**Федеральное агентство связи**

Ордена Трудового Красного Знамени

федеральное государственное бюджетное образовательное учреждение высшего образования   
«Московский технический университет связи и информатики»

Кафедра «Информатика»

Лабораторная работа №2: Основы объектно-ориентированного программирования

по дисциплине «Кроссплатформенные технологии программирования»

Java позволяет использовать объекты. В данной лабораторной работе необходимо использовать классы по одному на файл, чтобы описать, как эти объекты работают.

Выполнил: студент

группы БСТ1803

Шевцов Никита

Вариант №24

Москва 2020

**Содержание**

[1 Задание на разработку программы 3](#_Toc53411103)

[2 Разработанные функции и классы 4](#_Toc53411104)

[2.1 Класс Point3d 4](#_Toc53411105)

[2.1.1 Метод getCrd 4](#_Toc53411106)

[2.1.2 Метод void setCrd 5](#_Toc53411107)

[2.1.3 Метод boolean equals 5](#_Toc53411108)

[2.1.4 Метод double distanceTo 5](#_Toc53411109)

[2.2 Класс Lab1 5](#_Toc53411110)

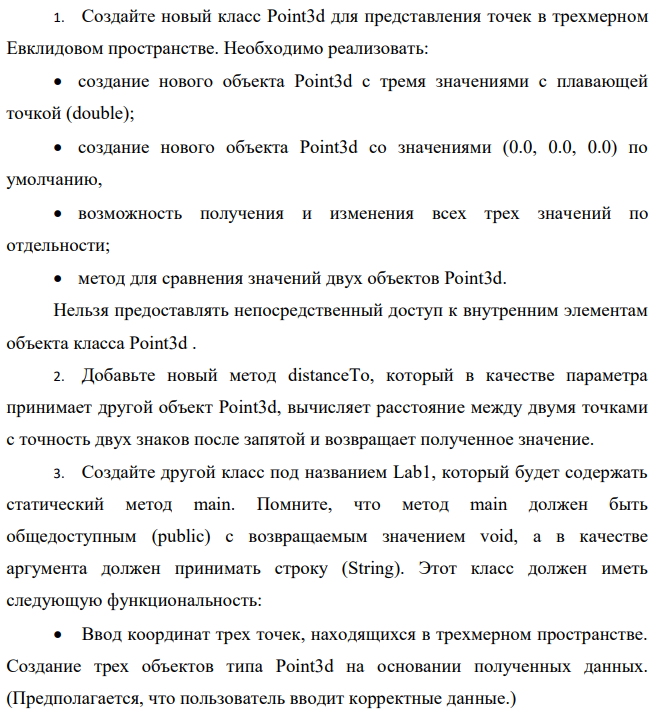
[2.2.1 Метод double computeArea 5](#_Toc53411111)

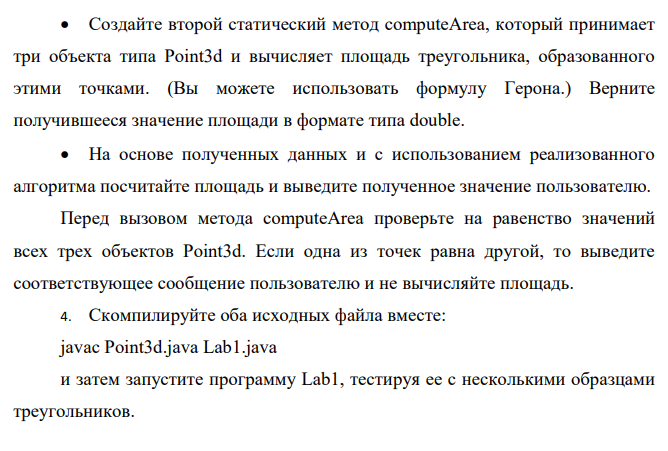
[2.2.2 Метод void main 6](#_Toc53411112)

[3 Код программы 6](#_Toc53411113)

[4 Результат работы 8](#_Toc53411114)

# Задание на разработку программы





# Разработанные функции и классы

## Класс Point3d

Класс описывающий трехмерную точку

* + 1. Метод getCrd

Возвращает выбранную координату

* + 1. Метод void setCrd

Устанавливает выбранную координату

* + 1. Метод boolean equals

Метод сравнения классов Point3d, возвращает **true** если точки совпадают, иначе **false**

* + 1. Метод double distanceTo

Метод возвращающий расстояние между двумя точками

## Класс Lab1

Класс возвращающий площадь треугольника, образованного точками в трехмерном пространстве. Входными параметрами являются строки, передающиеся через командную строку в формате x1 y1 z1 x2 y2 z2 x3 y3 z3, где x,y,z - соответствующие координаты. 1,2,3 - соответствуют номеру точки, к которой относится координата.

* + 1. Метод double computeArea

Метод вычисляющий площадь треугольника по формуле Герона

* + 1. Метод void main

Осуществляет парсинг точек из входных данных, проверку того, что входные точки не совпадают и выводит площадь введённого треугольника.

# Код программы

// Класс, возвращающий площадь треугольника в трехмерном пространстве, заданного по точкам

**public** **class** Lab1 {

**public** **static** **void** main(String[] args) {

Point3D[] points = **new** Point3D[3]; // Массив трех точек треугольника

String[] crds = **new** String[3]; // Массив координат точки

**for** (**int** i = 0; i < 3; i++) {

**for** (**int** j = 0; j < 3; j++) {

crds[j] = args[3\*i + j]; // Сформировать координаты одной точки из args

}

points[i] = **new** Point3D(crds); // Создать по ним точку

}

**if** (!points[0].equals(points[1]) & !points[0].equals(points[2]) & !points[1].equals(points[2])) System.***out***.print(*computeArea*(points));

**else** System.***out***.print("Две точки одинаковы, невозможно расчитать площадь треугольника.");

}

**public** **static** **double** computeArea(Point3D[] points) { // Вычисление площади треугольника по формуле Герона

**double** A, B, C, p;

A = points[0].distanceTo(points[1]);

B = points[1].distanceTo(points[2]);

C = points[0].distanceTo(points[2]);

p = (A + B + C)/2;

**return** Math.*sqrt*(p\*(p - A)\*(p - B)\*(p - C));

}

}

/\*\*

\* Класс, представляющий точку в 3D пространстве

\* **@author** niksh

\*/

**public** **class** Point3D {

/\*\* Массив координат, где [0] - x, [1] - y, [2] - z \*/

**private** **double**[] coordinates = **new** **double**[3];

/\*\*

\* Конструктор - создание точки с определенными координатами

\* **@param** crds - координаты точки в 3D пространстве (можно указать только x, или только x y)

\*/

**public** Point3D(String[] crds) {

**int** i;

**for** (i = 0; i < crds.length; i++) { // Заполнить массив координат по введенным данным

**this**.coordinates[i] = Double.*parseDouble*(crds[i]);

}

**for** (i += 0; i < 3; i++) { // Заполнить пустые координаты 0.0

**this**.coordinates[i] = 0.0;

}

}

/\*\*

\* Получить координату точки

\* **@param** crd - указатель на координату, x, y или z

\* **@return** Запрошенная координата

\*/

**public** **double** getCrd(**char** crd) {

**switch** (crd) {

**case** ('x'):

**return** **this**.coordinates[0];

**case** ('y'):

**return** **this**.coordinates[1];

**case** ('z'):

**return** **this**.coordinates[2];

}

**return** 0.0;

}

/\*\*

\* Изменить указанную координату точки

\* **@param** crd - указатель на координату, x, y или z

\* **@param** val - значение координаты после изменения

\* **@return** <code>true</code>, если изменение успешно, <code>flase</code> в противном случае

\*/

**public** **boolean** setCrd(**char** crd, **double** val) {

**switch** (crd) {

**case** ('x'):

**this**.coordinates[0] = val;

**return** **true**;

**case** ('y'):

**this**.coordinates[1] = val;

**return** **true**;

**case** ('z'):

**this**.coordinates[2] = val;

**return** **true**;

}

**return** **false**;

}

/\*\*

\* Проверить, являются две точки одним и тем же объектом, либо имеют одинаковые координаты

\* **@param** point - точка для сравнения

\* **@return** <code>true</code>, если являются, <code>false</code> в проивном случае

\*/

**public** **boolean** equals(Point3D point) {

**boolean**[] eqcrds = **new** **boolean**[3];

**for** (**int** i = 0; i < 3; i++) {

eqcrds[i] = (**this**.coordinates[i] == point.coordinates[i]);

}

**if** (eqcrds[0] & eqcrds[1] & eqcrds[2]) **return** **true**;

**return** **false**;

}

/\*\*

\* Вычисление расстояния между двумя точками с точностью 0.01

\* **@param** point - точка, до которой измеряется расстояние

\* **@return** Расстояние между точками

\*/

**public** **double** distanceTo(Point3D point) {

**double** dec = 100;

**double** distance = Math.*sqrt*(Math.*pow*(**this**.coordinates[0] - point.coordinates[0], 2) + Math.*pow*(**this**.coordinates[1] - point.coordinates[1], 2) + Math.*pow*(**this**.coordinates[2] - point.coordinates[2], 2))\*dec;

**double** distancernd = Math.*round*(distance)/dec;

**return** distancernd;

}

}

# Результат работы

Входные аргументы



Результат работы

