Федеральное государственное автономное образовательное учреждение высшего образования
«Московский политехнический университет»

Кафедра “СМАРТ технологий”

Расчетно-пояснительная записка

«Программирование в САПР на языке С++»

Дисциплина “Программирование в САПР”

Студент: Киселёва А. Н.

Группа: 211-324

Вариант _18_

Москва 2022

Оглавление

1. ПОСТАНОВКА ЗАДАЧИ

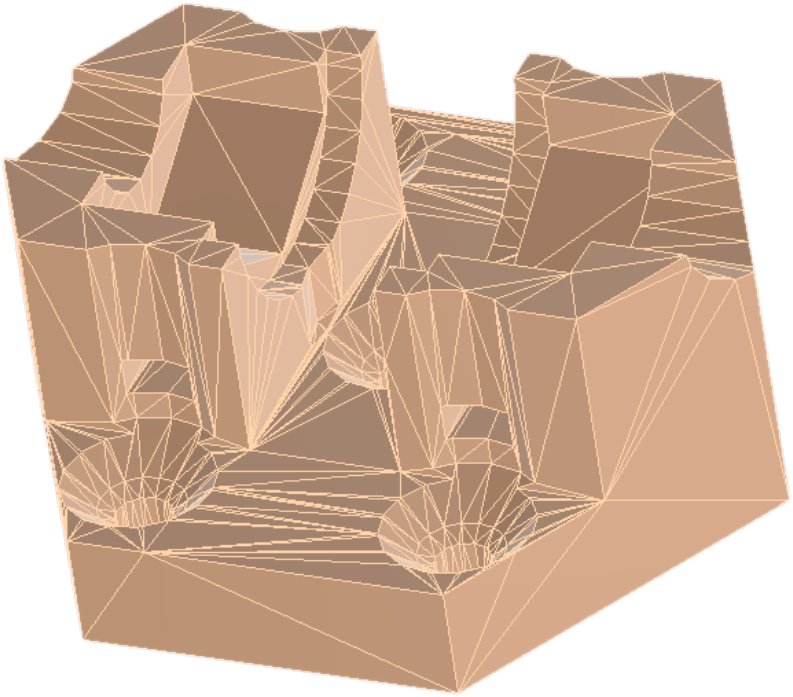


Рис. 1

Задание. Вариант 17

Курсовой проект по дисциплине «Программирование в САПР» на первом семестре ее изучения состоит в разработке конфигуратора машиностроительного изделия на основании ранее изученного материала по использованию API Autodesk Inventor и построению MFC Dialog Bases Application.

Суть проекта заключается в написании приложения, позволяющего производить параметрическое построение машиностроительной детали в Autodesk Inventor.

2. ОПИСАНИЕ КОМАНД И ОПЕРАТОРОВ

ExtrudeFeaturePtr extrude = ftExtrude->MethodAddByDistanceExtent(pProfile, H1/10, kPositiveExtentDirection, kJoinOperation) – функция выдавливания. Функция принимает профиль для выдавливания (struct Profile * Profile), расстояние выдавливания (const _variant_t & Distance), направление выдавливания (enum PartFeatureExtentDirectionEnum ExtentDirection), тип операции (enum PartFeatureOperationEnum Operation)

CircularPatternFeature* circFeat = pCircPatFeat->MethodAdd(pCollection, wax->GetItem(2), true, 8, "360 град", true, kIdenticalCompute) – функция задания кругового массива. Функция принимает следующие параметры: коллекция элементов для массива (`ObjectCollection* pCollection`), номер оси вокруг которой будут создаваться элементы массива(`wax->GetItem(n)`), где n – номер оси, 8 – количество создаваемых объектов, “360 град” – угол на котором будет создаваться круговой массив.

ChamferFeaturePtr chamFeature = pChamferFt->MethodAddUsingDistance(edgeColl, faska, false, false, false); - функция задания фаски.

3.СПЕЦИФИКАЦИЯ

3.1. Краткая характеристика программы

1. Автор: Киселёва А.Н., студент группы: 211-324
2. Дата создания: 25 мая 2022
3. Программа разработана в среде Microsoft Visual Studio

3.2. Определение перечня параметров

Разработка конфигуратора начинается с определения перечня параметров конфигурируемой детали. В случае варианта 18 было выбрано 2 параметра. На рисунке 2 представлены параметры построения для данной детали

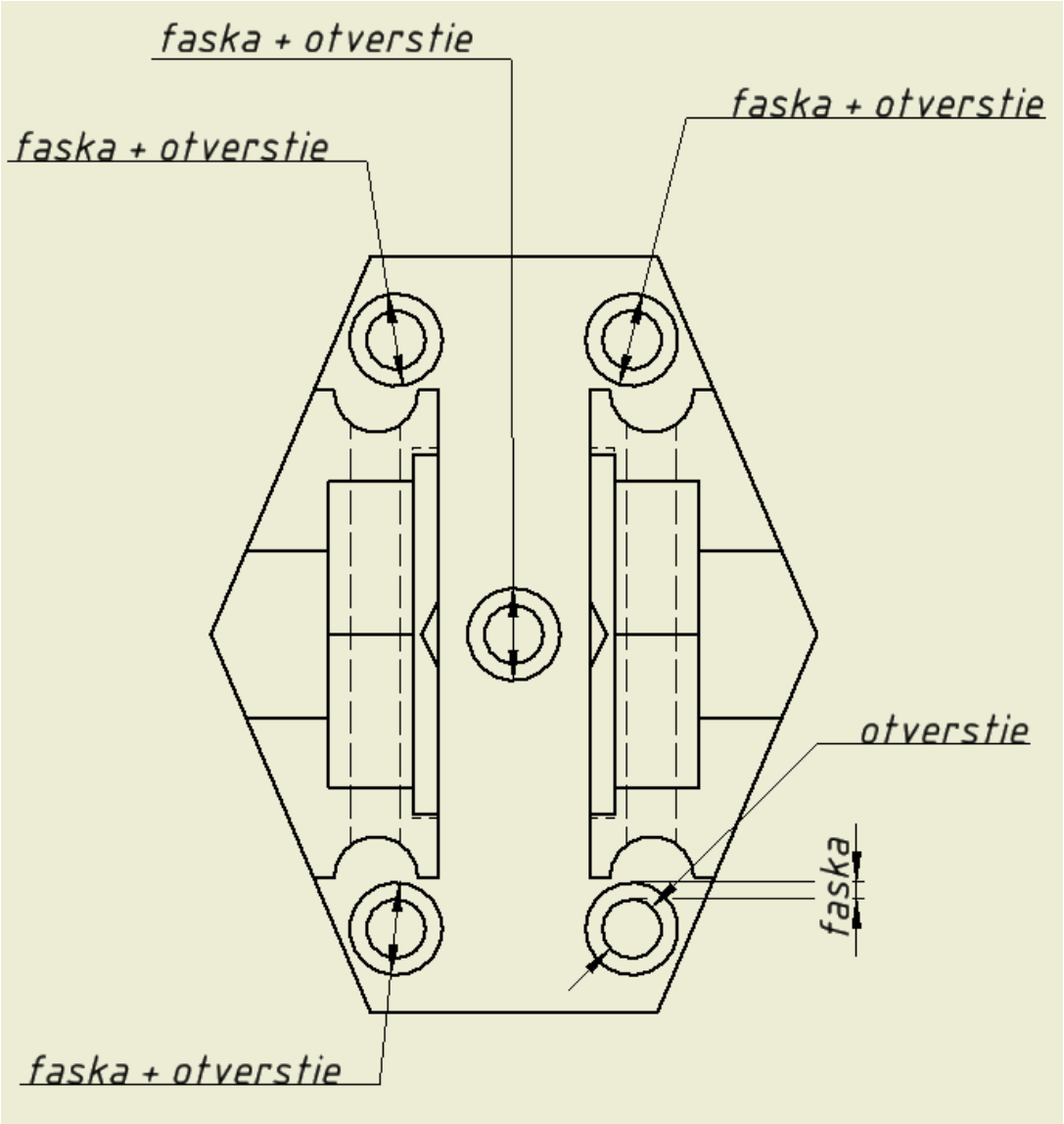


Рис. 2

3.3. Разработка MFC Dialog Based Application

После определения параметров детали приступаем к проектированию приложения – конфигуратора. Для этого использовалось MFC Dialog Based Application из Application Wizard Microsoft Visual Studio

После создания каркаса приступаем к созданию пользовательского интерфейса (Рис. 3). При проектировании было использовано некоторое количество графических элементов «Static text», «Edit Control», «Group Box», «Edit Control», «Picture Control»

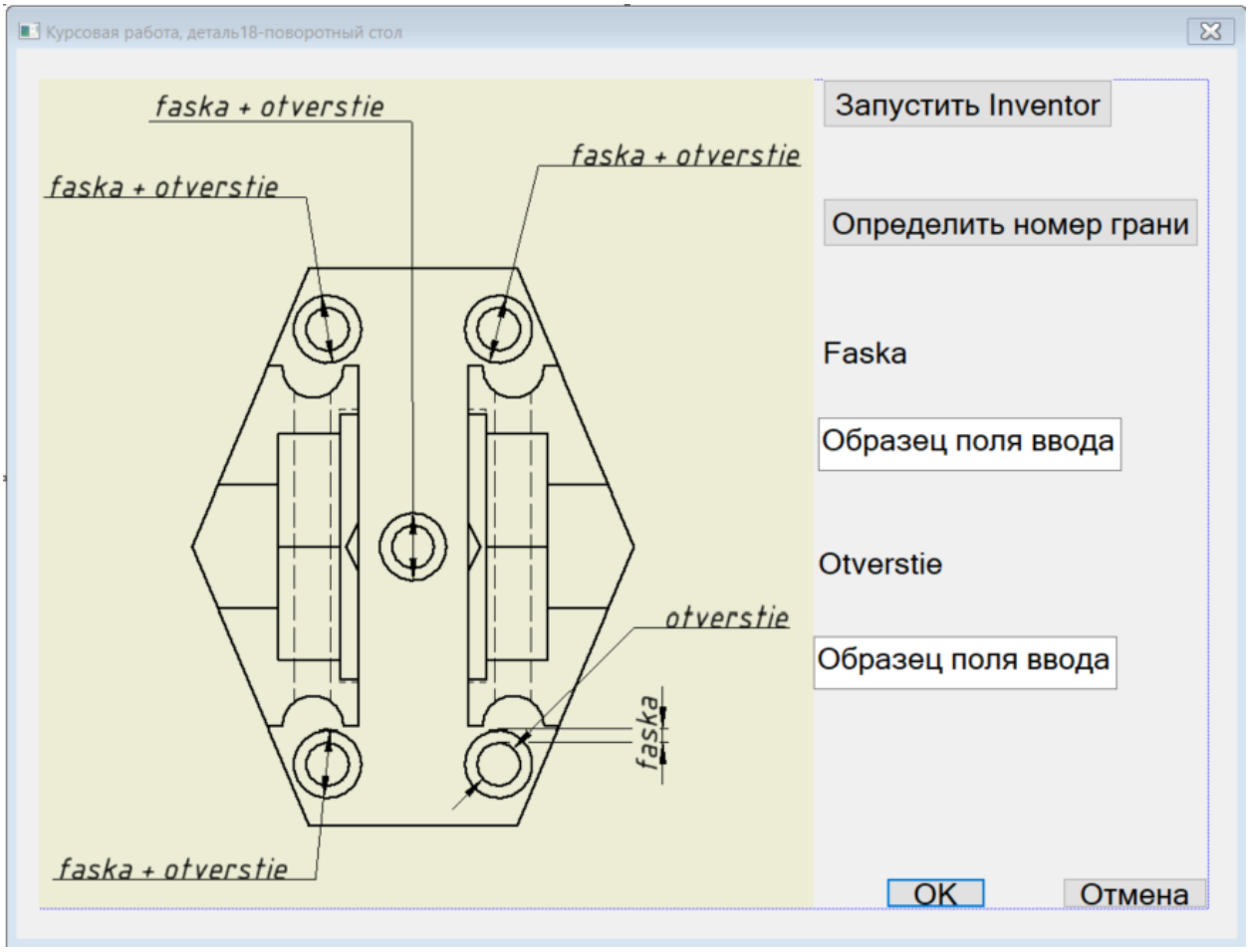


Рис. 3

В программе были созданы переменные и связаны с «Edit Control»-ми. Также вставлено изображение с названиями параметров для задания размеров детали. Все размеры должны указываться пользователем в сантиметрах.

Для работы с Autodesk Inventor необходимо подключить библиотеку типов RxInventor.tlb, которая входит в комплект его поставки и находится в установочной директории. Подключение библиотеки происходит с помощью следующего кода:

Необходимость переименования некоторых методов связана их

```
#import “\RxInventor.tlb”\
rename_namespace("InventorNative") \
named_guids raw_dispinterfaces \
high_method_prefix("Method") \
rename("DeleteFile", "APIDeleteFile") \
rename("CopyFile", "APICopyFile") \
rename("MoveFile", "APIMoveFile") \
rename("SetEnvironmentVariable", "APISetEnvironmentVariable") \
rename("GetObject", "APIGetObject") \
exclude("_FILETIME", "IStream", "ISequentialStream", \
"_LARGE_INTEGER", "_ULARGE_INTEGER", "tagSTATSTG", \
"IEnumUnknown", "IPersistFile", "IPersist", "IPictureDisp")


using namespace InventorNative;
```

«пересечением» с функциями Windows API.

Перед переходом к построениям необходимо запустить Autodesk Inventor и

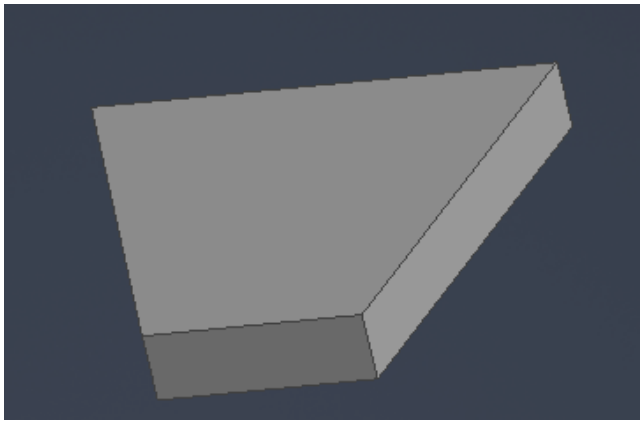
```
	CComPtr<Application> pInvApp = nullptr;
	CLSID InvAppClsid;
	HRESULT hRes = CLSIDFromProgID(L"Inventor.Application", &InvAppClsid);

	if (FAILED(hRes))
	{
		pInvApp = nullptr;
		return pInvApp;
	}
	CComPtr<IUnknown> pInvAppUnk = nullptr;
	hRes = ::GetActiveObject(InvAppClsid, NULL, &pInvAppUnk);
	if (FAILED(hRes))
	{
		hRes = CoCreateInstance(InvAppClsid, NULL, CLSCTX_LOCAL_SERVER,
__uuidof(IUnknown), (void**)&pInvAppUnk);

		if (FAILED(hRes))
		{
			pInvApp = nullptr;
			return pInvApp;
		}
	}
	hRes = pInvAppUnk->QueryInterface(__uuidof(Application),
(void**)&pInvApp);
	pInvApp->put_Visible(TRUE);
	return pInvApp;
```

инициализировать некоторые «главные» глобальные указатели, которые понадобятся нам в дальнейшем.

Построение первой геометрии

```
PlanarSketchPtr pSketch;
	sketches->raw_Add(wp->GetItem(3), false, &pSketch);
	//контейнеры

	SketchPointsPtr skPoints;
	SketchLinesPtr skLines;
	SketchCirclesPtr skCircles;
	SketchArcsPtr skArcs;
	ProfilesPtr skProfiles;

	pSketch->get_SketchPoints(&skPoints);
	pSketch->get_SketchLines(&skLines);
	pSketch->get_Profiles(&skProfiles);

	//13 точек
	SketchPointPtr point[4];

	//16 линий
	SketchLinePtr lines[4];

	point[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);
	point[1] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-36, 0), false);
	point[2] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-17, -45), false);
	point[3] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, -45), 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]);

	ProfilePtr pProfile;
	try
	{
		pProfile = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude;
	ft->get_ExtrudeFeatures(&ftExtrude);
	ExtrudeFeaturePtr extrude = ftExtrude->MethodAddByDistanceExtent(pProfile, 5,
kPositiveExtentDirection, kJoinOperation);
	wp->GetItem(3)->PutVisible(false);
```

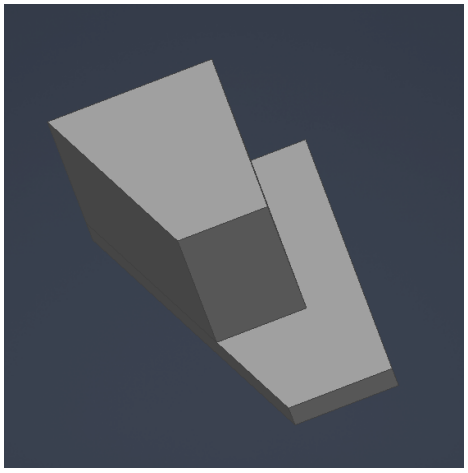
Результат:



Построение второй геометрии

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(3), 5, false);

	PlanarSketchPtr pSketch1;
	sketches->raw_Add(wp->GetItem(4), false, &pSketch1);

	pSketch1->get_SketchPoints(&skPoints);
	pSketch1->get_SketchLines(&skLines);
	pSketch1->get_Profiles(&skProfiles);

	SketchPointPtr point1[4];
	SketchLinePtr lines1[4];

	point1[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-9, 0),
false);
	point1[1] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-36, 0),
false);
	point1[2] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-23.756, -
29), false);
	point1[3] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-9, -29),
false);

	lines1[0] = skLines->MethodAddByTwoPoints(point1[0], point1[1]);
	lines1[1] = skLines->MethodAddByTwoPoints(point1[1], point1[2]);
	lines1[2] = skLines->MethodAddByTwoPoints(point1[2], point1[3]);
	lines1[3] = skLines->MethodAddByTwoPoints(point1[3], point1[0]);

	ProfilePtr pProfile1;
	try
	{
		pProfile1 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude1;
	ft->get_ExtrudeFeatures(&ftExtrude1);
	ExtrudeFeaturePtr extrude1 = ftExtrude1->MethodAddByDistanceExtent(pProfile1,
30, kPositiveExtentDirection, kJoinOperation);
	wp->GetItem(4)->PutVisible(false);
```

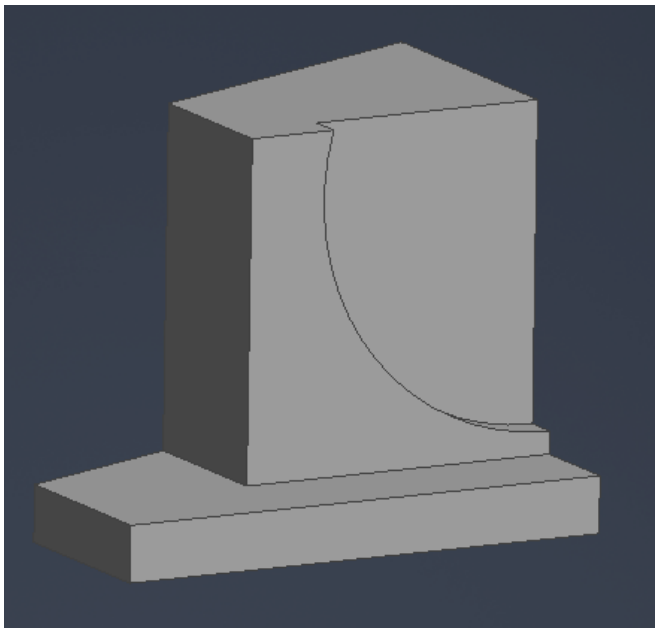
Результат:



Выдавливания дуги на второй выдавленной геометри:

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(1), -9, false);

	PlanarSketchPtr pSketch2;
	sketches->raw_Add(wp->GetItem(5), false, &pSketch2);

	pSketch2->get_SketchPoints(&skPoints);
	pSketch2->get_Profiles(&skProfiles);
	pSketch2->get_SketchCircles(&skCircles);
	//pSketch->get_SketchArcs(&skArcs);

	//13 точек
	SketchPointPtr point2[1];
	//9 линий
	//2 окружности
	SketchCirclePtr circ2[1];

	point2[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 29),
false);
	circ2[0] = skCircles->MethodAddByCenterRadius(point2[0], 22);

	ProfilePtr pProfile2;
	try
	{
		pProfile2 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude2;
	ft->get_ExtrudeFeatures(&ftExtrude2);

	ExtrudeFeaturePtr extrude2 = ftExtrude2->MethodAddByDistanceExtent(pProfile2,
3, kNegativeExtentDirection, kCutOperation);
	wp->GetItem(5)->PutVisible(false);
```

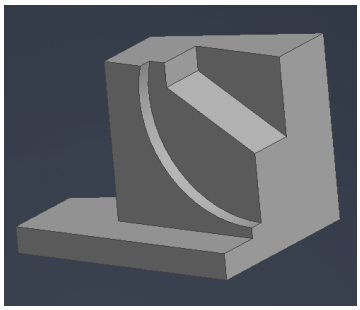
Результат:



Выдавливание шестиугольной геометрии (вырезание):

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(1), -12, false);

	PlanarSketchPtr pSketch3;
	sketches->raw_Add(wp->GetItem(6), false, &pSketch3);

	pSketch3->get_SketchPoints(&skPoints);
	pSketch3->get_SketchLines(&skLines);
	pSketch3->get_Profiles(&skProfiles);

	//13 точек
	SketchPointPtr point3[4];

	//16 линий
	SketchLinePtr lines3[4];

	point3[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 35),
false);
	point3[1] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-18, 35),
false);
	point3[2] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-18, 35-
4.609), false);
	point3[3] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 20),
false);


	lines3[0] = skLines->MethodAddByTwoPoints(point3[0], point3[1]);
	lines3[1] = skLines->MethodAddByTwoPoints(point3[1], point3[2]);
	lines3[2] = skLines->MethodAddByTwoPoints(point3[2], point3[3]);
	lines3[3] = skLines->MethodAddByTwoPoints(point3[3], point3[0]);


	ProfilePtr pProfile3;
	try
	{
		pProfile3 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude3;
	ft->get_ExtrudeFeatures(&ftExtrude3);

	ExtrudeFeaturePtr extrude3 = ftExtrude3->MethodAddByDistanceExtent(pProfile3,
10, kNegativeExtentDirection, kCutOperation);
	wp->GetItem(6)->PutVisible(false);
```

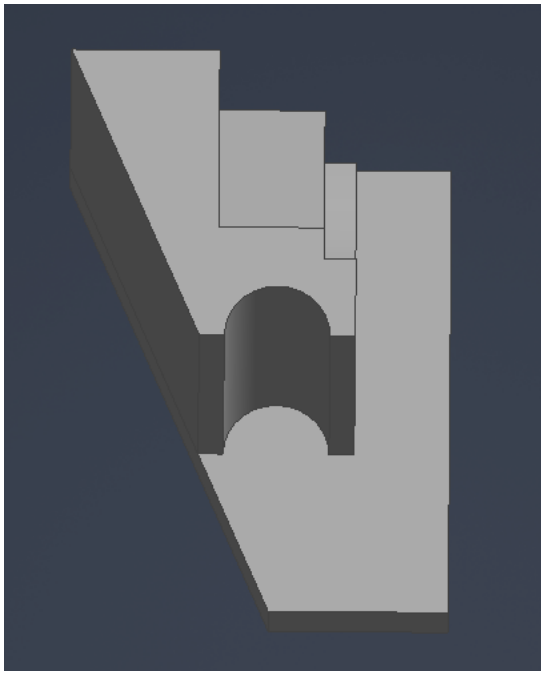
Результат:



Выдавливание отверстия до 1 геометрии:

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(3), 30+5, false);

	PlanarSketchPtr pSketch4;
	sketches->raw_Add(wp->GetItem(7), false, &pSketch4);

	pSketch4->get_SketchPoints(&skPoints);
	pSketch4->get_Profiles(&skProfiles);
	pSketch4->get_SketchCircles(&skCircles);
	//pSketch->get_SketchArcs(&skArcs);

	//13 точек
	SketchPointPtr point4[1];
	//9 линий
	//2 окружности
	SketchCirclePtr circ4[1];

	point4[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-16.378, -
29), false);
	circ4[0] = skCircles->MethodAddByCenterRadius(point4[0], 5);

	ProfilePtr pProfile4;
	try
	{
		pProfile4 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude4;
	ft->get_ExtrudeFeatures(&ftExtrude4);

	ExtrudeFeaturePtr extrude4 = ftExtrude4->MethodAddByDistanceExtent(pProfile4,
30, kNegativeExtentDirection, kCutOperation);
	wp->GetItem(7)->PutVisible(false);
```

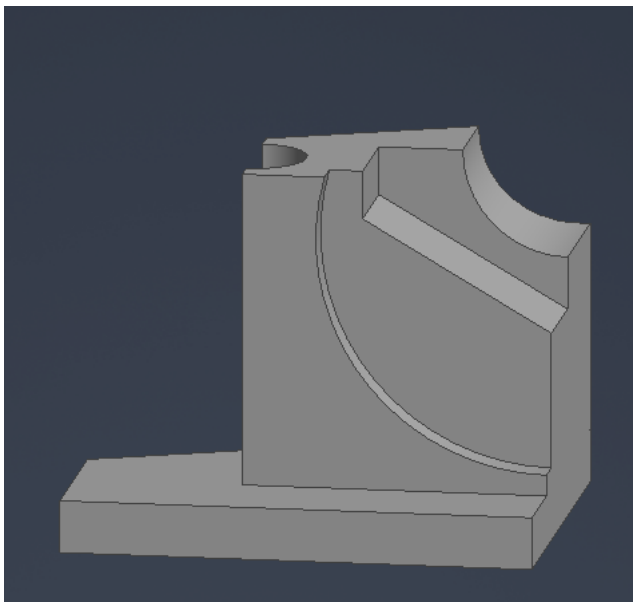
Результат:



Выдавливание дуги на угле геометрии 2:

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(1), -22, false);
	//---------------------------------------9999999999999999999999999--------
-----------------------------------------------------
	PlanarSketchPtr pSketch5;
	sketches->raw_Add(wp->GetItem(8), false, &pSketch5);

	pSketch5->get_SketchPoints(&skPoints);
	pSketch5->get_Profiles(&skProfiles);
	pSketch5->get_SketchCircles(&skCircles);
	//pSketch->get_SketchArcs(&skArcs);

	//13 точек
	SketchPointPtr point5[1];
	//9 линий
	//2 окружности
	SketchCirclePtr circ5[1];

	point5[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 35),
false);
	circ5[0] = skCircles->MethodAddByCenterRadius(point5[0], 10);

	ProfilePtr pProfile5;
	try
	{
		pProfile5 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude5;
	ft->get_ExtrudeFeatures(&ftExtrude5);

	ExtrudeFeaturePtr extrude5 = ftExtrude5->MethodAddByDistanceExtent(pProfile5,
37, kNegativeExtentDirection, kCutOperation);
	wp->GetItem(8)->PutVisible(false);
```

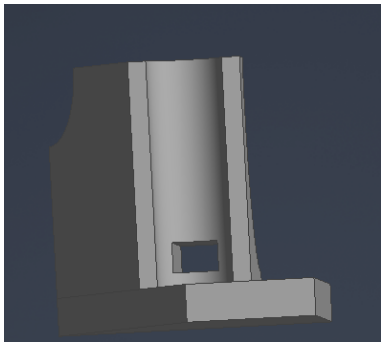
Результат:



Выдавливание прямоугольного отверстия квозного:

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(2), -29, false);
		PlanarSketchPtr pSketch6;
	sketches->raw_Add(wp->GetItem(9), false, &pSketch6);

	pSketch6->get_SketchPoints(&skPoints);
	pSketch6->get_Profiles(&skProfiles);
	//pSketch->get_SketchArcs(&skArcs);
	pSketch6->get_SketchLines(&skLines);

	SketchPointPtr point6[4];

	SketchLinePtr lines6[4];

	point6[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(13.378, 11),
false);
	point6[1] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(13.378+6,
11), false);
	point6[2] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(13.378+6, 7),
false);
	point6[3] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(13.378, 7),
false);


	lines6[0] = skLines->MethodAddByTwoPoints(point6[0], point6[1]);
	lines6[1] = skLines->MethodAddByTwoPoints(point6[1], point6[2]);
	lines6[2] = skLines->MethodAddByTwoPoints(point6[2], point6[3]);
	lines6[3] = skLines->MethodAddByTwoPoints(point6[3], point6[0]);

	point6[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 35),
false);

	ProfilePtr pProfile6;
	try
	{
		pProfile6 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude6;
	ft->get_ExtrudeFeatures(&ftExtrude6);

	ExtrudeFeaturePtr extrude6 = ftExtrude6->MethodAddByDistanceExtent(pProfile6,
100, kPositiveExtentDirection, kCutOperation);
	wp->GetItem(9)->PutVisible(false);
```

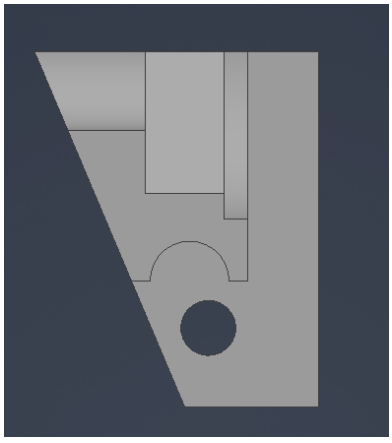
Результат:



Сквозное отверстие на 1 выдавленной геометрии:

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(3), 5, false);
                PlanarSketchPtr pSketch7;
        sketches->raw_Add(wp->GetItem(10), false, &pSketch7);

        pSketch7->get_SketchPoints(&skPoints);
        pSketch7->get_Profiles(&skProfiles);
        pSketch7->get_SketchCircles(&skCircles);
        //pSketch->get_SketchArcs(&skArcs);

        //13 точек
        SketchPointPtr point7[1];
        //9 линий
        //2 окружности
        SketchCirclePtr circ7[1];
        if (otverstie + faska + 2 > 12)
        {
                otverstie = 7;
                faska = 2;
        }
        else if (otverstie < 0)
        {
                otverstie = 7;
        }
        else if (faska < 0)
        {
                faska = 2;
                if (otverstie + faska + 2 > 12)
                {
                        otverstie = 7;
                        faska = 2;
                }
        }
        point7[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-14, -35), false);
        circ7[0] = skCircles->MethodAddByCenterRadius(point7[0], otverstie/2);

        ProfilePtr pProfile7;
        try
        {
                pProfile7 = skProfiles->MethodAddForSolid(true);
        }
        catch (...)
        {
                AfxMessageBox(L"Ошибочный контур!");
                return;
        }
        ExtrudeFeaturesPtr ftExtrude7;
        ft->get_ExtrudeFeatures(&ftExtrude7);

        ExtrudeFeaturePtr extrude7 = ftExtrude7->MethodAddByDistanceExtent(pProfile7, 5,
kNegativeExtentDirection, kCutOperation);
        wp->GetItem(10)->PutVisible(false);
```

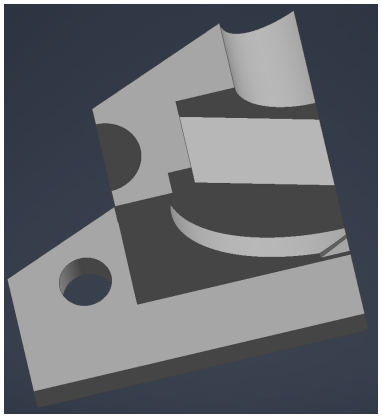
Результат:



Выдавливание маленького треугольника на плоскости отдаленной и параллельной от XY на 1 мм:

```
wp->MethodAddByPlaneAndOffset(wp->GetItem(3), 6, false);

	PlanarSketchPtr pSketch8;
	sketches->raw_Add(wp->GetItem(11), false, &pSketch8);

	pSketch8->get_SketchPoints(&skPoints);
	pSketch8->get_Profiles(&skProfiles);
	//pSketch->get_SketchArcs(&skArcs);
	pSketch8->get_SketchLines(&skLines);

	SketchPointPtr point8[3];

	SketchLinePtr lines8[3];

	point8[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-9, 0),
false);
	point8[1] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-11, 0),
false);
	point8[2] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(-9, -4),
false);


	lines8[0] = skLines->MethodAddByTwoPoints(point8[0], point8[1]);
	lines8[1] = skLines->MethodAddByTwoPoints(point8[1], point8[2]);
	lines8[2] = skLines->MethodAddByTwoPoints(point8[2], point8[0]);

	ProfilePtr pProfile8;
	try
	{
		pProfile8 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude8;
	ft->get_ExtrudeFeatures(&ftExtrude8);

	ExtrudeFeaturePtr extrude8 = ftExtrude8->MethodAddByDistanceExtent(pProfile8,
5, kPositiveExtentDirection, kCutOperation);
	wp->GetItem(11)->PutVisible(false);
```

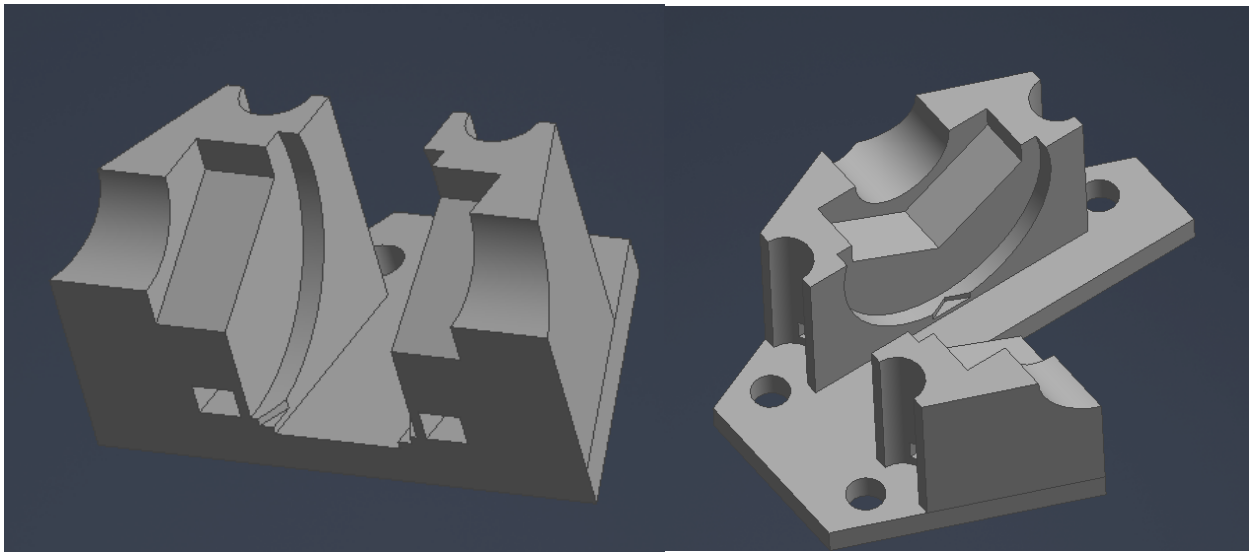
Результат:



Отзеркаливание всей предыдущей геометрии относительно плоскости YZ и XZ:

```
ObjectCollectionPtr objectColl;
     inventorApp->TransientObjects-
>raw_CreateObjectCollection(vtMissing, &objectColl);
     // в коллекцию элементов
     objectColl->MethodAdd(extrude);
     objectColl->MethodAdd(extrude1);
     objectColl->MethodAdd(extrude2);
     objectColl->MethodAdd(extrude3);
     objectColl->MethodAdd(extrude4);
     objectColl->MethodAdd(extrude5);
     objectColl->MethodAdd(extrude6);
     objectColl->MethodAdd(extrude7);
     objectColl->MethodAdd(extrude8);
     //objectColl->MethodAdd(pChamferFt);

     MirrorFeatures* pMirrorFeatures;
     ft->get_MirrorFeatures(&pMirrorFeatures);
     MirrorFeature* mirrorFeat = pMirrorFeatures-
>MethodAdd(objectColl, wp->GetItem(1), false, kIdenticalCompute);
     mirrorFeat = pMirrorFeatures->MethodAdd(objectColl, wp-
>GetItem(2), false, kIdenticalCompute);
```

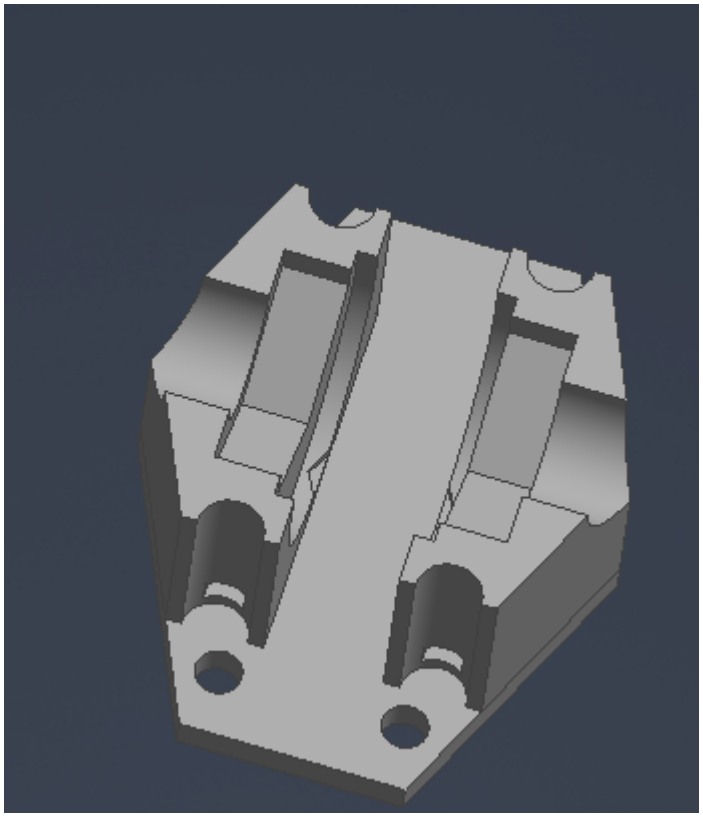
Результат:



Круговой массив на 180 градусов:

```
	ObjectCollection* pCollection;

	inventorApp->TransientObjects->raw_CreateObjectCollection(vtMissing,
&pCollection);

	pCollection->MethodAdd(extrude);
	pCollection->MethodAdd(extrude1);
	pCollection->MethodAdd(extrude2);
	pCollection->MethodAdd(extrude3);
	pCollection->MethodAdd(extrude4);
	pCollection->MethodAdd(extrude5);
	pCollection->MethodAdd(extrude6);
	pCollection->MethodAdd(extrude7);
	pCollection->MethodAdd(extrude8);
	//pCollection->MethodAdd(pChamferFt);
	CircularPatternFeatures* pCircPatFeat;

	ft->get_CircularPatternFeatures(&pCircPatFeat);

	CircularPatternFeaturePtr circFeat1 = pCircPatFeat->MethodAdd(pCollection,
wax->GetItem(3), true, 2, "180 град", true, kIdenticalCompute);
```

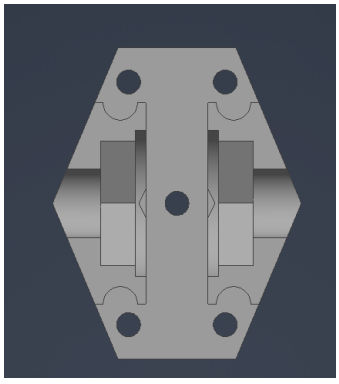
Результат:



На плоскости XY создаем по середине детали отверстие того же размера что и предыдущее:

```
PlanarSketchPtr pSketch9;
	sketches->raw_Add(wp->GetItem(3), false, &pSketch9);

	pSketch9->get_SketchPoints(&skPoints);
	pSketch9->get_Profiles(&skProfiles);
	pSketch9->get_SketchCircles(&skCircles);
	//pSketch->get_SketchArcs(&skArcs);

	//13 точек
	SketchPointPtr point9[1];
	//9 линий
	//2 окружности
	SketchCirclePtr circ9[1];

	point9[0] = skPoints->MethodAdd(pTransGeom->MethodCreatePoint2d(0, 0), false);
	if (otverstie + faska + 2 > 12)
	{
		otverstie = 7;
		faska = 2;
	}
	else if (otverstie < 0)
	{
		otverstie = 7;
	}
	else if (faska < 0)
	{
		faska = 2;
		if (otverstie + faska + 2 > 12)
		{
			otverstie = 7;
			faska = 2;
		}
	}
	circ9[0] = skCircles->MethodAddByCenterRadius(point9[0], otverstie / 2);

	ProfilePtr pProfile9;
	try
	{
		pProfile9 = skProfiles->MethodAddForSolid(true);
	}
	catch (...)
	{
		AfxMessageBox(L"Ошибочный контур!");
		return;
	}
	ExtrudeFeaturesPtr ftExtrude9;
	ft->get_ExtrudeFeatures(&ftExtrude9);

	ExtrudeFeaturePtr extrude9 = ftExtrude9->MethodAddByDistanceExtent(pProfile9, 5,
kPositiveExtentDirection, kCutOperation);
```

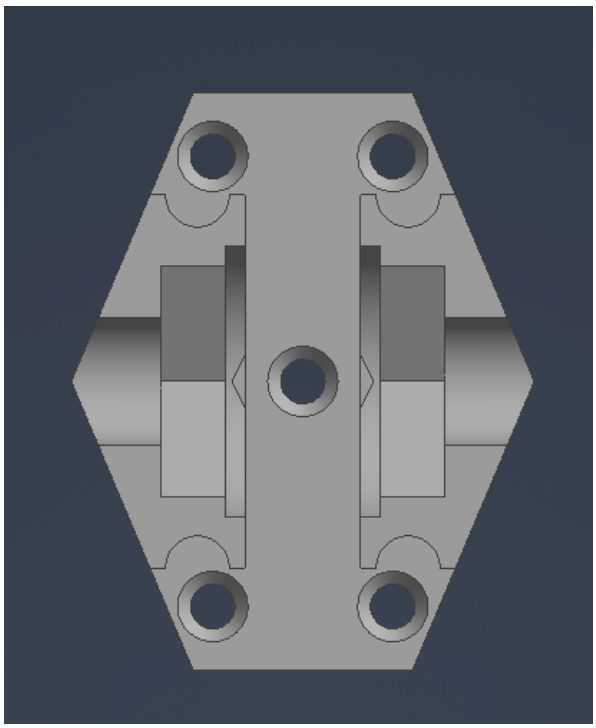
Результат:



Фаски на отверстия:

```
ChamferFeaturesPtr pChamferFt;
	ft->get_ChamferFeatures(&pChamferFt);

	EdgeCollectionPtr edgeColl;
	EdgeCollectionPtr edgeColl1;
	SurfaceBodyPtr SurfBody;
	SurfaceBodiesPtr SurfBodies;
	EdgesPtr edges;
	EdgePtr ed;
	inventorApp->TransientObjects->raw_CreateEdgeCollection(vtMissing,
&edgeColl);

	edgeColl->MethodClear();

	pPartComDef->get_SurfaceBodies(&SurfBodies);

	SurfBodies->get_Item(1, &SurfBody);

	SurfBody->get_Edges(&edges);

	edges->get_Item(1, &ed);//y
	edgeColl->MethodAdd(ed);
	edges->get_Item(49, &ed);//y
	edgeColl->MethodAdd(ed);
	edges->get_Item(48, &ed);//y
	edgeColl->MethodAdd(ed);
	edges->get_Item(50, &ed);//y
	edgeColl->MethodAdd(ed);
	edges->get_Item(23, &ed);//y
	edgeColl->MethodAdd(ed);
	ChamferFeaturePtr chamFeature = pChamferFt->MethodAddUsingDistance(edgeColl,
faska, false, false, false);
```

Результат:



Список литературы:

1. https://www.youtube.com/watch?v=jI4V6Oo5QLM
2. https://docs.microsoft.com/ru-ru/cpp/mfc/walkthroughs-mfc?view=msvc-160
3. https://coderlessons.com/tutorials/microsoft-technologies/izuchite-mfts/mfc-kratkoe-rukovodstvo