МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Нижегородский государственный университет

им. Н.И. Лобачевского»

**Факультет вычислительной математики и кибернетики**

**Кафедра: Центр прикладной информатики**

Направление: Прикладная информатика

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

Тема:

**«Название работы»**

**Допущен к защите:**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Заведующий кафедрой:**

Должность, уч. степень\_\_\_\_\_\_\_\_\_ФИО

«\_\_\_»\_\_\_\_\_\_\_\_\_\_2015 года

**Выполнил:** студент группы 8409

Дёгтев Александр Сергеевич

**Научный руководитель:**

к.т.н., доц. каф. ИАНИ

Васин Д. Ю.

Нижний Новгород

2015

# Аннотация

# Оглавление

# Глоссарий

Будем называть действительным пересечением двух объектов, пересечение, при котором у обоих объектов есть общая точка.

Будем называть мнимым пересечением двух объектов пересечение, если при продлении одного объекта у него образуется точка пересечения с другим объектом.

# Введение

В последнее время всё большее распространение получают электронные средства проектирования, в том числе и в машиностроительной отрасли. Это связано с удешивлением вычислительной техники и распространением программных продуктов, позволяющих создавать чертежи в электронном виде. Такие продукты называют САПР. САПР – система автоматизированного проектирования технических объектов, которая решает весь комплекс задач от анализа задания до разработки полного объема конструкторской и технологической документации. Это достигается за счет объединения современных технических средств и математического обеспечения, параметры и характеристики которых выбираются с максимальным учетом особенностей задач проектно-конструкторского процесса. САПР представляет собой крупные организационно-технические системы, состоящие из комплекса средств автоматизации проектирования, взаимосвязанного с подразделениями конкретной проектной организации. Примерами таких систем могут служить AutoCAD, КОМПАС-3D, CorelCAD.

Исходя из недостатков у перечисленных систем проектирования следует, что имеется необходимость в нахождении ошибок на машиностроительных чертежах, созданных в системах проектирования или оцифрованных при помощи сканера.

Из всего вышесказанного можно сделать следующий вывод, что целью данной работы является разработка информационного, алгоритмического и программного обеспечения подсистемы оценки качества формируемых документов типа машиностроительный чертёж.

# Теоритическая часть

## Обзор функциональных возможностей существующих САПР

AutoCAD – двух- и трёхмерная система автоматизированного проектирования и черчения, разработанная компанией Autodesk. Первая версия системы была выпущена в 1982 году.[4]

## Обзор некоторых существующих САПР

### AutoCAD

AutoCAD – двух- и трёхмерная система автоматизированного проектирования и черчения, разработанная компанией Autodesk. Первая версия системы была выпущена в 1982 году.[4]

#### Функциональные возможности

Ранние версии AutoCAD оперировали небольшим числом элементарных объектов, такими как круги, линии, дуги и текст, из которых составлялись более сложные. Однако на современном этапе возможности AutoCAD весьма широки.

В области двумерного проектирования AutoCAD по-прежнему позволяет использовать элементарные графические примитивы для получения более сложных объектов. Кроме того, программа предоставляет весьма обширные возможности работы со слоями и аннотативными объектами (размерами, текстом, обозначениями). Использование механизма внешних ссылок (XRef) позволяет разбивать чертеж на составные файлы, за которые ответственны различные разработчики, а динамические блоки расширяют возможности автоматизации 2D-проектирования обычным пользователем без использования программирования. В AutoCAD реализована поддержка двумерного параметрического черчения. В версии 2014 появилась возможность динамической связи чертежа с реальными картографическими данными (GeoLocation API).

В настоящее время AutoCAD включает в себя полный набор инструментов для комплексного трёхмерного моделирования (поддерживается [твердотельное](http://ru.wikipedia.org/w/index.php?title=%D0%A2%D0%B2%D0%B5%D1%80%D0%B4%D0%BE%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1), [поверхностное](http://ru.wikipedia.org/w/index.php?title=%D0%9F%D0%BE%D0%B2%D0%B5%D1%80%D1%85%D0%BD%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) и полигональное моделирование). AutoCAD позволяет получить высококачественную визуализацию моделей с помощью системы [рендеринга](http://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BD%D0%B4%D0%B5%D1%80%D0%B8%D0%BD%D0%B3) [mental ray](http://ru.wikipedia.org/wiki/Mental_ray). Также в программе реализовано управление трёхмерной печатью (результат моделирования можно отправить на [3D-принтер](http://ru.wikipedia.org/wiki/3D-%D0%BF%D1%80%D0%B8%D0%BD%D1%82%D0%B5%D1%80)) и поддержка [облаков точек](http://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D0%BF%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85#.D0.9E.D0.B1.D0.BB.D0.B0.D0.BA.D0.BE_.D1.82.D0.BE.D1.87.D0.B5.D0.BA) (позволяет работать с результатами [3D-сканирования](http://ru.wikipedia.org/wiki/3D-%D1%81%D0%BA%D0%B0%D0%BD%D0%B5%D1%80)). Тем не менее следует отметить, что отсутствие трёхмерной параметризации не позволяет AutoCAD напрямую конкурировать с машиностроительными САПР среднего класса, такими как [Inventor](http://ru.wikipedia.org/wiki/Autodesk_Inventor), [SolidWorks](http://ru.wikipedia.org/wiki/SolidWorks) и другими. В состав AutoCAD 2012 включена программа Inventor Fusion, реализующая технологию прямого моделирования.[5]

### КОМПАС-3D

Система КОМПАС-3D позволяет реализовать классический процесс трехмерного параметрического проектирования от идеи к ассоциативной объемной модели, от модели к конструкторской документации. Система предназначена для создания трехмерных ассоциативных моделей отдельных деталей и сборочных единиц, содержащих как оригинальные, так и стандартизованные конструктивные элементы. Параметрическая технология позволяет быстро получать модели типовых изделий на основе однажды спроектированного прототипа. Многочисленные сервисные функции облегчают решение вспомогательных задач проектирования и обслуживания производства

Основные компоненты КОМПАС-3D – система трехмерного твердотельного моделирования, универсальная система автоматизированного проектирования ,график и модуль проектирования спецификаций.

Базовый функционал системы включает в себя:

* средства работы над проектами, включающими подсбороки, деталей и стандартных изделий;
* функционал моделирования деталей из листового материала : команды создания листового тела, сгибов, отверстий, жалюзи, буртиков, штамповок и вырезов в листовом теле, замыкания углов и т.д., а также выполнения развертки полученного листового тела (в том числе формирования ассоциативного чертежа развертки);
* средства создания поверхностей;
* инструменты создания пользовательских параметрических библиотек типовых элементов;
* возможность получения конструкторской и технологической документации: встроенная система КОМПАС-График позволяет выпускать чертежи, сᴨȇцификации, схемы, таблицы, текстовые документы;
* возможность простановки размеров и обозначений в трехмерных моделях (поддержка стандарта ГОСТ 2.052-2006 «ЕСКД. Электронная модель изделия»);
* поддержку стандарта Unicode;
* средства интеграции с различными CAD/CAM/CAE системами;
* средства защиты пользовательских данных, интеллектуальной собственности и сведений, составляющих коммерческую и государственную тайну (реализовано отдельным программным модулем КОМПАС-Защита).

Универсальная система автоматизированного проектирования КОМПАС-График.

Для автоматизации разработки и выпуска конструкторской документации АСКОН предлагает универсальную систему автоматизированного проектирования КОМПАС-График, позволяющую в скоростном режиме выпускать чертежи изделий, схемы, спецификации, различные текстовые документы, таблицы, инструкции и прочие документы.

Система изначально ориентирована на полную поддержку стандартов ЕСКД. При этом она обладает возможностью гибкой настройки на стандарты предприятия. Средства импорта/экспорта графических документов (КОМПАС-График поддерживает форматы DXF, DWG, IGES, eDrawings) позволяют организовать обмен данными со смежниками и заказчиками, использующими любые чертежно-графические системы. Весь функционал КОМПАС-График подчинен целям скоростного создания высококачественных чертежей, схем, расчетно-пояснительных записок, технических условий, инструкций и прочих документов. [3]

## Существующие аналоги

### Autodesk Navisworks

Программа Autodesk Navisworks служит для экспертизы архитектурно-строительных проектов. С её помощью можно координировать выполняемые работы, моделировать процесс строительства и проводить комплексный анализ. В данной программе имеются инструменты моделирования и оптимизации строительных графиков, выявления коллизий и пересечений, совместной работы (посредством добавления комментариев) и обнаружения потенциальных проблем, выявления конфликтов. Имеется поддержка приложений сторонних разработчиков.

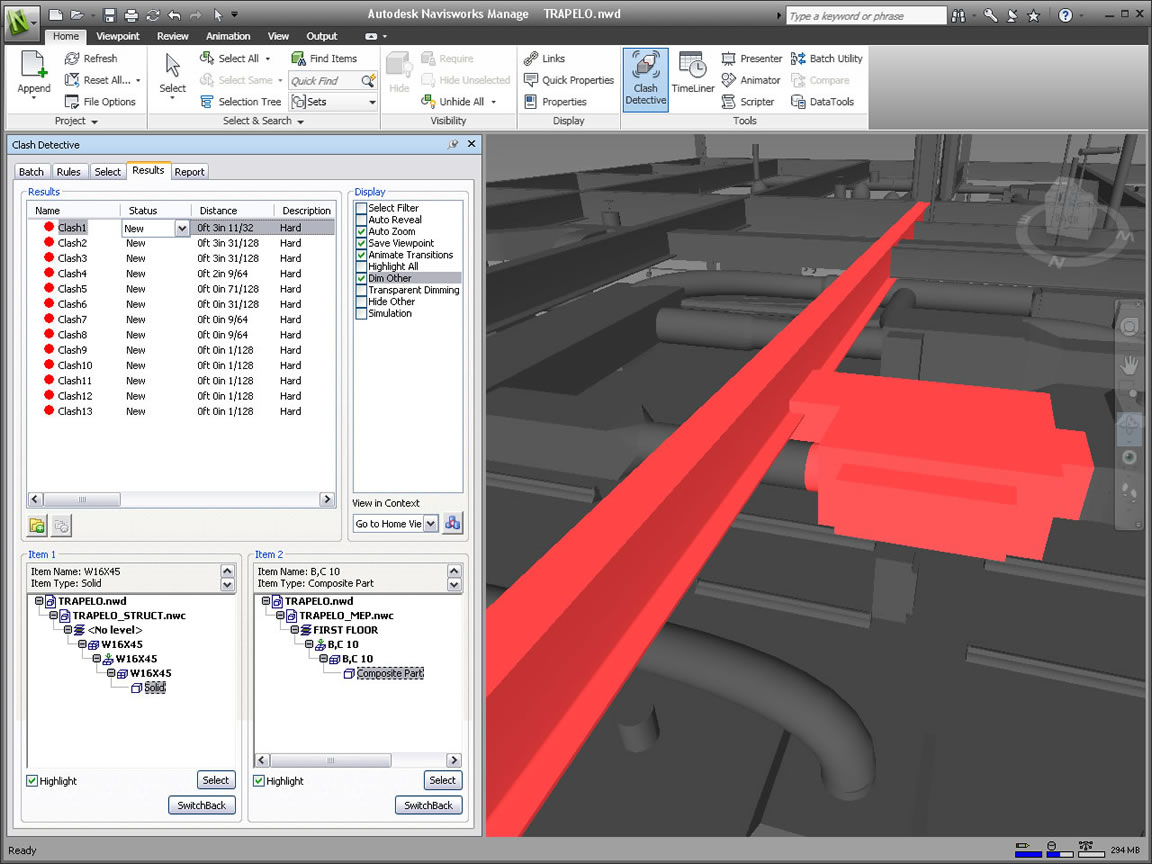


Рис. 16 Скриншот работы программы.

На рисунке 16 показано обнаружение программой Autodesk Navisworks пересечения деталей и информирование пользователя о них.

В Autodesk Navisworks используются собственные форматы файлов - .nwd, .nwf, .nwc; имеется поддержка приложений AutoCAD, Revit, 3ds Max и продуктов на их основе, а также многих из широко используемых форматов в области САПР.

Autodesk Navisworks устанавливается отдельно от AutoCAD.

Данная программа платная, но имеются демо-версии (30 дней) и версия для студентов (бесплатная лицензия, действительная в течение 3 лет).

### CorelCAD

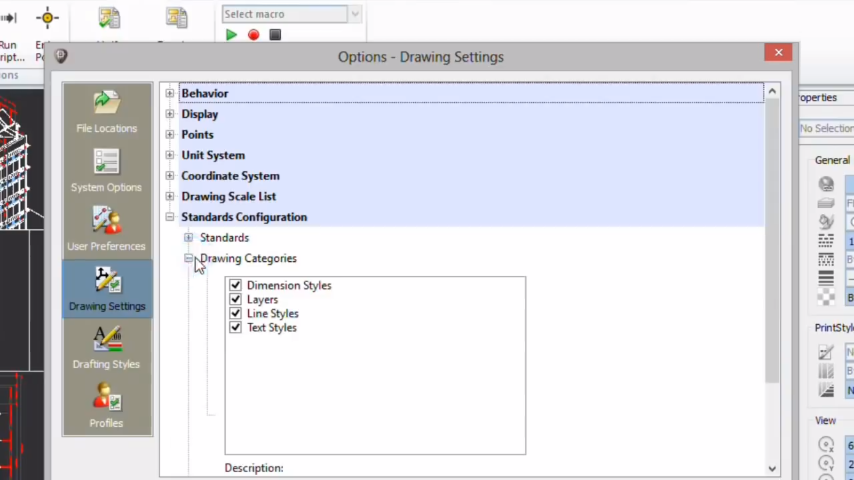
CorelCAD – это эффективное, высокопроизводительное и экономичное решение САПР для выполнения повседневных работ по проектированию, требующих высокой точности и тщательной проработки деталей [1].

Данная программа позволяет проверить чертёж и выявить нарушения отраслевых, корпоративных и проектных стандартов. Команда «Проверка стандартов» инспектирует следующие параметры на соответствие стандартам наименований из файла чертёжных стандартов:

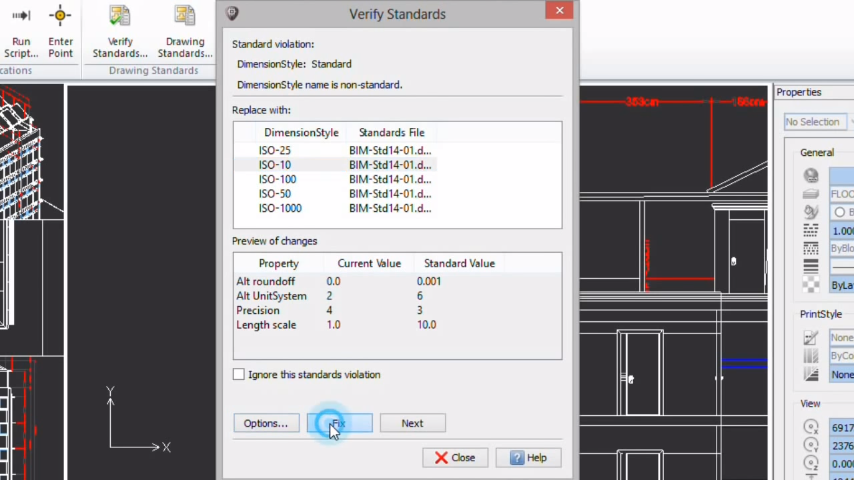
* Свойства слоя
* Свойства линий
* Размер
* Текст.

При нахождении ошибки в командном окне появляется ее описание. Данная команда создает ASCII файл с расширением .adt, который содержит отчет об обнаруженных ошибках. Файл .adt находится в одной папке с чертежом [2].

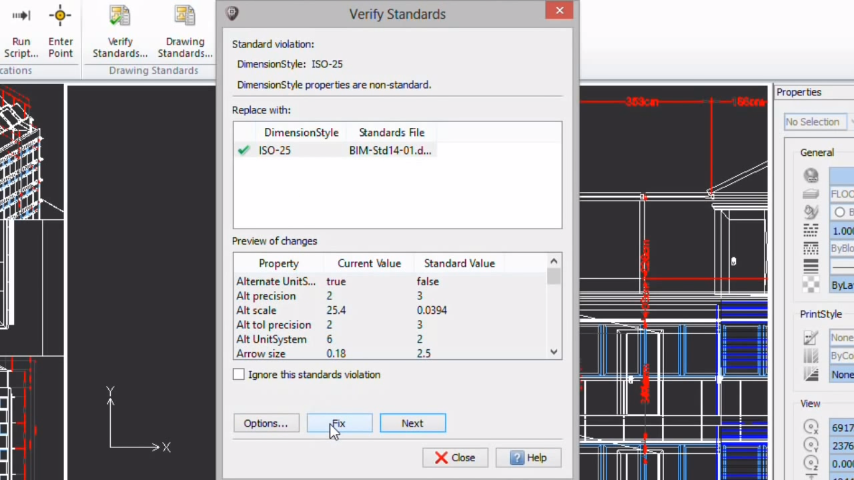
Имеется возможность заменить несоответствующие стандартам элементы.



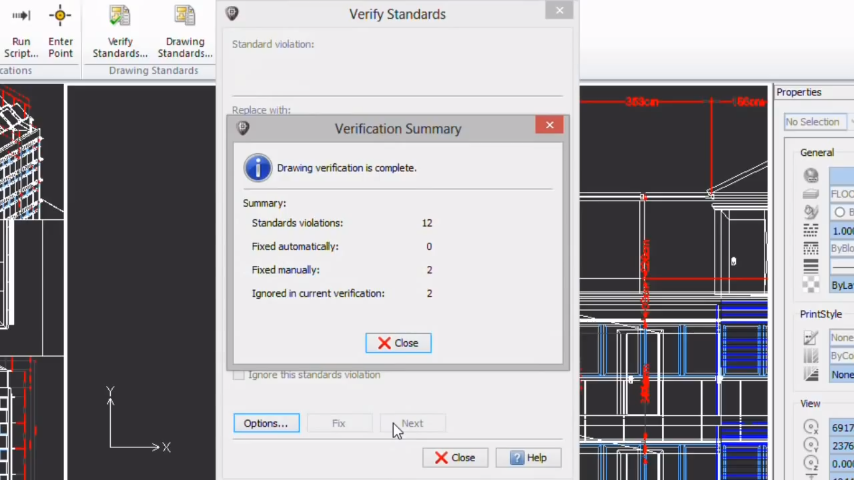
Выбор параметров



Выбор стандарта



Исправление чертежа



Результат работы функции

Данная функция интегрирована в среду проектирования и не требует установки дополнительных компонент.

Программа CorelCAD распространяется на платной основе. Также имеется 30-ти дневная ознакомительная версия.

## Базовые форматы хранения САПР

### DWF

DWF - формат файла, развитый [Autodesk](http://cadobzor.ru/Autodesk) для того, чтобы представить данные проекта в виде, который был бы независим от оригинального прикладного программного обеспечения, аппаратных средств, или операционной системы, с помощью которой создавались данные проекта. Файл DWF может описать данные проекта, содержащие любую комбинацию текста, графики, и изображений в независимом устройстве. Эти файлы могут быть одним листом или многократными листами, очень простыми или чрезвычайно сложными с богатым использованием шрифтов, графики, цвета, и изображений. [6]

Файлы DWF призваны не для замены родных форматов CAD, таких как рисунки [AutoCAD](http://cadobzor.ru/AutoCAD) ([DWG](http://cadobzor.ru/DWG)). Единственная цель DWF состоит в том, чтобы позволить проектировщикам, инженерам и их коллегам сообщать информацию проекта и содержание проекта к любому человеку, которому необходимо рассмотреть или напечатать информацию проекта – без этих участников команды, знающих [AutoCAD](http://cadobzor.ru/AutoCAD) или другое программное обеспечение для проектирования.

Технология DWF файлов базируется на трех компонентах:

* С++ библиотеки для разработки;
* программа для просмотра DWF файлов, которая предоставляет пользователям без знания [AutoCAD](http://cadobzor.ru/AutoCAD) просматривать файлы этого формата;
* программа для создания DWF файлов, позволяющая пользователям создавать файлы этого формата.

DWF - это открытый формат файлов. [Autodesk](http://cadobzor.ru/Autodesk) опубликовала спецификацию данного формата и С++ библиотеки для любого разработчика, который хочет связать свой продукт с форматом DWF. Кроме того, DWF основан на другим стандартах, таких как ZLIB, XML и общие форматы изображения.

Файлы DWF (начиная с версии 6.0) являются ZIP контейнерами для файлов рисунка; несмотря на малое количество первых байтов файла, содержащего заголовок DWF, переименование \*.dwf файл в \*.zip, позволит составляющим файлам внутри просматриваться с помощью программного обеспечения, таким как Winzip.

DWF может соединяться с.NET Библиотеками.

### DWG

DWG-формат — это универсальный формат для хранения данных 2D- и 3D-чертежей, название которого происходит от английского слова *Drawing —* чертеж. DWG (от англ. drawing — чертеж) — бинарный формат файла, используемый для хранения двухмерных (2D) и трёхмерных (3D) проектных данных и метаданных. Является основным форматом для некоторых САПР-программ Формат DWG поддерживается многими САПР-приложениями косвенно: то есть данные из одного формата данных перемещаются в другой через функции импорт-экспорт. Форматы .dws («drawing standards» — стандарты чертежа), .dwt («drawing template» — шаблон чертежа) также являются форматом DWG. Современные программы, работающие в dwg, часто создают на диске временные и резервные копии документов в формате dwg в файлах с расширениями .sv$ («temporary automatic save» — временное автоматическое сохранение) и .bak («backup» — резервная копия).».[7]

### **DXF**

Система AutoCAD может использоваться сама по себе, как полный редактор чертежей. Однако иногда в других программах возникает необходимость анализировать чертежи, созданные системой AutoCAD, или же осуществлять генерацию чертежей с помощью других программ, а их оценку, изменение и вывод на плоттер с помощью системы AutoCAD.

Примером может служить использование системы AutoCAD для описания структур, которые затем передаются на ЭВМ для проведения структурного анализа по методу конечных элементов. Вы можете вычислить напряжения и смещения, а затем передать информацию обратно для изображения деформированной структуры с помощью системы AutoCAD.

Так как база данных чертежей системы AutoCAD хранится в очень сжатом формате, то программам пользователя непосредственно прочитать эту информацию трудно. Кроме того, различные машинные реализации системы AutoCAD могут использовать различные внутренние форматы для базы данных, подобранные для получения максимальной производительности вычислительной машины, на которой запущена система AutoCAD. Для обеспечения возможности обмена файлами чертежей между различными машинными реализациями системы AutoCAD, а также между системой AutoCAD и другими программами был определен формат файла «обмена чертежами».

Данный формат воспринимается всеми машинными реализациями системы AutoCAD, и существует возможность его преобразования как в их внутренний файл чертежа, так и наоборот. [8]

# Глава 2.

# Алгоритмическое обеспечение

## Пересечение двух отрезков

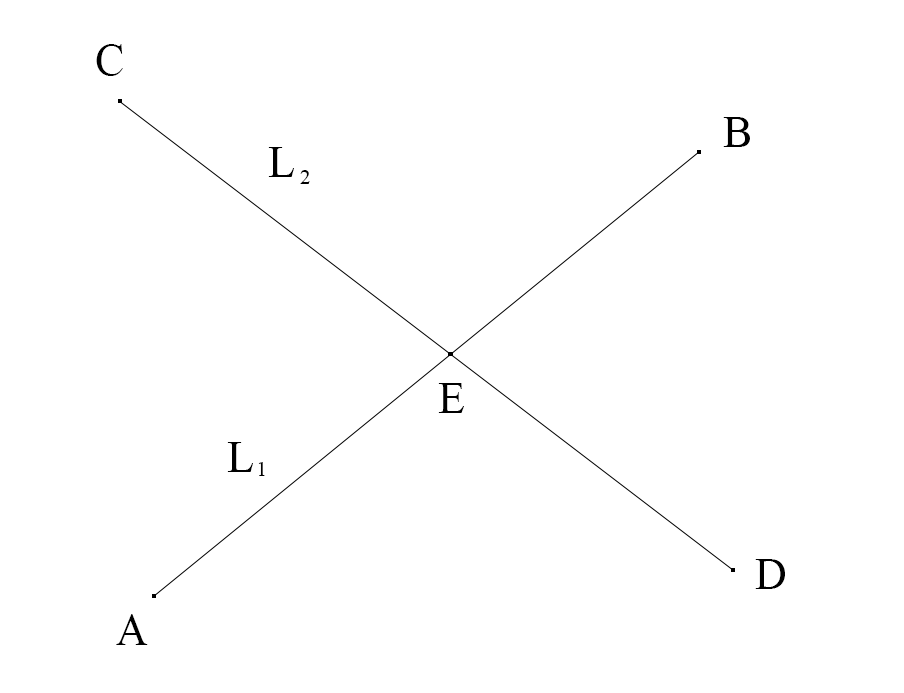


Рис 2

Уравнения отрезков имеют вид:

Так как и являются одной и той же точкой, то уравнения (1) и (2) равны, из чего можно получить следующую систему:

Решая её относительно и получим:

Подставляя полученные значения в начальное уравнение, получим координаты точки пересечения:

Если и , то точка принадлежит и первому и второму отрезку, значит это действительное пересечение отрезков, иначе пересечение будет мнимым.

## Пересечение отрезка и окружности

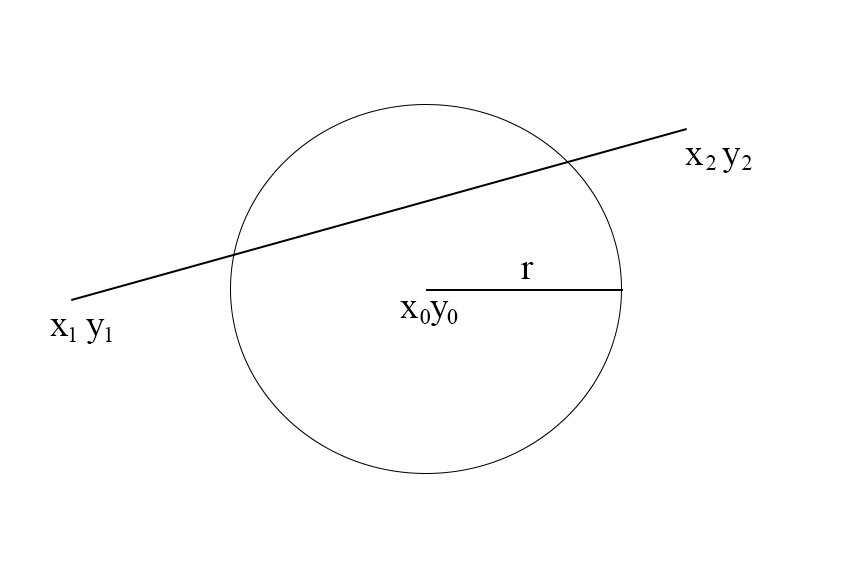


Рис. 3

Для нахождения точек пересечения отрезка и окружности нужно решить систему:

Преобразуем уравнение (1) к уравнению вида

Где

Подставив полученное уравнение в (2) получим координаты и точки пересечения.

Если и , то пересечение будет действительным, иначе оно мнимое, т. к. точка лежит вне отрезка.

## Пересечение отрезка и дуги

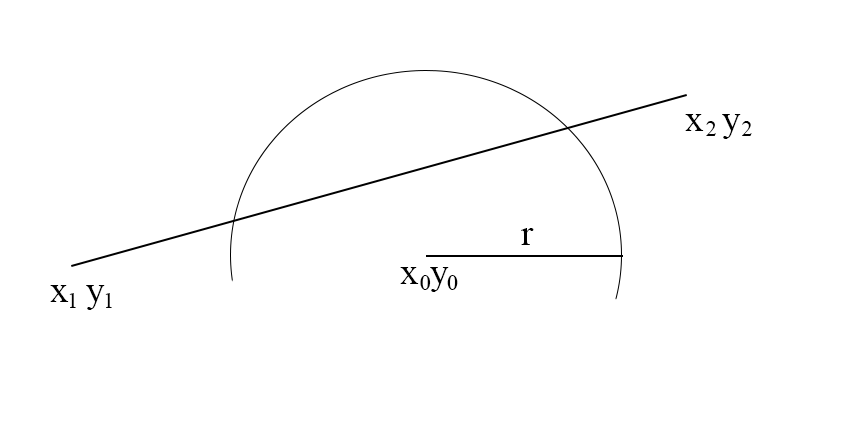


Рис. 4

Поиск точек пересечения отрезка и эллипса осуществляется точно так же как и при поиске пересечений отрезка и окружности.

Далее находим угол между прямой, проходящей через точку и параллельной оси абсцисс, и прямой, проходящей через точки и . Если , то эта точка является точкой пересечения отрезка и эллипса.

Если и , то пересечение будет действительным, иначе оно мнимое, т. к. точка лежит вне отрезка.

## Пересечение отрезка и эллипса

Пусть изначально имеем следующее расположение отрезка и эллипса:

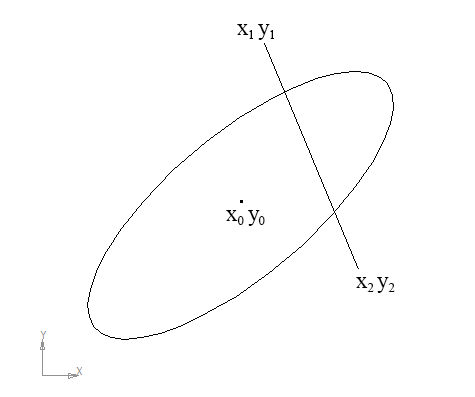


Рис. 5

Для упрощения вычислений выполним сдвиг системы координат так, чтобы центр эллипса оказался в точке :

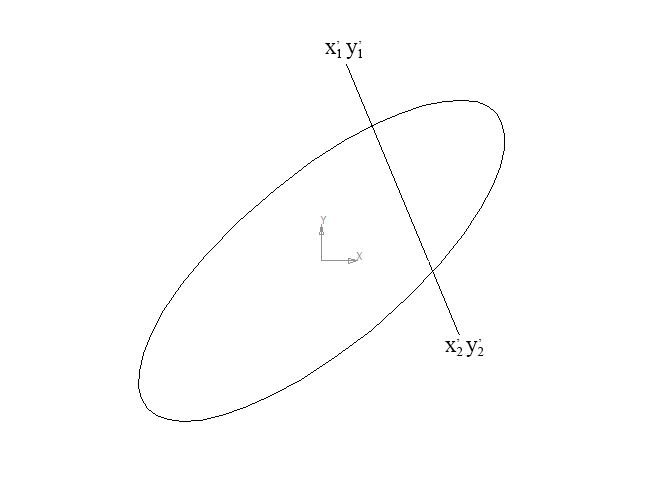


Рис. 6

И выполним поворот системы координат так, чтобы большая ось эллипса лежала на оси абсцисс и меньшая ось лежала на оси ординат:

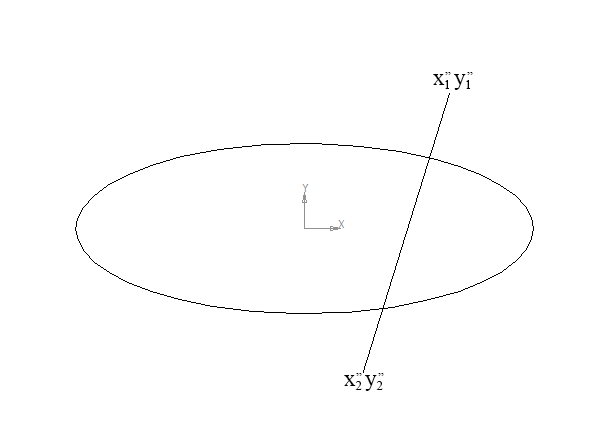


Рис. 7

Таким образом, у точек A и B координаты будут равны и соответственно. И получим систему:

Преобразуем уравнение (1) к виду:

Где

И уравнение (2) к виду:

Подстановкой (3) в (4) получим:

Найдя корни уравнения (6) и подставив их в (3), получим координаты точек пересечения отрезка и эллипса и (Рис. 8).



Рис. 8

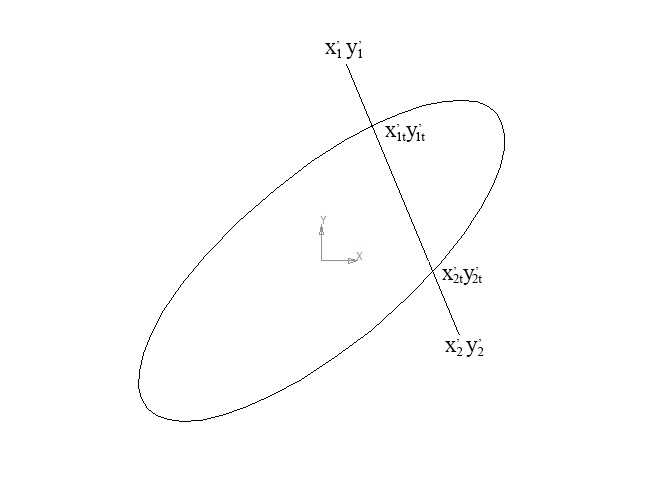


Рис. 9

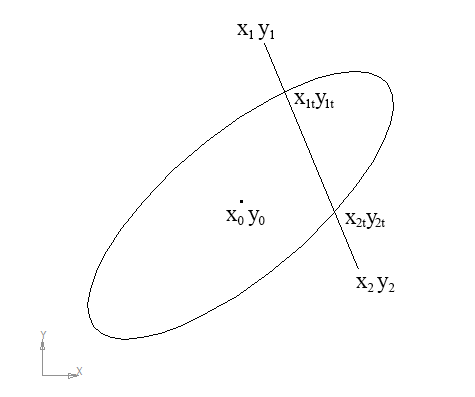


Рис. 10

Далее обратным преобразованием координат (поворот на угол (Рис. 9) и переноса центра координат на и (Рис. 10)) получаем координаты точек пересечения в исходной системе координат и соответственно.

## Пересечение двух окружностей

Рассматриваются 3 случая взаимного расположения двух окружностей:

* Окружности не пересекаются
* Окружности пересекаются (1 или 2 точки пересечения)
* Одна окружность находится внутри другой окружности

### Окружности не имеют общих точек

Две окружности не будут иметь общих точек, если расстояние между их центрами больше суммы радиусов.

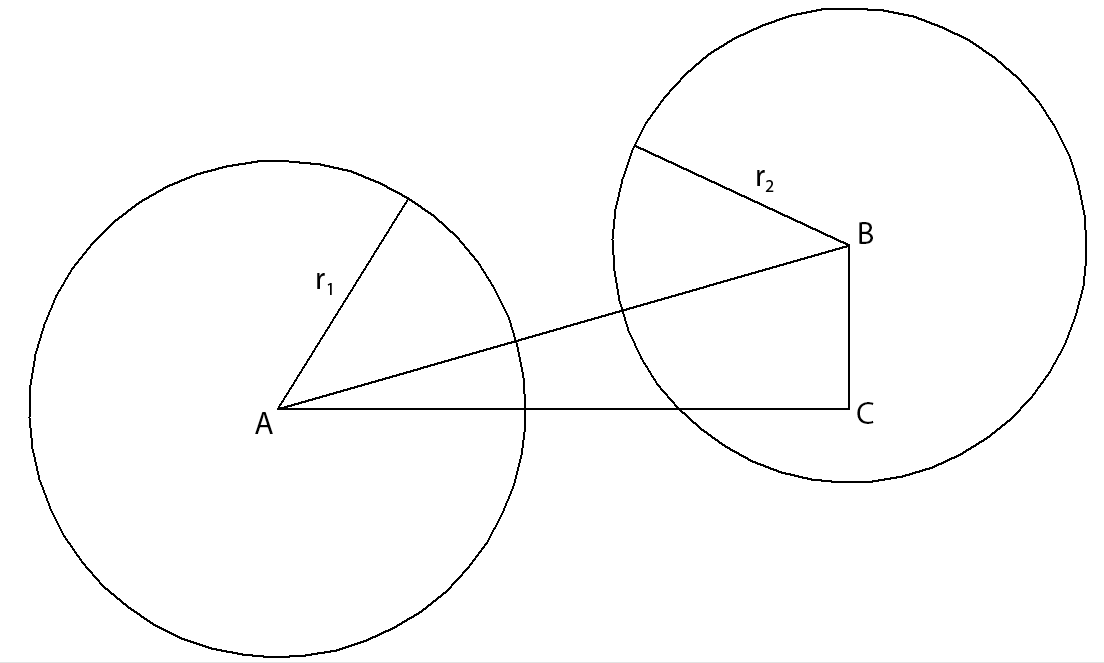


Рис. 11

Расстояние между центрами определяется по прямоугольному треугольнику ABC, по формуле

### Окружности имеют одну общую точку

Две окружности будут иметь общую точку, если расстояние между их центрами равно сумме радиусов.

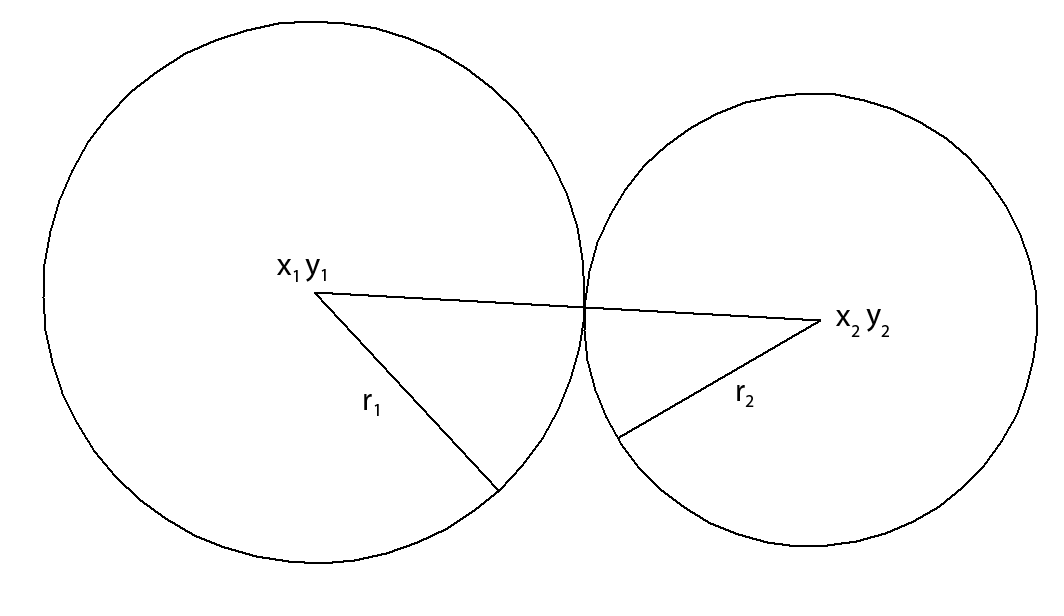


Рис. 12

Расстояние между центрами будет определяться подобным образом.

Координата x будет рассчитываться по формуле

Координата y будет рассчитываться по формуле

Таким образом, получили координаты точки касания окружностей.

### Окружности имеют две общих точки

Две окружности будут иметь две общих точки, если расстояние между их центрами меньше суммы радиусов.

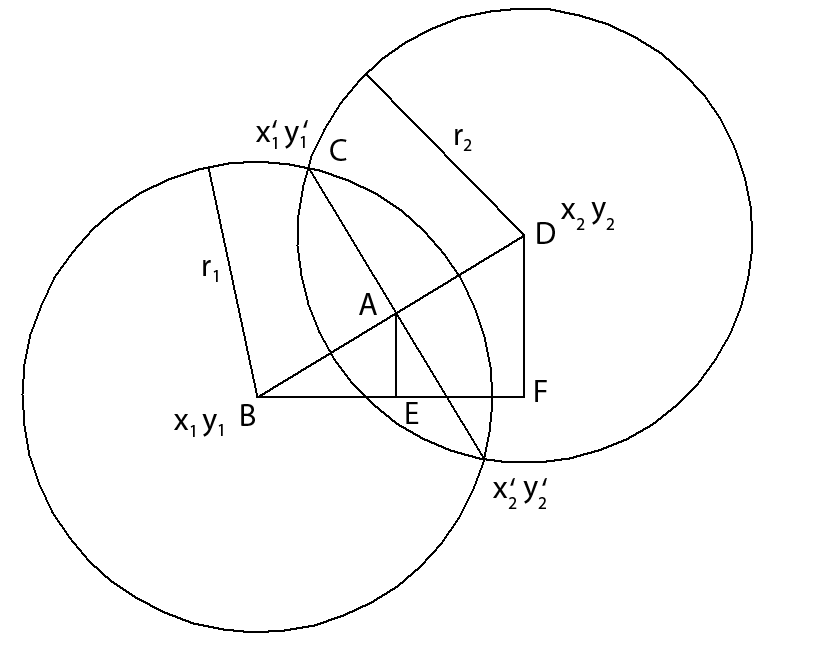


Рис. 13

AC можно найти, выразив высоту треугольника BCD через формулы для нахождения площади треугольника:

где

Т.е.

Отрезок AB можно найти из прямоугольного треугольника ABC, т.е.

Из подобия треугольников BDF и ABE находим AE

Соответственно

Таким образом, координаты точки A будут равны:

Координаты точек пересечения находятся следующим образом:

## Пересечение окружности и дуги

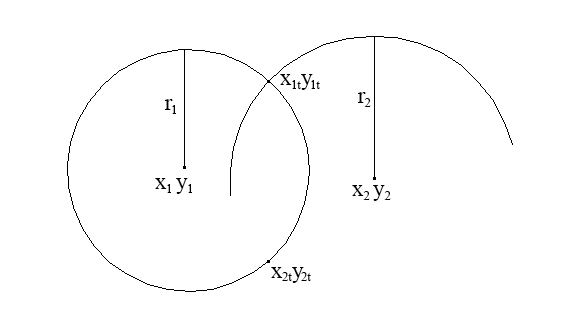


Рис. 14

Поиск точек пересечения окружности и эллипса осуществляется точно так же, как и при поиске пересечений двух окружностей.

Далее находим угол между прямой, проходящей через центр дуги и параллельной оси абсцисс, и прямой, проходящей через точки и . Если , то эта точка является точкой пересечения отрезка и эллипса. Аналогично нужно поступить с точкой .

## Пересечение дуг

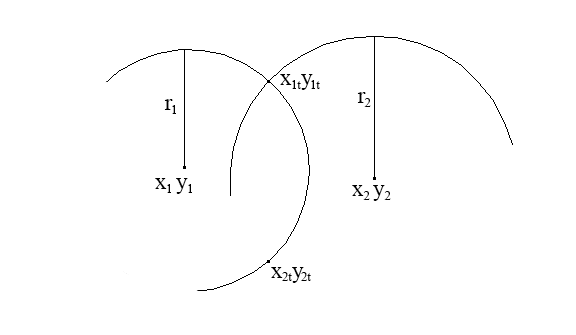


Рис. 15

Поиск точек пересечения двух дуг осуществляется точно так же, как и при поиске пересечений двух окружностей.

Затем необходимо определить, принадлежат ли найденные точки обоим дугам. Для этого нужно найти угол между прямой, параллельной оси абсцисс и проходящей через точку , и прямой, проходящей через точки и ; и угол , между прямой, параллельной оси абсцисс и проходящей через точку , и прямой, проходящей через точки и .

Если и , то найденная точка является точкой пересечения двух дуг.

Аналогичные действия выполняются для точки .

## Совпадение двух отрезков

### Полное совпадение

Два отрезка полностью совпадают друг с другом, если их начальные и конечные координаты попарно равны или если начальные координаты первого отрезка равны конечным координатам второго отрезка и конечные координаты первого отрезка равны начальным координатам второго отрезка.

### Частичное совпадение

Два отрезка частично совпадают друг с другом, если:

* Точки C и D лежат на прямой AB

## Совпадение двух окружностей

Две окружности совпадают друг с другом, если координаты их центров, а также радиусы равны.

## Совпадение двух дуг

Рассматривается полное совпадение дуг. Две дуги полностью совпадают друг с другом, если равны координаты их центров, радиусы, а также начальные и конечные углы.

## Совпадение двух эллипсов

Два эллипса совпадают друг с другом, если у них равны координаты центра, длины обоих осей и углы поворота.

## Совпадение окружности и дуги

Окружность и эллипс совпадают, если координаты их центров и радиусы равны между собой.

# Программное обеспечение

## Условия функционирования ПО

Операционная система: программа предназначена для использования в ОС Windows 7 64bit, с архитектурой процессора x64.

Для использования программы необходимы процессор (Intel Core i3), монитор (рекомендуемое разрешение не ниже 1024х768px), клавиатура, мышь.

Программное обеспечение разрабатывалось в среде Microsoft Visual Studio 2012.

## Разработанная библиотека Cl\_DxfLib

Данная библиотека служит для работы с файлами формата dxf (открытие, чтение, сохранение и закрытие файла).

### Используемые структуры данных

struct POINT {

double x;

double y;

string layer;

};

Переменные, входящие в структуру POINT:

* x, y – координаты точки;
* layer – название слоя, на котором расположена точка.

struct LINE {

POINT p[2];

unsigned short int type;

string layer;

bool current;

};

Переменные, входящие в структуру LINE:

* p – массив, состоящий из двух элементов, которые содержат координаты конечных точек отрезка;
* type – тип линии. Содержит следующие значения: 0 – обычный отрезок, 1 – направляющая прямая.
* layer – название слоя, на котором расположен отрезок;
* current – содержит следующие значения: true, если объект выделен, и false в противном случае.

struct LWPOLYLINE {

vector<POINT> p;

string layer;

bool current;

bool closed;

double width;

};

Переменные, входящие в структуру LWPOLYLINE:

* p – вектор, содержащий координаты точек, из которых состоит полилиния;
* layer – название слоя, на котором расположена полилиния;
* current – содержит следующие значения: true, если объект выделен, и false в противном случае;
* closed – содержит следующие значения: true, если полилиния замкнута, и false в противном случае;
* width – толщина полилинии.

struct CIRCLE {

POINT p;

double r;

string layer;

bool current;

};

Переменные, входящие в структуру CIRCLE:

* p – точка, содержащая координаты центра окружности;
* r – радиус окружности;
* layer – название слоя, на котором расположена окружность;
* current – содержит следующие значения: true, если объект выделен, и false в противном случае.

struct ELLIPSE {

POINT p;

double ratio;

double width;

double height;

double angle;

string layer;

};

Переменные, входящие в структуру ELLIPSE:

* p – точка, содержащая координаты центра эллипса;
* ratio – отношение длинны наибольшей оси к наименьшей;
* width – длина наибольшей полуоси;
* height – длина наименьшей полуоси;
* angle – угол поворота большей оси относительно OX;
* layer – название слоя, на котором расположен эллипс.

struct ARC {

POINT p;

double r;

double angleStart;

double angleEnd;

string layer;

bool current;

};

Переменные, входящие в структуру ARC:

* p – точка, содержащая координаты центра дуги;
* r – диаметр дуги;
* angleStart – начальный угол дуги;
* angleEnd – конечный угол дуги;
* layer – название слоя, на котором расположена дуга;
* current – содержит следующие значения: true, если объект выделен, и false в противном случае.

struct BLOCK {

int beginAddr;

string number;

};

Переменные, входящие в структуру BLOCK:

* beginAddr – позиция начала блока в файле
* number – шестнадцатеричный номер блока

struct INSERT {

string layer;

string blockNumber;

};

Переменные, входящие в структуру INSERT:

* layer – название слоя, на котором расположен блок для вставки;
* blockNumber – номер блока для вставки.

struct ENTITIES {

vector<POINT> Points;

vector<LINE> Lines;

vector<LWPOLYLINE> Polylines;

vector<CIRCLE> Circles;

vector<ELLIPSE> Ellipses;

vector<ARC> Arcs;

vector<INSERT> Inserts;

};

Переменные, входящие в структуру ENTITIES:

* Points – вектор точек, входящих в чертёж;
* Lines – вектор отрезков, входящих в чертёж;
* Polylines – вектор полилиний, входящих в чертёж;
* Circles – вектор окружностей, входящих в чертёж;
* Ellipses – вектор эллипсов, входящих в чертёж;
* Arcs – вектор дуг, входящих в чертёж;
* Inserts – вектор блоков для вставки, входящих в чертёж.

struct SECTION {

ENTITIES Entities;

};

Переменные, входящие в структуру SECTION: ENTITIES – структура, в которой содержатся объекты, входящие в секцию ENTITIES чертежа.

struct CROSSPOINT {

double x;

double y;

bool type;

bool current;

bool isCrossPoint;

vector<LINE> Lines;

vector<CIRCLE> Circles;

vector<ELLIPSE> Ellipses;

vector<ARC> Arcs;

};

Переменные, входящие в структуру CROSSPOINT:

* x, y – координаты точки пересечения;
* type – содержит следующие значения: true, если пересечение действительное, и false в противном случае;
* current – содержит следующие значения: true, если объект выделен, и false в противном случае;
* isCrossPoint – содержит следующие значения: true, если объект является точкой печесечения двух других объектов, и false, если объект является наложением друг на друга двух других объектов;
* Lines – вектор, содержащий два пересекающихся или накладывающихся друг на друга отрезка;
* Circles – вектор, содержащий две пересекающиеся или накладывающиеся друг на друга окружности;
* Ellipses – вектор, содержащий два пересекающихся или накладывающихся друг на друга эллипса;
* Arcs – вектор, содержащий две пересекающиеся или накладывающиеся друг на друга дуги.

fstream file – поток файла для чтения.

string fileName – имя файла для чтения.

ofstream saveFile – поток файла для записи.

string saveFileName – имя файла для записи.

int LastNumberOfObject – последний порядковый номер объектов.

int EndEntitiesPointer – позиция конца блока ENTITIES в исходном файле.

SECTION \_Section – структура для записи объектов.

vector<BLOCK> Blocks – вектор объектов типа BLOCK, входящих в чертёж.

int NumBlock – число объектов типа BLOCK, входящих в чертёж.

### Публичные функции

Функции, перечисленные в этом разделе, объявлены публичными, так как составляют интерфейс для работы пользователя с рассматриваемой библиотекой.

#### bool Open(string dxfFileName)

Функция открытия файла по указанному имени и пути к файлу.

##### Описание входных и выходных параметров

Входной параметр string dxfFileName – имя файла без указания пути к нему, если он находится в той же категории, что и программа. Если же программа и файл находятся в разных категориях, то необходимо указать полный путь к файлу.

Выходной параметр имеет тип boolean, содержащий значение true в случае успешного открытия файла или false в случае возникновения какой-либо ошибки.

#### int Read()

Функция чтения информации о чертеже из ранее открытого файла.

При построчном чтении файла считывается общая информация о чертеже и его объекты. При обнаружении каждого из возможных объектов, вызывается соответствующая ему функция, которая получает информацию о текущем объекте и записывает её в переменные.

##### Описание входных и выходных параметров

Входные параметры отсутствуют.

Выходной параметр имеет тип integer. Может содежать следуюшие значения: 0 – при успешном выполнении функции, 1 – при не обнаружении секции ENTITIES в файле.

#### bool SaveErrorPoints(vector<CROSSPOINT> errPoints)

Функция сохранения файла с отметками о возможных ошибках.

##### Описание входных и выходных параметров

Входной параметр vector<CROSSPOINT> errPoints – вектор структур CROSSPOINT, которые содержат информацию о точках возможных ошибок.

Выходной параметр имеет тип boolean. Содержит значение true в случае успешного сохранения файла или false в случае возникновения какой-либо ошибки.

#### bool SavePolyLine(string \_saveFileName, double \*\*points, int size, bool closed)

Функция сохранения файла с полилинией, передаваемой в параметрах.

##### Описание входных и выходных параметров

Входные параметры:

* string \_saveFileName – содержит имя сохраняемого файла;
* double \*\*points – массив точек полилинии, где points[i][0] - координата X, points[i][1] - координата Y;
* int size – толщина полилинии;
* bool closed – содержит true, если полилиния замкнута, и false в противном случае.

Выходной параметр имеет тип boolean. Содержит значение true в случае успешного сохранения файла или false в случае возникновения какой-либо ошибки.

#### void Close()

Функция закрытия файла и обнуления использованных переменных.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### SECTION GetSection()

Функция предоставляет доступ к приватной переменной \_Section.

##### Описание входных и выходных параметров

Входные параметры отсутствуют.

Выходной параметр имеет тип SECTION. В данном параметре содержится приватная переменная \_Section, в которой содержатся полученные из файла объекты.

### Приватные функции

Функции, объявленные в этом разделе, являются приватными, так как не входят в интерфейс работы с рассматриваемой библиотекой и не предполагают непосредственной работы пользователя с ними.

#### void ReadLine()

Функция void ReadLine() вызывается функцией int Read() при обнаружении объекта LINE в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Lines с параметром type, равным 0.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadPolyline()

Функция void ReadPolyline() вызывается функцией int Read() при обнаружении объекта LWPOLYLINE в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Polylines.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadXLine()

Функция void ReadXLine() вызывается функцией int Read() при обнаружении объекта XLINE в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Lines с параметром type, равным 1.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadCircle()

Функция void ReadCircle() вызывается функцией int Read() при обнаружении объекта CIRCLE в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Circles.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadBlock()

Функция void ReadBlock() вызывается функцией int Read() при обнаружении объекта BLOCK в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор Blocks. Во время выполнения функции происходит анализ блока посредством чтения строк файла и при обнаружении объекта LINE вызывается функция void ReadLine().

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadInsert()

Функция void ReadInsert() вызывается функцией int Read() при обнаружении объекта INSERT в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Inserts.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadEllipse()

Функция void ReadEllipse() вызывается функцией int Read() при обнаружении объекта ELLIPSE в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Ellipses.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void ReadArc()

Функция void ReadArc() вызывается функцией int Read() при обнаружении объекта ARC в обрабатываемом файле. Данная функция создаёт временную переменную, считывает из файла информацию об обнаруженном объекте во временную переменную и добавляет её в вектор \_Section.Entities.Arcs.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### int TextToHex(string str)

Функция преобразования шестнадцатеричного числа, представленного в строковом выражении, в шестнадцатеричное число типа integer.

##### Описание входных и выходных параметров

Входной параметр string str представляет собой строку, в которой содержится шестнадцатеричное число, представленное в строковом выражении.

Выходной параметр типа integer представляет собой шестнадцатеричное число в целочисленном формате.

#### string LineToString(LINE l)

Функция преобразования объекта LINE в строковое представление для записи в файл dxf.

##### Описание входных и выходных параметров

Входной параметр LINE l представляет собой объект типа LINE.

Выходной параметр типа string имеет строковое представление для записи в файл dxf.

#### string PolylineToString(LWPOLYLINE l)

Функция преобразования объекта LWPOLYLINE в строковое представление для записи в файл dxf.

##### Описание входных и выходных параметров

Входной параметр LWPOLYLINE l представляет собой объект типа LWPOLYLINE.

Выходной параметр типа string имеет строковое представление для записи в файл dxf.

#### string CircleToString(CIRCLE c)

Функция преобразования объекта CIRCLE в строковое представление для записи в файл dxf.

##### Описание входных и выходных параметров

Входной параметр CIRCLE c представляет собой объект типа CIRCLE.

Выходной параметр типа string имеет строковое представление для записи в файл dxf.

#### string ErrPointToString(CROSSPOINT point)

Функция преобразования объекта CROSSPOINT в строковое представление для записи в файл dxf.

##### Описание входных и выходных параметров

Входной параметр CROSSPOINT point c представляет собой объект типа CROSSPOINT.

Выходной параметр типа string имеет строковое представление для записи в файл dxf.

#### string LayerToString(string name)

Функция составления имени слоя в его строковое представление для записи в файл dxf.

##### Описание входных и выходных параметров

Входной параметр string name c представляет собой имя слоя.

Выходной параметр типа string имеет строковое представление для записи в файл dxf.

## Разработанная библиотека Cl\_ObjectsLib

Данная библиотека служит для обнаружения пересечений и совпадений объектов, полученных из файла dxf, а так же для обнаружения взаимодействия с объектом, посредством клика по нему мышью.

### Используемые структуры данных

struct BITSFIELD {

unsigned char crossKind;

bool line\_line;

bool line\_circle;

bool line\_ellipse;

bool line\_arc;

bool circle\_circle;

bool circle\_ellipse;

bool circle\_arc;

bool ellipse\_ellipse;

bool ellipse\_arc;

bool arc\_arc;

bool overlap\_lines;

bool overlap\_circles;

bool overlap\_ellipses;

bool overlap\_arcs;

bool overlap\_circle\_arc;

};

Переменные, входящие в структуру BITSFIELD:

* crossKind – виды пересечений, которые должны быть обработаны. Может содержать следующие значения: 0 - все пересечения, 1 - только действительные, 2 - только мнимые.
* line\_line – содержит true, если необходимо проверить пересечение двух отрезков;
* line\_circle – содержит true, если необходимо проверить пересечение отрезка и окружности;
* line\_ellipse – содержит true, если необходимо проверить пересечение отрезка и эллипса;
* line\_arc – содержит true, если необходимо проверить пересечение отрезка и дуги;
* circle\_circle – содержит true, если необходимо проверить пересечение двух окружностей;
* circle\_ellipse – содержит true, если необходимо проверить пересечение окружности и эллипса;
* circle\_arc – содержит true, если необходимо проверить пересечение окружности и дуги;
* ellipse\_ellipse – содержит true, если необходимо проверить пересечение двух эллипсов;
* ellipse\_arc – содержит true, если необходимо проверить пересечение эллипса и дуги;
* arc\_arc – содержит true, если необходимо проверить пересечение двух дуг;
* overlap\_lines – содержит true, если необходимо проверить совпадение двух отрезков;
* overlap\_circles – содержит true, если необходимо проверить совпадение двух окружностей;
* overlap\_ellipses – содержит true, если необходимо проверить совпадение двух эллипсов;
* overlap\_arcs – содержит true, если необходимо проверить совпадение двух дуг;
* overlap\_circle\_arc – содержит true, если необходимо проверить совпадение окружности и дуги.

SECTION Section – структура, в которой содержатся ранее считанные из dxf-файла объекты чертежа.

vector<CROSSPOINT> ErrPoints – вектор, содержащий точки пересечений и совпадений объектов.

### Публичные функции

Функции, перечисленные в этом разделе, объявлены публичными, так как составляют интерфейс для работы пользователя с рассматриваемой библиотекой.

#### void CheckCross(BITSFIELD bitsfield)

В рассматриваемой функции происходит попарная передача объектов, содержащихся в переменной Section, в функции, где непосредственно выполняется проверка пересечений и совпадений, в соответствии с флагами, выставленными в передаваемом параметре bitsfield.

##### Описание входных и выходных параметров

Входной параметр BITSFIELD bitsfield. В нём передаются флаги для определения того, какие объекты необходимо проверить на пересечение и совпадение.

Выходные параметры отсутствуют.

#### void InitBitsField(BITSFIELD \*field)

Функция инициализации структуры BITSFIELD.

##### Описание входных и выходных параметров

Входной параметр BITSFIELD \*field – указатель на структуру типа BITSFIELD.

Выходные параметры отсутствуют.

#### void ClickOnObject(double x, double y, double scale, DataGridView^ table)

Функция распознавания объекта, по которому был совершён клик мыши, вывода информации об этом объекте и изменения параметра current этого объекта.

##### Описание входных и выходных параметров

Входные параметры:

* x, y – координаты места, по которому был совершён клик мыши;
* scale – коэффициент масштабирования при отображении объектов на экране;
* table – таблица, в которую будет выведена информация о выбранном файле.

Выходные параметры отсутствуют.

### Приватные функции

Функции, объявленные в этом разделе, являются приватными, так как не входят в интерфейс работы с рассматриваемой библиотекой и не предполагают непосредственной работы пользователя с ними.

#### void CrossLines(LINE line1, LINE line2, unsigned char crossKind)

Функция определения пересечения двух отрезков. Алгоритм её работы описан в разделе 3.1. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool line\_line равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* line1 – первый объект типа LINE;
* line2 – второй объект типа LINE;
* crossKind – виды пересечений, которые должны быть обработаны.

Выходные параметры отсутствуют.

#### void CrossLineCircle(LINE line, CIRCLE circle, unsigned char crossKind)

Функция определения пересечения отрезка и окружности. Алгоритм её работы описан в разделе 3.2. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool line\_circle равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* line – объект типа LINE;
* circle – объект типа CIRCLE;
* crossKind – виды пересечений, которые должны быть обработаны.

Выходные параметры отсутствуют.

#### void CrossLineArc(LINE line, ARC arc, unsigned char crossKind)

Функция определения пересечения отрезка и дуги. Алгоритм её работы описан в разделе 3.3. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool line\_arc равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* line – объект типа LINE;
* arc – объект типа ARC;
* crossKind – виды пересечений, которые должны быть обработаны.

Выходные параметры отсутствуют.

#### void CrossLineEllipse(LINE line, ELLIPSE ellipse, unsigned char crossKind)

Функция определения пересечения отрезка и эллипса. Алгоритм её работы описан в разделе 3.4. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool line\_ellipse равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* line – объект типа LINE;
* ellipse – объект типа ELLIPSE;
* crossKind – виды пересечений, которые должны быть обработаны.

Выходные параметры отсутствуют.

#### void CrossCircles(CIRCLE circle1, CIRCLE circle2)

Функция определения пересечения двух окружностей. Алгоритм её работы описан в разделе 3.5. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool circle\_circle равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* circle1– первый объект типа CIRCLE;
* circle2– второй объект типа CIRCLE.

Выходные параметры отсутствуют.

#### void CrossCircleArc(CIRCLE c, ARC a)

Функция определения пересечения окружности и дуги. Алгоритм её работы описан в разделе 3.6. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool circle\_arc равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* c – первый объект типа CIRCLE;
* a – второй объект типа ARC.

Выходные параметры отсутствуют.

#### void CrossArcs(ARC a1, ARC a2)

Функция определения пересечения двух дуг. Алгоритм её работы описан в разделе 3.7. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool arc\_arc равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* a1 – первый объект типа ARC;
* a2 – второй объект типа ARC.

Выходные параметры отсутствуют.

#### void OverlapLines(LINE line1, LINE line2)

Функция определения совпадения (полного и частичного) двух отрезков. Алгоритм её работы описан в разделе 3.8. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool overlap\_lines равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* line1 – первый объект типа LINE;
* line2 – второй объект типа LINE.

Выходные параметры отсутствуют.

#### void OverlapCircles(CIRCLE circle1, CIRCLE circle2)

Функция определения совпадения двух окружностей. Алгоритм её работы описан в разделе 3.9. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool overlap\_circles равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* circle1 – первый объект типа CIRCLE;
* circle2 – второй объект типа CIRCLE.

Выходные параметры отсутствуют.

#### void OverlapArcs(ARC a1, ARC a2)

Функция определения совпадения (полного и частичного) двух дуг. Алгоритм её работы описан в разделе 3.10. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool overlap\_arcs равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* a1 – первый объект типа ARC;
* a2 – второй объект типа ARC.

Выходные параметры отсутствуют.

#### void OverlapEllipses(ELLIPSE e1, ELLIPSE e2)

Функция определения совпадения двух эллипсов. Алгоритм её работы описан в разделе 3.11. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool overlap\_ellipses равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* e1 – первый объект типа ELLIPSE;
* e2 – второй объект типа ELLIPSE.

Выходные параметры отсутствуют.

#### void OverlapCircleArc(CIRCLE c, ARC a)

Функция определения совпадения окружности и дуги. Алгоритм её работы описан в разделе 3.12. Вызывается из функции void CheckCross(BITSFIELD bitsfield) при bool overlap\_circle\_arc равным true в BITSFIELD bitsfield.

##### Описание входных и выходных параметров

Входные параметры:

* c – первый объект типа CIRCLE;
* a – второй объект типа ARC.

Выходные параметры отсутствуют.

#### double Det(double x1, double x2, double x3, double x4)

Функция вычисления детерминанта матрицы:

##### Описание входных и выходных параметров

Входные параметры: x1, x2, x3, x4 – параметры матрицы.

Выходной параметр имеет тип double и является значением детерминанта матрицы.

#### double GetAngle(double x0, double y0, double x1, double y1)

Функция определения угла между прямой, проведённой через точку и параллельной оси абсцисс, и прямой, проведённой через точки и .

##### Описание входных и выходных параметров

Входные параметры:

* x0, y0 – координаты точки ;
* x1, y1 – координаты точки .

Выходной параметр имеет тип double и является значением искомого угла.

#### bool IsPointOnLine(LINE line, double x, double y)

Функция определения принадлежности точки линии line.

##### Описание входных и выходных параметров

Входные параметры:

* line – линия;
* x, y – координаты точки .

Выходной параметр имеет тип bool и принимает значение true, если точка принадлежит отрезку, иначе false.

#### double GetDistance(double x0, double y0, double x1, double y1)

Функция определения расстояния между точкой и точкой .

##### Описание входных и выходных параметров

Входные параметры:

* x0, y0 – координаты точки ;
* x1, y1 – координаты точки .

Выходной параметр имеет тип double и является расстоянием между рассматриваемыми точками.

## Разработанная библиотека Cl\_GraphicsLib

Данная библиотека служит для работы с графической частью программы: отображение, масштабирование и перемещение объектов чертежа посредством OpenGL.

### Используемые структуры данных

SimpleOpenGlControl^ OpenGlControl – элемент Windows Forms, позволяющий работать с графикой посредством OpenGL.

bool IsInit – переменная, содержащая true, если была выполнена инициализация OpenGlControl, и false в противном случае.

### Публичные функции

Функции, перечисленные в этом разделе, объявлены публичными, так как составляют интерфейс для работы пользователя с рассматриваемой библиотекой.

#### void GetOpenGlControl(SimpleOpenGlControl^ simpleOpenGlControl)

Функция, получающая элемент Windows Forms, для дальнейшей работы с ним.

##### Описание входных и выходных параметров

Входной параметр SimpleOpenGlControl^ simpleOpenGlControl. В данном параметре передаётся элемент Windows Forms, для дальнейшей работы библиотеки с ним.

Выходные параметры отсутствуют.

#### int InitGL()

Функция инициализации OpenGL, задание цветов отчистки, используемой цветовой модели и размера буфера; задание размеров области вывода.

##### Описание входных и выходных параметров

Входные параметры отсутствуют.

Выходной параметр имеет тип integer и принимает значение 0, при успешном выполнении функции, или -1, если в библиотеку не был передан элемент формы.

#### void Resize(double width, double height)

Функция изменения размеров области отрисовки.

##### Описание входных и выходных параметров

Входные параметры:

* width – новая ширина области отрисовки;
* height – новая высота области отрисовки.

Выходные параметры отсутствуют.

#### void Render()

Функция задания текущего цвета для отрисовки элементов, отрисовка осей абсцисс и ординат.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void SwapBuffers()

Очистка буферов OpenGL. Посредством чего отрисовываются элементы, находившиеся в этих буферах.

##### Описание входных и выходных параметров

Входные и выходные параметры отсутствуют.

#### void Scale(signed short int s)

Функция изменения масштаба отрисовываемых элементов.

##### Описание входных и выходных параметров

Входной параметр s принимает значения 1, для увеличения масштаба в 2 раза, или -1 для уменьшения масштаба в 2 раза.

Выходные параметры отсутствуют.

#### void Translate(double x, double y)

Функция смещения отображаемых элементов в горизонтальном и вертикальном направления.

##### Описание входных и выходных параметров

Входные параметры:

* x – сдвиг по оси абсцисс;
* y – сдвиг по оси ординат.

Выходные параметры отсутствуют.

#### void DrawPoint(double x, double y, float size)

Функция отображения точки заданного размера в заданных координатах.

##### Описание входных и выходных параметров

Входные параметры:

* x, y – координаты точки;
* size – размер точки в пикселях.

Выходные параметры отсутствуют.

#### void DrawLine(double x1, double y1, double x2, double y2)

Функция отображения отрезка по заданным координатам.

##### Описание входных и выходных параметров

Входные параметры:

* x1, y1 – координаты первой точки;
* x2, y2 – координаты второй точки.

Выходные параметры отсутствуют.

#### void DrawCircle(double x, double y, double r)

Функция отображения окружности с заданными координатами центра и радиусом.

##### Описание входных и выходных параметров

Входные параметры:

* x, y – координаты центра окружности;
* r – радиус окружности.

Выходные параметры отсутствуют.

#### void DrawErrorPoint(double x, double y, double r)

Функция отображения точки пересечения (совпадения) с заданными координатами центра и радиусом. Принцип работы заключается в изменении цвета отображения и вызове функции DrawCircle.

##### Описание входных и выходных параметров

Входные параметры:

* x, y – координаты центра окружности;
* r – радиус окружности.

Выходные параметры отсутствуют.

#### void DrawArc(double x, double y, double r, double begin, double end)

Функция отображения дуги с заданными координатами центра, радиуса, начальным и конечным углами.

##### Описание входных и выходных параметров

Входные параметры:

* x, y – координаты центра дуги;
* r – радиус дуги;
* begin – начальный угол дуги;
* end – конечный угол дуги.

Выходные параметры отсутствуют.

#### void DrawEllipse(double x, double y, double a, double b, double angle)

Функция отображения эллипса с заданными координатами центра, длинами большой и малой полуосей, углом наклона.

##### Описание входных и выходных параметров

Входные параметры:

* x, y – координаты центра эллипса;
* a – длина большей полуоси;
* b – длина меньшей полуоси;
* angle – угол наклона эллипса относительно оси абсцисс.

Выходные параметры отсутствуют.

#### void SetColor(float red, float green, float blue)

Функция установки цвета модели RGB, которым будут отображаться объекты. Параметры могут принимать значения в диапазоне .

##### Описание входных и выходных параметров

Входные параметры:

* red – интенсивность красного цвета;
* green – интенсивность зелёного цвета;
* blue – интенсивность голубого цвета.

Выходные параметры отсутствуют.

#### bool GetIsInit()

Функция, возвращающая значение переменной IsInit.

##### Описание входных и выходных параметров

Входные параметры отсутствуют.

Выходной параметр принимает значение boolean и является значением переменной IsInit.

## Используемое программное обеспечение

### Библиотека The Tao Framework

Библиотека The Tao Framework используется для отрисовки объектов чертежа посредством OpenGL на элементах Windows Forms. Непосредственное использование OpenGL в программе является затруднительным, поскольку при разработке программы использовался язык C++/CLI.

Данная библиотека распространяется по лицензии MIT, что предполагает право на неограниченное использование, копирование, изменение, добавление, публикацию, распространение, сублицензирование и/или продажу копий программного обеспечения.

### Библиотека MySql.Data

Библиотека MySql.Data используется для работы с базой данных MySQL. Данная библиотека входит в пакет Connector.NET, поставляемый с СУБД MySQL.

# Литература

CorelCAD 2014 Руководство обозревателя [1]

Описание формата dxf: <http://www.autodesk.com/techpubs/autocad/acadr14/dxf/>

The Tao Framework: <http://sourceforge.net/projects/taoframework/>

Руководство пользователя CorelCAD [http://corel-cad.ru/html/hlpid\_file\_audit.htm [2](http://corel-cad.ru/html/hlpid_file_audit.htm%20%5b2)]

Функции CorelCAD <http://www.coreldraw.com/ru/product/cad-software/#tab2>