->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, 20), false);

point18[1] = skPoints18->MethodAdd(pTransGeom->MethodCreatePoint2d(-25, -20), false);

point18[2] = skPoints18->MethodAdd(pTransGeom->MethodCreatePoint2d(25, -20), false);

point18[3] = skPoints18->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 20), false);

lines18[0] = skLines18->MethodAddByTwoPoints(point18[0], point18[1]);

lines18[1] = skLines18->MethodAddByTwoPoints(point18[1], point18[2]);

lines18[2] = skLines18->MethodAddByTwoPoints(point18[2], point18[3]);

lines18[3] = skLines18->MethodAddByTwoPoints(point18[3], point18[0]);

ProfilePtr pProfile18;

Try { pProfile18 = skProfiles18->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude18;

ft->get\_ExtrudeFeatures(&ftExtrude18);

ExtrudeFeaturePtr extrude18 = ftExtrude18->MethodAddByDistanceExtent(pProfile18, 50.0f, kSymmetricExtentDirection, kJoinOperation);

//2 эскиз

PlanarSketch\* pSketch19; sketches->raw\_Add(wp->GetItem(3), false, &pSketch19);

SketchPointPtr point19[5]; SketchLinePtr lines19[5]; SketchPoints\* skPoints19;

SketchLines\* skLines19; Profiles\* skProfiles19;

pSketch19->get\_SketchPoints(&skPoints19); pSketch19->get\_SketchLines(&skLines19);

pSketch19->get\_Profiles(&skProfiles19);

point19[0] = skPoints19->MethodAdd(pTransGeom->MethodCreatePoint2d(-2.5, 13), false);

point19[1] = skPoints19->MethodAdd(pTransGeom->MethodCreatePoint2d(-2.5, -25), false);

point19[2] = skPoints19->MethodAdd(pTransGeom->MethodCreatePoint2d(25, -25), false);

point19[3] = skPoints19->MethodAdd(pTransGeom->MethodCreatePoint2d(25, 13), false);

lines19[0] = skLines19->MethodAddByTwoPoints(point19[0], point19[1]);

lines19[1] = skLines19->MethodAddByTwoPoints(point19[1], point19[2]);

lines19[2] = skLines19->MethodAddByTwoPoints(point19[2], point19[3]);

lines19[3] = skLines19->MethodAddByTwoPoints(point19[3], point19[0]);

ProfilePtr pProfile19;

Try { pProfile19 = skProfiles19->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude19;

ft->get\_ExtrudeFeatures(&ftExtrude19);

ExtrudeFeaturePtr extrude19 = ftExtrude19->MethodAddByDistanceExtent(pProfile19, 21.5f, kPositiveExtentDirection, kJoinOperation);

//3 эскиз

PlanarSketch\* pSketch20;

sketches->raw\_Add(wp->GetItem(3), false, &pSketch20);

SketchPointPtr point20[5]; SketchLinePtr lines20[5]; SketchPoints\* skPoints20;

SketchLines\* skLines20; Profiles\* skProfiles20;

pSketch20->get\_SketchPoints(&skPoints20); pSketch20->get\_SketchLines(&skLines20);

pSketch20->get\_Profiles(&skProfiles20);

point20[0] = skPoints20->MethodAdd(pTransGeom->MethodCreatePoint2d(0.5, 10), false);

point20[1] = skPoints20->MethodAdd(pTransGeom->MethodCreatePoint2d(22, 10), false);

point20[2] = skPoints20->MethodAdd(pTransGeom->MethodCreatePoint2d(22, -22), false);

point20[3] = skPoints20->MethodAdd(pTransGeom->MethodCreatePoint2d(0.5, -22), false);

lines20[0] = skLines20->MethodAddByTwoPoints(point20[0], point20[1]);

lines20[1] = skLines20->MethodAddByTwoPoints(point20[1], point20[2]);

lines20[2] = skLines20->MethodAddByTwoPoints(point20[2], point20[3]);

lines20[3] = skLines20->MethodAddByTwoPoints(point20[3], point20[0]);

ProfilePtr pProfile20;

Try { pProfile20 = skProfiles20->MethodAddForSolid(true); }

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return; }

ExtrudeFeaturesPtr ftExtrude20;

ft->get\_ExtrudeFeatures(&ftExtrude20);

ExtrudeFeaturePtr extrude20 = ftExtrude20->MethodAddByDistanceExtent(pProfile20, 22.0f, kPositiveExtentDirection, kJoinOperation);

//4 эскиз

SurfaceBodyPtr SurfBody6; SurfaceBodiesPtr SurfBodies6;

pPartComDef->get\_SurfaceBodies(&SurfBodies6); SurfBodies6->get\_Item(1, &SurfBody6);

PlanarSketch\* pSketch21;

sketches->raw\_Add(SurfBody6->Faces->GetItem(10), false, &pSketch21); //номер грани

SketchPointPtr point21[5];SketchLinePtr lines21[5];SketchPoints\* skPoints21;

SketchLines\* skLines21;Profiles\* skProfiles21;

pSketch21->get\_SketchPoints(&skPoints21);pSketch21->get\_SketchLines(&skLines21);

pSketch21->get\_Profiles(&skProfiles21);

point21[0] = skPoints21->MethodAdd(pTransGeom->MethodCreatePoint2d(-39.5, 21), false);

point21[1] = skPoints21->MethodAdd(pTransGeom->MethodCreatePoint2d(-33.5, 21), false);

point21[2] = skPoints21->MethodAdd(pTransGeom->MethodCreatePoint2d(-33.5, 17), false);

point21[3] = skPoints21->MethodAdd(pTransGeom->MethodCreatePoint2d(-39.5, 17), false);

lines21[0] = skLines21->MethodAddByTwoPoints(point21[0], point21[1]);

lines21[1] = skLines21->MethodAddByTwoPoints(point21[1], point21[2]);

lines21[2] = skLines21->MethodAddByTwoPoints(point21[2], point21[3]);

lines21[3] = skLines21->MethodAddByTwoPoints(point21[3], point21[0]);

ProfilePtr pProfile21;

Try { pProfile21 = skProfiles21->MethodAddForSolid(true);}

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return;}

ExtrudeFeaturesPtr ftExtrude21;

ft->get\_ExtrudeFeatures(&ftExtrude21);

ExtrudeFeaturePtr extrude21 = ftExtrude21->MethodAddByDistanceExtent(pProfile21, 3.0f, kPositiveExtentDirection, kJoinOperation);

//5 эскиз

SurfaceBodyPtr SurfBody7; SurfaceBodiesPtr SurfBodies7;

pPartComDef->get\_SurfaceBodies(&SurfBodies7); SurfBodies7->get\_Item(1, &SurfBody7);

PlanarSketch\* pSketch22;

sketches->raw\_Add(SurfBody7->Faces->GetItem(3), false, &pSketch22); //номер грани

SketchPointPtr point22[5]; SketchLinePtr lines22[5];

SketchPoints\* skPoints22; SketchLines\* skLines22; Profiles\* skProfiles22;

pSketch22->get\_SketchPoints(&skPoints22); pSketch22->get\_SketchLines(&skLines22);

pSketch22->get\_Profiles(&skProfiles22);

point22[0] = skPoints22->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);

point22[1] = skPoints22->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 4), false);

point22[2] = skPoints22->MethodAdd(pTransGeom->MethodCreatePoint2d(3, 4), false);

point22[3] = skPoints22->MethodAdd(pTransGeom->MethodCreatePoint2d(3, 0), false)

lines22[0] = skLines22->MethodAddByTwoPoints(point22[0], point22[1]);

lines22[1] = skLines22->MethodAddByTwoPoints(point22[1], point22[2]);

lines22[2] = skLines22->MethodAddByTwoPoints(point22[2], point22[3]);

lines22[3] = skLines22->MethodAddByTwoPoints(point22[3], point22[0]);

ProfilePtr pProfile22;

try{pProfile22 = skProfiles22->MethodAddForSolid(true);}

catch (...) {AfxMessageBox(L"Ошибочный контур!");return;}

ExtrudeFeaturesPtr ftExtrude22;

ft->get\_ExtrudeFeatures(&ftExtrude22);

ExtrudeFeaturePtr extrude22 = ftExtrude22->MethodAddByDistanceExtent(pProfile22, 6.0f, kPositiveExtentDirection, kJoinOperation);

//6 эскиз

SurfaceBodyPtr SurfBody8; SurfaceBodiesPtr SurfBodies8;

pPartComDef->get\_SurfaceBodies(&SurfBodies8); SurfBodies8->get\_Item(1, &SurfBody8);

PlanarSketch\* pSketch23;

sketches->raw\_Add(SurfBody8->Faces->GetItem(1), false, &pSketch23); //номер грани

SketchPointPtr point23[5]; SketchLinePtr lines23[5];

SketchPoints\* skPoints23; SketchLines\* skLines23; Profiles\* skProfiles23;

pSketch23->get\_SketchPoints(&skPoints23); pSketch23->get\_SketchLines(&skLines23);

pSketch23->get\_Profiles(&skProfiles23);

point23[0] = skPoints23->MethodAdd(pTransGeom->MethodCreatePoint2d(1, 0), false);

point23[1] = skPoints23->MethodAdd(pTransGeom->MethodCreatePoint2d(1, 4), false);

point23[2] = skPoints23->MethodAdd(pTransGeom->MethodCreatePoint2d(4, 4), false);

point23[3] = skPoints23->MethodAdd(pTransGeom->MethodCreatePoint2d(4, 0), false);

lines23[0] = skLines23->MethodAddByTwoPoints(point23[0], point23[1]);

lines23[1] = skLines23->MethodAddByTwoPoints(point23[1], point23[2]);

lines23[2] = skLines23->MethodAddByTwoPoints(point23[2], point23[3]);

lines23[3] = skLines23->MethodAddByTwoPoints(point23[3], point23[0]);

ProfilePtr pProfile23;

try{pProfile23 = skProfiles23->MethodAddForSolid(true);}

catch (...) {AfxMessageBox(L"Ошибочный контур!");return;}

ExtrudeFeaturesPtr ftExtrude23;

ft->get\_ExtrudeFeatures(&ftExtrude23);

ExtrudeFeaturePtr extrude23 = ftExtrude23->MethodAddByDistanceExtent(pProfile23, 30.0f, kPositiveExtentDirection, kJoinOperation);

//7 эскиз

SurfaceBodyPtr SurfBody9;SurfaceBodiesPtr SurfBodies9;

pPartComDef->get\_SurfaceBodies(&SurfBodies9); SurfBodies9->get\_Item(1, &SurfBody9);

PlanarSketch\* pSketch24;

sketches->raw\_Add(SurfBody9->Faces->GetItem(4), false, &pSketch24); //номер грани

SketchPointPtr point24[5]; SketchLinePtr lines24[5]; SketchPoints\* skPoints24;

SketchLines\* skLines24; Profiles\* skProfiles24; pSketch24->get\_SketchPoints(&skPoints24);

pSketch24->get\_SketchLines(&skLines24); pSketch24->get\_Profiles(&skProfiles24);

point24[0] = skPoints24->MethodAdd(pTransGeom->MethodCreatePoint2d(-24.5, 5.8), false);

point24[1] = skPoints24->MethodAdd(pTransGeom->MethodCreatePoint2d(-32, 5.8), false);

point24[2] = skPoints24->MethodAdd(pTransGeom->MethodCreatePoint2d(-32, -1.8), false);

point24[3] = skPoints24->MethodAdd(pTransGeom->MethodCreatePoint2d(-24.5, -1.8), false);

lines24[0] = skLines24->MethodAddByTwoPoints(point24[0], point24[1]);

lines24[1] = skLines24->MethodAddByTwoPoints(point24[1], point24[2]);

lines24[2] = skLines24->MethodAddByTwoPoints(point24[2], point24[3]);

lines24[3] = skLines24->MethodAddByTwoPoints(point24[3], point24[0]);

ProfilePtr pProfile24;

Try {pProfile24 = skProfiles24->MethodAddForSolid(true);}

catch (...) { AfxMessageBox(L"Ошибочный контур!"); return;}

ExtrudeFeaturesPtr ftExtrude24;

ft->get\_ExtrudeFeatures(&ftExtrude24);

ExtrudeFeaturePtr extrude24 = ftExtrude24->MethodAddByDistanceExtent(pProfile24, 5.0f, kNegativeExtentDirection, kJoinOperation);

//8 эскиз

SurfaceBodyPtr SurfBody10; SurfaceBodiesPtr SurfBodies10;

pPartComDef->get\_SurfaceBodies(&SurfBodies10); SurfBodies10->get\_Item(1, &SurfBody10);

PlanarSketch\* pSketch25;

sketches->raw\_Add(SurfBody10->Faces->GetItem(24), false, &pSketch25); //номер грани

SketchPointPtr point25[5]; SketchLinePtr lines25[5];

SketchPoints\* skPoints25; SketchLines\* skLines25;

Profiles\* skProfiles25; pSketch25->get\_SketchPoints(&skPoints25);

pSketch25->get\_SketchLines(&skLines25); pSketch25->get\_Profiles(&skProfiles25);

point25[0] = skPoints25->MethodAdd(pTransGeom->MethodCreatePoint2d(28,21.5 ), false);

point25[1] = skPoints25->MethodAdd(pTransGeom->MethodCreatePoint2d( 28,-21.5), false);

point25[2] = skPoints25->MethodAdd(pTransGeom->MethodCreatePoint2d(-31,-21.5 ), false);

point25[3] = skPoints25->MethodAdd(pTransGeom->MethodCreatePoint2d(-31,21.5 ), false);

lines25[0] = skLines25->MethodAddByTwoPoints(point25[0], point25[1]);

lines25[1] = skLines25->MethodAddByTwoPoints(point25[1], point25[2]);

lines25[2] = skLines25->MethodAddByTwoPoints(point25[2], point25[3]);

lines25[3] = skLines25->MethodAddByTwoPoints(point25[3], point25[0]);

ProfilePtr pProfile25;

Try {pProfile25 = skProfiles25->MethodAddForSolid(true);}

catch (...) {AfxMessageBox(L"Ошибочный контур!");return;}

ExtrudeFeaturesPtr ftExtrude25;

ft->get\_ExtrudeFeatures(&ftExtrude25);

ExtrudeFeaturePtr extrude25 = ftExtrude25->MethodAddByDistanceExtent(pProfile25, 2.0f, kNegativeExtentDirection, kJoinOperation);

//9 эскиз

PlanarSketch\* pSketch26; sketches->raw\_Add(wp->GetItem(3), false, &pSketch26);

SketchPointPtr point26[9]; SketchLinePtr lines26[9]; SketchPoints\* skPoints26;

SketchLines\* skLines26; Profiles\* skProfiles26; pSketch26->get\_SketchPoints(&skPoints26);

pSketch26->get\_SketchLines(&skLines26); pSketch26->get\_Profiles(&skProfiles26);

double razz = (l - 340)/2;

point26[0] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(24, 5), false);

point26[1] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(39 + razz, 5), false);

point26[2] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(39 + razz, 8.5), false);

point26[3] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(42+ razz, 8.5), false);

point26[4] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(42+ razz, 7.5), false);

point26[5] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(50+ razz, 2.5), false);

point26[6] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(54+ razz, 2.5), false);

point26[7] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(64+ razz, 7.5), false);

point26[8] = skPoints26->MethodAdd(pTransGeom->MethodCreatePoint2d(64+ razz, 0), false);

lines26[0] = skLines26->MethodAddByTwoPoints(point26[0], point26[1]);

lines26[1] = skLines26->MethodAddByTwoPoints(point26[1], point26[2]);

lines26[2] = skLines26->MethodAddByTwoPoints(point26[2], point26[3]);

lines26[3] = skLines26->MethodAddByTwoPoints(point26[3], point26[4]);

lines26[4] = skLines26->MethodAddByTwoPoints(point26[4], point26[5]);

lines26[5] = skLines26->MethodAddByTwoPoints(point26[5], point26[6]);

lines26[6] = skLines26->MethodAddByTwoPoints(point26[6], point26[7]);

lines26[7] = skLines26->MethodAddByTwoPoints(point26[7], point26[8]);

lines26[8] = skLines26->MethodAddByTwoPoints(point26[8], point26[0]);

ProfilePtr pProfile26;

Try {pProfile26 = skProfiles26->MethodAddForSolid(true);}

catch (...) {AfxMessageBox(L"Ошибочный контур!");return;}

RevolveFeaturesPtr ftRefolve26;

ft->get\_RevolveFeatures(&ftRefolve26);

ExtrudeFeaturePtr extrude26 = ftRefolve26->MethodAddFull(pProfile26, wax->GetItem(1), kJoinOperation);

pPartDoc7->MethodSaveAs("D://Прога//Детали//Нижний отсек.ipt", true);

}

}

void CDlgAssembly::OnBnClickedButton4()

{

UpdateData(TRUE); BeginWaitCursor();

// Get hold of the program id of Inventor.

CLSID InvAppClsid; HRESULT hRes = CLSIDFromProgID(L"Inventor.Application", &InvAppClsid);

if (FAILED(hRes)) {

pInvApp = nullptr;

return;// ReturnAndShowCOMError(hRes, L"ConnectToInventor,CLSIDFromProgID failed");

}

// See if Inventor is already running...

CComPtr<IUnknown> pInvAppUnk = nullptr;

hRes = ::GetActiveObject(InvAppClsid, NULL, &pInvAppUnk);

if (FAILED(hRes)) {

// Inventor is not already running, so try to start it...

TRACE(L"Could not get hold of an active Inventor, will start a new session\n");

hRes = CoCreateInstance(InvAppClsid, NULL, CLSCTX\_LOCAL\_SERVER, \_\_uuidof(IUnknown), (void\*\*)&pInvAppUnk);

if (FAILED(hRes)) {

pInvApp = nullptr;

return;// ReturnAndShowCOMError(hRes, L"ConnectToInventor,CoCreateInstance failed");

}

}

// Get the pointer to the Inventor application...

if (pInvApp == NULL) { hRes = pInvAppUnk->QueryInterface(\_\_uuidof(Application), (void\*\*)&pInvApp);

}

if (FAILED(hRes)) { return;// ReturnAndShowCOMError(hRes, L"ConnectToInventor,QueryInterface failed");

}

pInvApp->put\_Visible(TRUE);

//СБОРКА//

pAsmDoc = pInvApp->Documents->MethodAdd(kAssemblyDocumentObject, pInvApp->FileManager->MethodGetTemplateFile(kAssemblyDocumentObject, kMetricSystemOfMeasure, kGOST\_DraftingStandard), true);

pAsmDoc->DisplayName = \_T("Плунжерный смеситель");

pAsmDoc->get\_ComponentDefinition(&pAsmComDef);

pInvApp->get\_TransientGeometry(&pTransGeomAsm);

pTransGeomAsm->raw\_CreateMatrix(&oPositionMatrix);

ComponentOccurrencePtr Model1, Model2, Model3, Model4, Model5, Model6, Model7, Model8, Model9, Model10;

Model1 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Корпус.ipt", oPositionMatrix);

Model2 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Крышка.ipt", oPositionMatrix);

Model3 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Крышка.ipt", oPositionMatrix);

Model4 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Верхний отсек.ipt", oPositionMatrix);

Model5 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Нижний отсек.ipt", oPositionMatrix);

Model6 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Ножка.ipt", oPositionMatrix);

Model7 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Ножка.ipt", oPositionMatrix);

Model8 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Муфта.ipt", oPositionMatrix);

Model9 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Опорный элемент.ipt", oPositionMatrix);

Model10 = pAsmDoc->ComponentDefinition->Occurrences->MethodAdd("D://Прога//Детали//Балки.ipt", oPositionMatrix);

double razz = (d - 84) / 2;

//корпус крышка право

SurfaceBody\* SurfBody1, \* SurfBody2; SurfaceBodies\* SurfBodies1, \* SurfBodies2;

SurfaceBody\* SurfBody3, \* SurfBody4; SurfaceBodies\* SurfBodies3, \* SurfBodies4;

SurfaceBody\* SurfBody5, \* SurfBody6, \* SurfBody7, \* SurfBody8, \* SurfBody9, \* SurfBody10;

SurfaceBodies\* SurfBodies5, \* SurfBodies6, \* SurfBodies7, \* SurfBodies8, \* SurfBodies9, \* SurfBodies10;

Model1->get\_SurfaceBodies(&SurfBodies1); SurfBodies1->get\_Item(1, &SurfBody1);

Model2->get\_SurfaceBodies(&SurfBodies2); SurfBodies2->get\_Item(1, &SurfBody2);

Model3->get\_SurfaceBodies(&SurfBodies3); SurfBodies3->get\_Item(1, &SurfBody3);

Model4->get\_SurfaceBodies(&SurfBodies4); SurfBodies4->get\_Item(1, &SurfBody4);

Model5->get\_SurfaceBodies(&SurfBodies5); SurfBodies5->get\_Item(1, &SurfBody5);

Model6->get\_SurfaceBodies(&SurfBodies6); SurfBodies6->get\_Item(1, &SurfBody6);

Model7->get\_SurfaceBodies(&SurfBodies7); SurfBodies7->get\_Item(1, &SurfBody7);

Model8->get\_SurfaceBodies(&SurfBodies8); SurfBodies8->get\_Item(1, &SurfBody8);

Model9->get\_SurfaceBodies(&SurfBodies9); SurfBodies9->get\_Item(1, &SurfBody9);

Model10->get\_SurfaceBodies(&SurfBodies10); SurfBodies10->get\_Item(1, &SurfBody10);

Edges\* oEdges1, \* oEdges2; EdgePtr oEdge1, oEdge2;

SurfBody1->get\_Edges(&oEdges1); oEdges1->get\_Item(110, &oEdge1);

SurfBody2->get\_Edges(&oEdges2); oEdges2->get\_Item(12, &oEdge2);

MateConstraintPtr sPlane1;

sPlane1 = pAsmComDef->Constraints->MethodAddInsertConstraint(oEdge1, oEdge2, true, 0);

//корпус крышка лево

Edges\* oEdges3, \* oEdges4; EdgePtr oEdge3, oEdge4;

SurfBody1->get\_Edges(&oEdges3); oEdges3->get\_Item(109, &oEdge3);

SurfBody3->get\_Edges(&oEdges4); oEdges4->get\_Item(12, &oEdge4);

MateConstraintPtr sPlane2;

sPlane2 = pAsmComDef->Constraints->MethodAddInsertConstraint(oEdge3, oEdge4, true, 0);

// совмещение плоскостей верхний отсек+корпус

Faces\* oFaces5, \* oFaces6; FacePtr oFace5, oFace6;

SurfBody1->get\_Faces(&oFaces5); oFaces5->get\_Item(4, &oFace5); SurfBody4->get\_Faces(&oFaces6);

oFaces6->get\_Item(30, &oFace6); MateConstraintPtr sPlane3;

sPlane3 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace5, oFace6, true, kNoInference, kNoInference);

Faces\* oFaces7, \* oFaces8; FacePtr oFace7, oFace8; SurfBody1->get\_Faces(&oFaces7);

oFaces7->get\_Item(2, &oFace7); SurfBody4->get\_Faces(&oFaces8);

oFaces8->get\_Item(6, &oFace8); MateConstraintPtr sPlane4;

sPlane4 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace7, oFace8, true, kNoInference, kNoInference);

Faces\* oFaces9, \* oFaces10; FacePtr oFace9, oFace10; SurfBody1->get\_Faces(&oFaces9);

oFaces9->get\_Item(60, &oFace9); SurfBody4->get\_Faces(&oFaces10); oFaces10->get\_Item(33, &oFace10);

MateConstraintPtr sPlane5;

sPlane5 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace9, oFace10, razz, kNoInference, kNoInference);

//нижний отсек+корпус

Faces\* oFaces11, \* oFaces12; FacePtr oFace11, oFace12;

SurfBody1->get\_Faces(&oFaces11); oFaces11->get\_Item(5, &oFace11);

SurfBody5->get\_Faces(&oFaces12); oFaces12->get\_Item(39, &oFace12); MateConstraintPtr sPlane6;

sPlane6 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace11, oFace12, true, kNoInference, kNoInference);

Faces\* oFaces13, \* oFaces14; FacePtr oFace13, oFace14; SurfBody1->get\_Faces(&oFaces13);

oFaces13->get\_Item(3, &oFace13);SurfBody5->get\_Faces(&oFaces14);

oFaces14->get\_Item(15, &oFace14); MateConstraintPtr sPlane7;

sPlane7 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace13, oFace14, true, kNoInference, kNoInference);

Faces\* oFaces15, \* oFaces16; FacePtr oFace15, oFace16;

SurfBody1->get\_Faces(&oFaces15); oFaces15->get\_Item(55, &oFace15);

SurfBody5->get\_Faces(&oFaces16); oFaces16->get\_Item(41, &oFace16);

MateConstraintPtr sPlane8;

sPlane8 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace15, oFace16, razz, kNoInference, kNoInference);

//нога + нижний отсек

Faces\* oFaces17, \* oFaces18; FacePtr oFace17, oFace18; SurfBody5->get\_Faces(&oFaces17);

oFaces17->get\_Item(3, &oFace17); SurfBody6->get\_Faces(&oFaces18); oFaces18->get\_Item(8, &oFace18);

MateConstraintPtr sPlane9;

sPlane9 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace17, oFace18, false, kNoInference, kNoInference);

//крышка+ногаа право

Edges\* oEdges19, \* oEdges20; EdgePtr oEdge19, oEdge20; SurfBody2->get\_Edges(&oEdges19);

oEdges19->get\_Item(9, &oEdge19); SurfBody6->get\_Edges(&oEdges20);

oEdges20->get\_Item(14, &oEdge20); MateConstraintPtr sPlane10;

sPlane10 = pAsmComDef->Constraints->MethodAddInsertConstraint(oEdge19, oEdge20, false, 0);

//крышка+нога лево

Edges\* oEdges21, \* oEdges22; EdgePtr oEdge21, oEdge22; SurfBody3->get\_Edges(&oEdges21);

oEdges21->get\_Item(9, &oEdge21); SurfBody7->get\_Edges(&oEdges22);

oEdges22->get\_Item(14, &oEdge22); MateConstraintPtr sPlane11;

sPlane11 = pAsmComDef->Constraints->MethodAddInsertConstraint(oEdge21, oEdge22, false, 0);

//муфта+крышка

Edges\* oEdges23, \* oEdges24; EdgePtr oEdge23, oEdge24; SurfBody2->get\_Edges(&oEdges23);

oEdges23->get\_Item(1, &oEdge23); SurfBody8->get\_Edges(&oEdges24);

oEdges24->get\_Item(14, &oEdge24); MateConstraintPtr sPlane12;

sPlane12 = pAsmComDef->Constraints->MethodAddInsertConstraint(oEdge23, oEdge24,true, 0);

//муфта опора

Edges\* oEdges25, \* oEdges26; EdgePtr oEdge25, oEdge26; SurfBody9->get\_Edges(&oEdges25); //опора

oEdges25->get\_Item(29, &oEdge25); SurfBody8->get\_Edges(&oEdges26); //муфта

oEdges26->get\_Item(1, &oEdge26); MateConstraintPtr sPlane13;

sPlane13 = pAsmComDef->Constraints->MethodAddInsertConstraint(oEdge25, oEdge26, true, 0);

//балкa опора

Faces\* oFaces27, \* oFaces28; FacePtr oFace27, oFace28;

SurfBody9->get\_Faces(&oFaces27); oFaces27->get\_Item(10, &oFace27);

SurfBody10->get\_Faces(&oFaces28); oFaces28->get\_Item(9, &oFace28); MateConstraintPtr sPlane14;

sPlane14 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace27, oFace28, -5.0f , kNoInference, kNoInference);

Faces\* oFaces29, \* oFaces30; FacePtr oFace29, oFace30; SurfBody9->get\_Faces(&oFaces29);

oFaces29->get\_Item(12, &oFace29); SurfBody10->get\_Faces(&oFaces30);

oFaces30->get\_Item(2, &oFace30); MateConstraintPtr sPlane15;

sPlane15 = pAsmComDef->Constraints->MethodAddMateConstraint(oFace29, oFace30, -75.0f, kNoInference, kNoInference);

//угловая зависимость

Faces\* oFaces31, \* oFaces32, \* oFaces33, \* oFaces34; FacePtr oFace31, oFace32, oFace33, oFace34;

SurfBody5->get\_Faces(&oFaces31); oFaces31->get\_Item(38, &oFace31);

SurfBody6->get\_Faces(&oFaces32); oFaces32->get\_Item(6, &oFace32);

SurfBody7->get\_Faces(&oFaces33); oFaces33->get\_Item(6, &oFace33);

SurfBody9->get\_Faces(&oFaces34); oFaces34->get\_Item(6, &oFace34);

MateConstraintPtr sPlane16; MateConstraintPtr sPlane17; MateConstraintPtr sPlane18;

sPlane16 = pAsmComDef->Constraints->MethodAddAngleConstraint(oFace31, oFace32, 0, kDirectedSolution);

sPlane17 = pAsmComDef->Constraints->MethodAddAngleConstraint(oFace31, oFace33, 0, kDirectedSolution);

sPlane18 = pAsmComDef->Constraints->MethodAddAngleConstraint(oFace31, oFace34, 0, kDirectedSolution);

}