

Федеральное агентство по образованию Российской Федерации
Государственное образовательное учреждение
Высшего профессионального образования
Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий математики механики

СТРУКТУРЫ ХРАНЕНИЯ ГЕОМЕТРИЧЕСКИХ ОБЪЕКТОВ

Отчет по лабораторной работе

Выполнил:

студент ИИТММ гр. 381903-3

Алилуев А.О. _____

Проверил:

ассистент каф. МОСТ, ИИТММ

Лебедев И.Г. _____.

Содержание

1.Введение	3
2.Постановка целей и задач	4
3.Руководство пользователя	5
4.Руководство программиста	7
4.1.Описание структуры программы	7
4.2.Описание структур данных	8
4.3.Описание алгоритмов	12
6.Заключение	13
7.Литература	14

1. Введение

Простейшие геометрические объекты являются неотъемлемой частью любого раздела геометрии, поэтому их хранение и обработка является достаточно актуальной и нетривиальной задачей. Для хранения относительно простых объектов, таких как точка, отрезок, окружность и т.д. существует достаточно простое представление, например, точку можно представить, как координаты по оси X и по оси Y . Отрезок можно представить, как координаты двух точек, а окружность, как точку, являющуюся ее центром, и радиус этой окружности. Но не все фигуры имеют такое простое представление. Иногда требуется сохранить «ломаную линию», у которой изначально не известно количество перегибов. Тогда требуется структура данных, которая динамически сможет добавлять новые элементы в уже существующий объект. В качестве такой структуры рассмотрим плекс.

Для лабораторной работы плекс будет представлять из себя структуру, которая хранит указатель либо на какую-либо точку ломаной линии, либо указатель на другой плекс. Таким образом у нас получается бинарное дерево, для обхода которого будем использовать рекурсивный метод с динамическим преобразованием типов.

2. Постановка целей и задач

Целью лабораторной работы является создание структур хранения простых геометрических объектов, а также структуры «Плекс» для хранения нетипичных геометрических объектов и методов работы с ним, таких как:

- Создание плекса от 2-ух точек или от точки и другого плекса;
- Добавление линии в плекс;
- Вывод плекса на экран.

Для реализации алгоритмов будет использоваться несколько классов:

- TPoint;
- TLine;
- TCircle;
- TTriangle;
- TPlex.

Для проверки правильности работы этих классов будут написаны тесты с использованием фреймворка Google Test, а также тестовый образец программы, которая будет показана пользователю.

3.Руководство пользователя

После запуска программы пользователя встречает консольное окно (рис. 1):

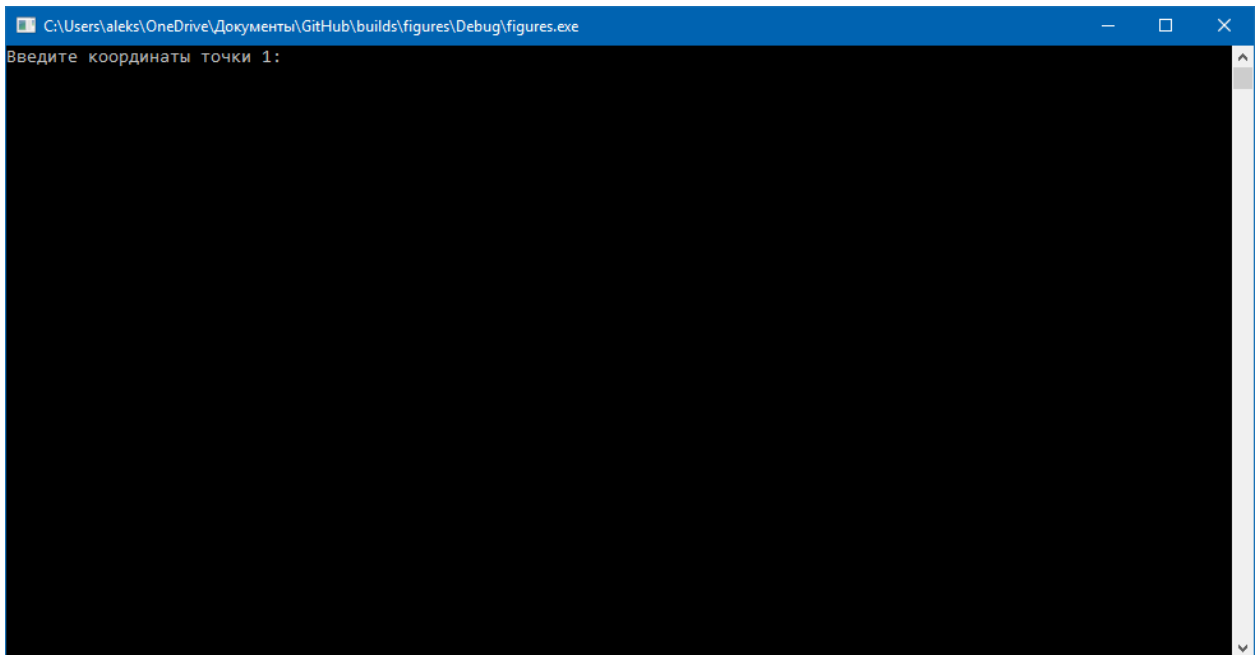


рис. 1 (вывод программы тестирования списка для пользователя)

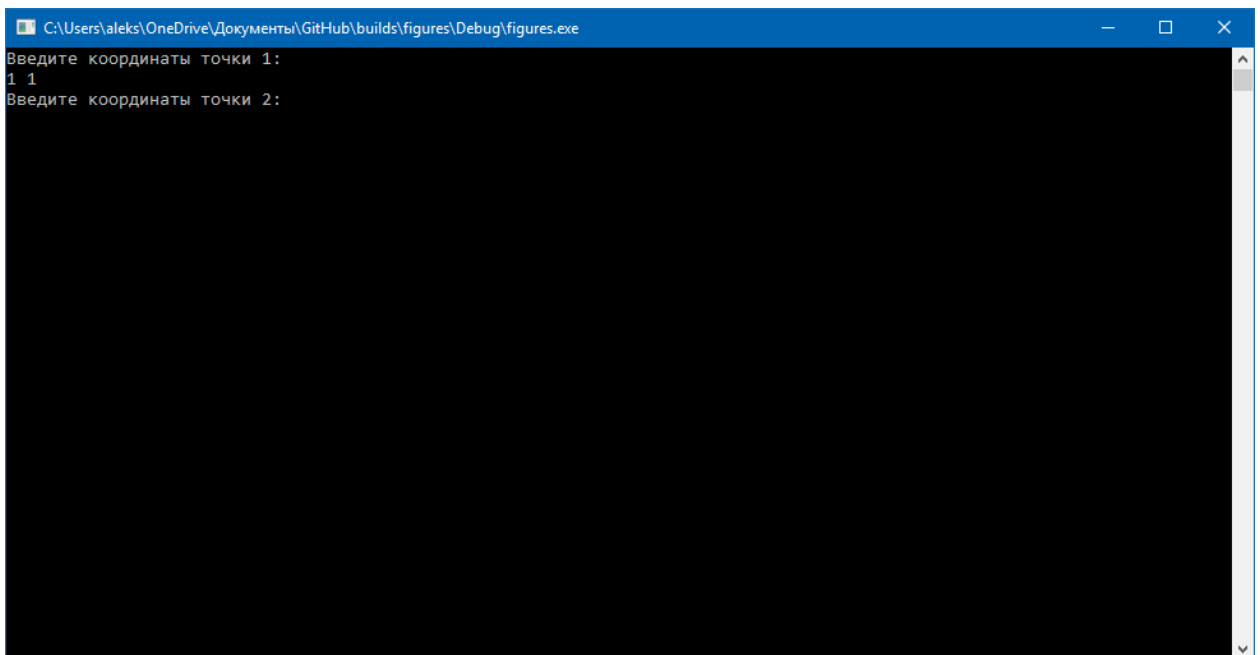
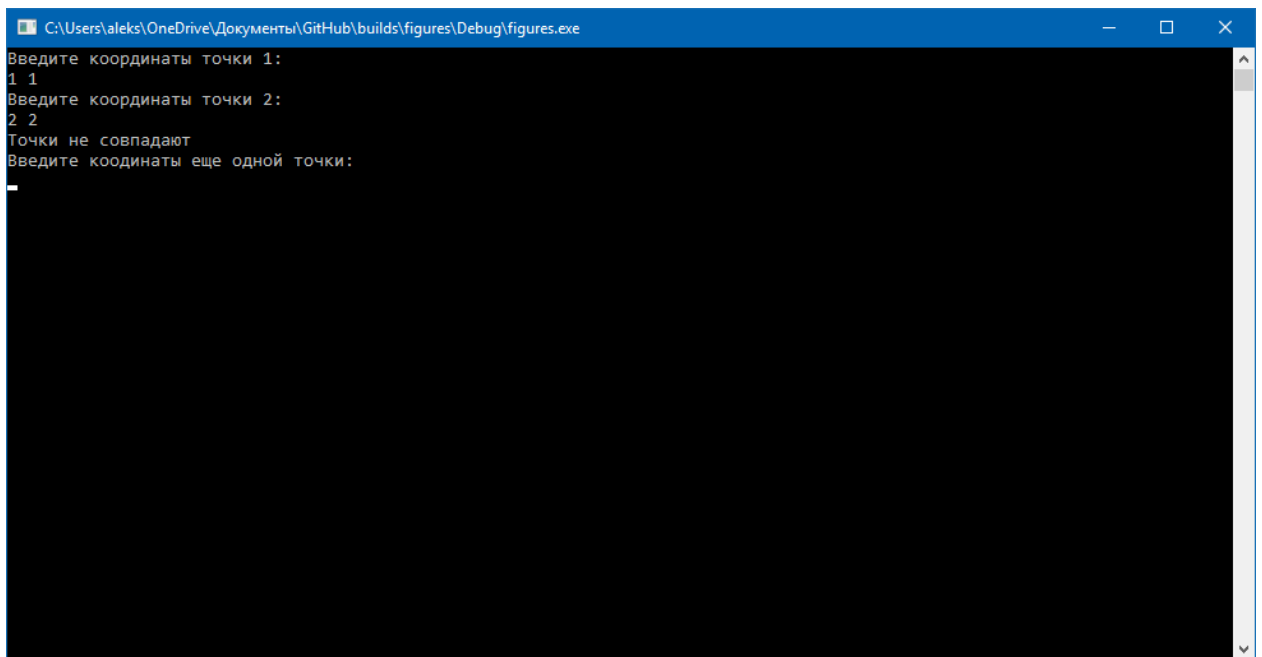


рис. 2 (ввод координат второй точки)

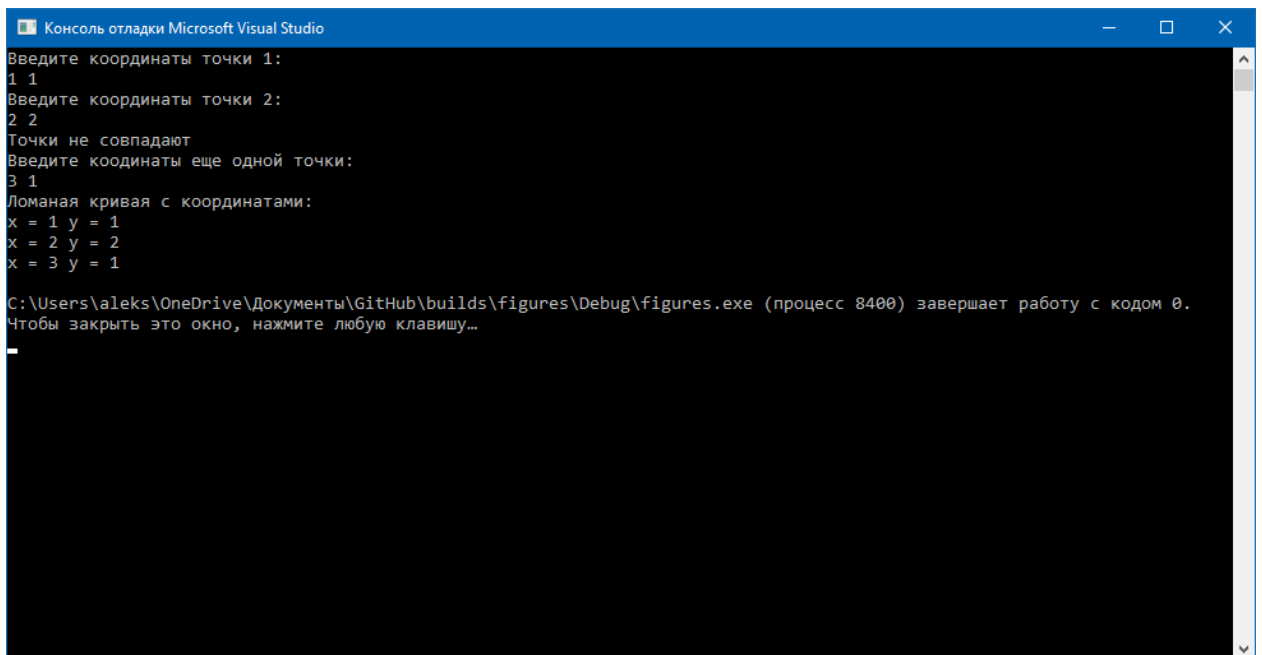
в которой сначала требуется ввести координаты первой точки, а затем координаты второй точки (рис. 2). Затем программа выдаст сообщение о том совпадают ли эти точки или нет, после чего попросит ввести координаты еще одной точки для построения плекса из трех точек (рис. 3).



```
C:\Users\aleks\OneDrive\Документы\GitHub\builds\figures\Debug\figures.exe
Введите координаты точки 1:
1 1
Введите координаты точки 2:
2 2
Точки не совпадают
Введите координаты еще одной точки:

```

рис. 3 (вывод сообщения о совпадении точек и ввод третьей точки)



```
Консоль отладки Microsoft Visual Studio
Введите координаты точки 1:
1 1
Введите координаты точки 2:
2 2
Точки не совпадают
Введите координаты еще одной точки:
3 1
Ломаная кривая с координатами:
x = 1 y = 1
x = 2 y = 2
x = 3 y = 1
C:\Users\aleks\OneDrive\Документы\GitHub\builds\figures\Debug\figures.exe (процесс 8400) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...

```

рис. 4 (вывод координат ломаной прямой)

Окончательный вывод программы – координаты ломаной прямой (рис. 4).

4.Руководство программиста

4.1.Описание структуры программы

Класс TPoint будет реализован с помощью координат по X и по Y, а остальные классы будут использовать класс точки, как свои поля:

То есть для реализации всех структур данных нам потребуется 5 классов:

- Класс TPoint – точка;
- Класс TLine - отрезок;
- Класс TCircle - окружность;
- Класс TTriangle - треугольник;
- Класс TPlex – плекс.

А также проект использующий фреймворк Google Test, для проверки правильности работы этих классов и тестовый проект, который будет показываться пользователю.

Класс TPoint:

Класс точка задает работу с остальными классами, так как точку будут использовать в качестве своего поля остальные классы. У точки есть методы сравнения, который определяют совпадают ли две точки или нет.

Класс TLine:

Класс TLine содержит реализацию класс отрезок с использованием двух точек.

Класс TCircle:

Класс TCircle содержит реализацию класс окружность с использованием точки и радиуса.

Класс TTriangle:

Класс TTriangle содержит реализацию класс треугольник с использованием трех точек.

Класс TPlex:

Класс `TPlex` содержит реализацию класса плекс с указателями на две точки, которые при необходимости будут динамически преобразованы к указателям на другие плексы.

Класс `gtest`:

Класс `gtest` реализует тестирование всех классов по средствам фреймворка `Google Test`. Тесты пишутся для каждого метода классов, каждого ветвления этих методов и для всех возможных исключений этих методов.

Проект `figures`:

В данном проекте реализован проект использования геометрических фигур, который будет доступен пользователю.

4.2.Описание структур данных

Класс `TPoint`:

`double` `x` - координата по X;

`double` `y` - координата по Y.

Класс `TLine`:

`TPoint*` `point_1` – указатель на первую точку;

`TPoint*` `point_2` – указатель на вторую точку;

`bool` `del_flag_1` – флаг, который указывает была ли удалена первая точка;

`bool` `del_flag_2` – флаг, который указывает была ли удалена вторая точка;

Класс `TCircle`:

`TPoint*` `center` – указатель на центр окружности;

`double` `radius` – радиус окружности;

`Exceptions_from_figures` `exception` – переменная для вызова исключений;

Класс `TTriangle`:

`TPoint*` `a` – указатель на первую точку треугольника;

`TPoint*` `b` – указатель на вторую точку треугольника;

`TPoint*` `c` – указатель на третью точку треугольника;

`Exceptions_from_figures` `exception` – переменная для вызова исключений;

Класс `TPlex`:

`TPoint*` `right` – указатель на правую ветку плекса;

`TPoint*` `left` – указатель на левую ветку плекса;

Описание методов:

Метод:	Описание:
<code>TPoint::TPoint()</code>	Конструктор по умолчанию для класса <code>TPoint</code> .
<code>TPoint::TPoint(const double _x, const double _y)</code>	Конструктор с параметром для класса <code>TPoint</code> , который принимает координаты точки.
<code>TPoint::TPoint(const TPoint& point)</code>	Конструктор копирования класса <code>TPoint</code> .
<code>TPoint::~~TPoint()</code>	Деструктор для класса точка.
<code>bool TPoint::operator==(const TPoint & point)</code>	Перегрузка оператора сравнения для точки.
<code>bool TPoint::operator!=(const TPoint & point)</code>	Перегрузка оператора не равно для точки.
<code>void TPoint::SetX(const double _x)</code>	Метод, который позволяет изменить значение координаты по X.
<code>void TPoint::SetY(const double _y)</code>	Метод, который позволяет изменить значение координаты по Y.
<code>double TPoint::GetX()</code>	Метод, который возвращает текущее значение координаты по X.
<code>double TPoint::GetY()</code>	Метод, который возвращает текущее значение координаты по Y.
<code>void TPoint::Show()</code>	Метод, который выводит на экран значения координат точки.
<code>TLine::TLine()</code>	Конструктор по умолчанию для отрезка.
<code>TLine::TLine(const TPoint* _point_1, const TPoint* _point_2)</code>	Конструктор с параметром для отрезка, который принимает две точки.
<code>TLine::TLine(const double x1, const double y1, const double x2, const double y2)</code>	Конструктор с параметрами, который принимает координаты двух точек.
<code>TLine::TLine(const TLine& line)</code>	Конструктор копирования для класс отрезок.
<code>TLine::~~TLine()</code>	Деструктор для отрезка.
<code>void TLine::SetX(const double _x1)</code>	Установка координаты по X первой точки.
<code>void TLine::SetY(const double _y1)</code>	Установка координаты по Y первой точки.

<code>void TLine::SetX1(const double _x2)</code>	Установка координаты по X второй точки.
<code>void TLine::SetY1(const double _y2)</code>	Установка координаты по Y второй точки.
<code>double TLine::GetX()</code>	Возврат координаты по X первой точки.
<code>double TLine::GetY()</code>	Возврат координаты по Y первой точки.
<code>double TLine::GetX1()</code>	Возврат координаты по X второй точки.
<code>double TLine::GetY1()</code>	Возврат координаты по Y второй точки.
<code>TPoint* TLine::GetPoint1()</code>	Возврат указателя на первую точку.
<code>TPoint* TLine::GetPoint2()</code>	Возврат указателя на вторую точку.
<code>void TLine::SetPoint1(TPoint * point1)</code>	Установка указателя на первую точку.
<code>void TLine::SetPoint2(TPoint * point2)</code>	Установка указателя на вторую точку.
<code>void TLine::Show()</code>	Метод вывода координат отрезка на экран.
<code>TCircle::TCircle()</code>	Конструктор круга по умолчанию.
<code>TCircle::TCircle(const TPoint* point, double rad)</code>	Конструктор круга, который принимает точку и радиус.
<code>TCircle::TCircle(const double _x, const double _y, const double rad)</code>	Конструктор круга, который принимает координаты точки и радиус.
<code>TCircle::TCircle(const TCircle& circle)</code>	Конструктор копирования для круга
<code>TCircle::~~TCircle()</code>	Деструктор для круга.
<code>void TCircle::SetX(const double _x)</code>	Установить координату по X.
<code>void TCircle::SetY(const double _y)</code>	Установить координату по Y.
<code>void TCircle::SetR(const double rad)</code>	Установить радиус окружности.
<code>double TCircle::GetX()</code>	Вернуть координату по X.
<code>double TCircle::GetY()</code>	Вернуть координату по Y.
<code>double TCircle::GetR()</code>	Вернуть радиус окружности.
<code>void TCircle::Show()</code>	Метод, который выводит на экран координаты центра и радиус окружности.
<code>TTriangle::TTriangle()</code>	Конструктор по умолчанию для треугольника.
<code>TTriangle::TTriangle(TPoint* _a, TPoint* _b, TPoint* _c)</code>	Конструктор с параметрами для треугольника, который принимает три точки.
<code>TTriangle::TTriangle(const double _x1, const double _y1, const double _x2, const double _y2, const double _x3, const double _y3)</code>	Конструктор с параметрами для треугольника, который принимает координаты трех точек.

<code>TTriangle::TTriangle(const TTriangle& triangle)</code>	Конструктор копирования для треугольника.
<code>TTriangle::~~TTriangle()</code>	Деструктор для треугольника.
<code>void TTriangle::SetXA(const double _x1)</code>	Установить координату по X первой точки.
<code>void TTriangle::SetYA(const double _y1)</code>	Установить координату по Y первой точки.
<code>void TTriangle::SetXB(const double _x2)</code>	Установить координату по X второй точки.
<code>void TTriangle::SetYB(const double _y2)</code>	Установить координату по Y второй точки.
<code>void TTriangle::SetXC(const double _x3)</code>	Установить координату по X третьей точки.
<code>void TTriangle::SetYC(const double _y3)</code>	Установить координату по Y третьей точки.
<code>double TTriangle::GetXA()</code>	Вернуть координату по X первой точки.
<code>double TTriangle::GetYA()</code>	Вернуть координату по Y первой точки.
<code>double TTriangle::GetXB()</code>	Вернуть координату по X второй точки.
<code>double TTriangle::GetYB()</code>	Вернуть координату по Y второй точки.
<code>double TTriangle::GetXC()</code>	Вернуть координату по X третьей точки.
<code>double TTriangle::GetYC()</code>	Вернуть координату по Y третьей точки.
<code>void TTriangle::Show()</code>	Метод, который выводит на экран координаты треугольника.
<code>TPlex::TPlex()</code>	Конструктор по умолчанию для плекса.
<code>TPlex::TPlex(const T Plex& plex)</code>	Конструктор копирования плекса.
<code>TPlex::TPlex(TPoint* point1, TPoint* point2)</code>	Конструктор с параметрами для плекса, который принимает два указателя на точки.
<code>TPlex::~~TPlex()</code>	Деструктор плекса.
<code>void T Plex::Delete()</code>	Метод, который позволяет удалить плекс.
<code>TPlex& T Plex::operator+=(TLine& line)</code>	Метод, который позволяет добавить линию, при условии, что одна из точек этой прямой уже принадлежит плексу.
<code>void T Plex::Show()</code>	Метод, который выводит на экран все точки плекса.

4.3.Описание алгоритмов

Подробное описание некоторых методов

Конструктор копирования плекса:

- Создаются два указателя на плекса, к которым присваиваются динамически приведенные левая и правая точка текущего плекса.
- Если удастся привести левую точку к плексу, то конструктор копирования плекса вызывается рекурсивно для этой точки. Иначе создается новая точка.
- Та же операция проводится и для правой точки.

Добавление линии к плексу:

- Если в плексе не хранились элементы, то точки прямой присваиваются полям плекса.
- В противном случае, создаются два стека, в первый кладутся указатели на левую и правую точку плекса, а другой указатель на сам плекс два раза.
- Начинается бесконечный цикл, в котором забираются значения из стеков и кладутся в новые указатели. Указатель из второго стека пытается привести к плексу.
- Значение по указатель из первого стека сравнивается с значением точек переданной прямой. Если они не равны, то в стеки заносятся новые параметры. Если значения равны одной из точек (правой или левой), то цикл прекращается, а к этой точке приписывается новая.

6. Заключение

В заключении можно сказать, что все поставленные цели и задачи были выполнены, а именно: созданы все классы простейших геометрических фигур и класс ломанной прямой (плекс) с реализованными методами их хранения, а также написаны к ним тесты, и они успешно пройдены.

7.Литература

- Учебные материалы к учебному курсу «Методы программирования» - Гергель В.П.