МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ЭКОНОМИКИ И СЕРВИСА

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ  
ПО УЧЕБНОЙ ПРАКТИКЕ ПО ПОЛУЧЕНИЮ ПЕРВИЧНЫХ ПРОФЕССИОНАЛЬНЫХ УМЕНИЙ И НАВЫКОВ

Студент

гр. БИС–18–01 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Я.И. Илюшин

Руководитель

канд. экон. наук, доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Ивин

Владивосток 2020

Задание

Используя объектно-ориентированную методологию разработки программных продуктов, создайте кроссплатформенное приложение с оконным интерфейсом, реализующее алгоритм в соответствии с вариантом задания.

Используя произвольные языки программирования и среду разработки создайте программу, которая отображает на экране монитора график кривой или поверхности (в соответствии с вариантом задания) в декартовой и полярной системах координат[[1]](#footnote-1) с центром в центре экрана монитора (окна или иной прямоугольной области экрана). При изменении размеров окна, график и все его атрибуты (координатная сетка, метки на шкале, подписи и т.д.) должны автоматически масштабироваться.

Параметры уравнения кривой или поверхности должны вводиться в специально отведённые ячейки экранной формы программы: Кубика.

Используя результаты предыдущего задания создайте анимацию примитива, движущегося по траектории построенной кривой: в качестве примитива используется закрашенный квадрат синего цвета со стороной a > 2.

**Содержание**

[Задание 2](#_Toc46392458)

[Введение 5](#_Toc46392459)

[1 Анализ технического задания 5](#_Toc46392460)

[2 Алгоритм программы 6](#_Toc46392461)

[3 Выбор языка программирования и среды разработки 7](#_Toc46392462)

[3.1 Выборязыкапрограммирования 7](#_Toc46392463)

[3.2 Выбор среды разработки 8](#_Toc46392464)

[4 Программный продукт «Cubic Curve» 8](#_Toc46392465)

[5 Принцип работы кода в программе 13](#_Toc46392466)

[6 Тестирование программы 17](#_Toc46392467)

[6.1 Тестирование в нормальных условиях 17](#_Toc46392468)

[6.2 Стресс**-**тестирование 18](#_Toc46392469)

[7 Инструкция пользователя 19](#_Toc46392470)

[7.1 Описаниеполей 19](#_Toc46392471)

[7.2 Описаниекнопок 19](#_Toc46392472)

[Заключение 20](#_Toc46392473)

[Список использованных источников 21](#_Toc46392474)

[Приложение А 22](#_Toc46392475)

[22](#_Toc46392476)

Введение

Для интеграции приобретённых в процессе теоретического обучения в ВУЗе профессиональных и общекультурных знаний и навыков важен опыт самостоятельной профессиональной деятельности, направленный на закрепление теоретической подготовки обучающегося, приобретение им практических навыков и компетенций, а также опыта самостоятельной профессиональной деятельности. Таким образом, главными целями практики являются: приобретения первичных практических навыков и компетенций в сфере профессиональной деятельности.

Для достижения поставленных целей, были определены и решены следующие задачи:

* получить первичные профессиональные умения и навыки;
* научиться анализировать и обобщать результаты научно-исследовательской работы с использованием современных достижений науки и техники;
* подготовиться к осознанному и углублённому изучению специальных дисциплин;
* овладеть знаниями о видах, структуре, организации, основных методах ведения научно-исследовательской работы;
* сформировать стремление к самосовершенствованию и повышению культурного уровня;
* развить умения самоорганизации и самоконтроля.

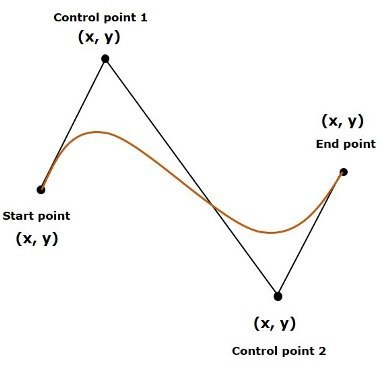
1 Анализ технического задания

Кубика – это плоская алгебраическая кривая 3-го порядка, то есть множество точек плоскости, заданных кубическим уравнением.

Уравнение (1) может быть записано в виде:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Кубическая кривая в двухмерной плоскости рисуется с использованием четырех точек – начальной точки, двумя управляющими и конечной точкой (рисунок 1.1).



Источник: [1]

Рисунок 1.1 – Кубическая кривая

Начальная и конечная точки определяют протяженность кубики, а управляющие точки определяют ее искривление.

2 Алгоритм программы

Для построения кубики по формуле (1), составим блок-схему алгоритма, которая находится в Приложении А.

Изначально происходит создание обработчика просчета и отрисовки систем координат, у которого затем вызывается процедура отрисовки декартовой системы координат, которая отображается на форме по умолчанию. После чего происходит считывание значения ComboBox, если он имеет значение «Polar», то у обработчика вызовется процедура отрисовки полярной координатной сетки, а если «Descartes», то вызовется процедура отрисовки декартовой координатной сетки. Далее по алгоритму, ожидается нажатие на кнопку «Create» пользователем, если кнопка нажата, то начнется отрисовка кубической кривой вместе с системой координат, в ином случае ничего не произойдет. По завершению отрисовки графика произойдет проверка на активацию кнопки «Start», которая при активации вызовет процедуру расчета координат кривой и запустит синий квадрат по вычисленной траектории, иначе алгоритм завершится.

В предыдущей блок-схеме алгоритма программы используются также и классы. Составим блок-схему алгоритма для класса, отвечающего за расчет точек кривой (рисунок 2.1).

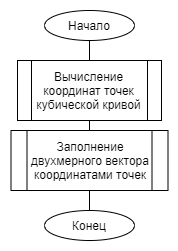


Рисунок 2.1 – Блок-схема алгоритма расчета точек кубической кривой

При запросе к классу в цикле происходит запрос метода, отвечающего за вычисление координат точек кубической кривой и заполнение ими массива, который вернется в месте вызова в изначальной функции.

Также составим блок-схему алгоритма для класса, отвечающего за просчет и отрисовки систем координат (рисунок 2.2).

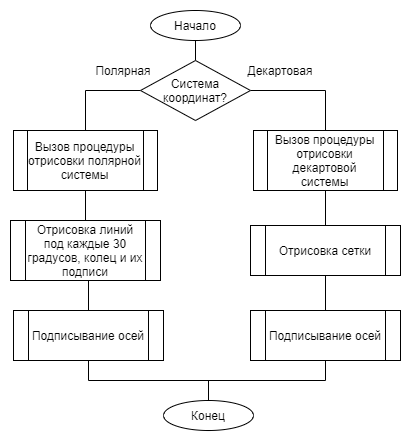


Рисунок 2.2 – Блок-схема алгоритма расчета точек кубической кривой

При запросе к классу необходимо определиться какую систему координат нам необходимо просчитать и отрисовать. После этого запустится для каждой системы координат очередь из логических блоков, которые просчитают и отрисуют нужную систему координат вместе с подписями.

3 Выбор языка программирования и среды разработки

3.1 Выборязыкапрограммирования

По условию задания программа должна использовать объектно-ориентированную методологию разработки программных продуктов, быть кроссплатформенной и иметь оконный интерфейс. Решением, которое идеально удовлетворяет все пункты задачи, стала Java. Java является объектно-ориентированным языком программирования, который компилируется в байт-код и запускается на виртуальных машинах JVM, что делает Java еще и кроссплатформенной технологией, поскольку программы на данном языке не требуется перекомпиливать для запуска на других платформах. Для создания оконного приложения будет использоваться платформа основанная на Java – JavaFX. Основной особенностью платформы является огромная кроссплатформенность. Приложения созданные на JavaFX могут запускаться на Linux, Windows, Solaris, Mac OS, а также на мобильных платформах.

3.2 Выбор среды разработки

Средой разработки была выбрана IntelliJ IDEA, потому что она является одним из самых мощных инструментов на рынке. Ключевыми особенностями для меня стали – возможность быстро развернуть проект, огромное количество подсказок при написании кода, а также простота создания JavaFX приложения.

Также я воспользовался конструктором форм для JavaFX – JavaFX Scene Builder. Он позволяет в удобной форме настроить поведения оконного приложения без необходимости открывать fxml файл.

4 Программный продукт «Cubic Curve»

Разработанное программное обеспечение «Cubic Curve» предназначено для построения графика кривой кубики, с возможностью ввода нескольких параметров.

Используя скачанный конструктор форм JavaFX Scene Builder, был сделан простой интерфейс, в котором находятся такие элементы, как: Button, ComboBox, Label, TextBox, Pane и AcnhorPane. Окно программы включает в себя Label с названием кривой, поля для ввода данных с названиями точек, они уже содержат некоторые значения для быстрой проверки работоспособности программы. Также две кнопки, первая кнопка с наименованием «Create», отвечающая за построение графика кривой, а вторая кнопка с наименованием «Start» отвечает за запуск анимированного примитива по координатам кривой. Также на форме присутствует ComboBox с элеиентами «Descartes» и «Polar», который отвечает за переключение вариантов отрисовок систем координат с декартовой системы в полярную и наоборот (рисунок 4.1).

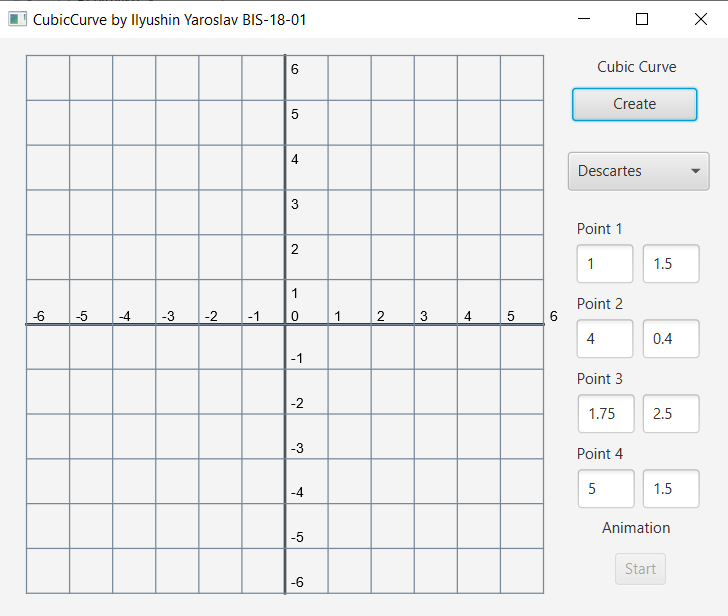


Рисунок 4.1 –Интерфейс программы

Для того, чтобы построить кривую, требуется ввести новые значения для точек в TextBox или же ничего не менять и использовать уже заданные значения. После этого следует нажать на кнопку «Create», после чего пользователь увидит систему координат с кривой по формуле (1) (рисунок 4.2).

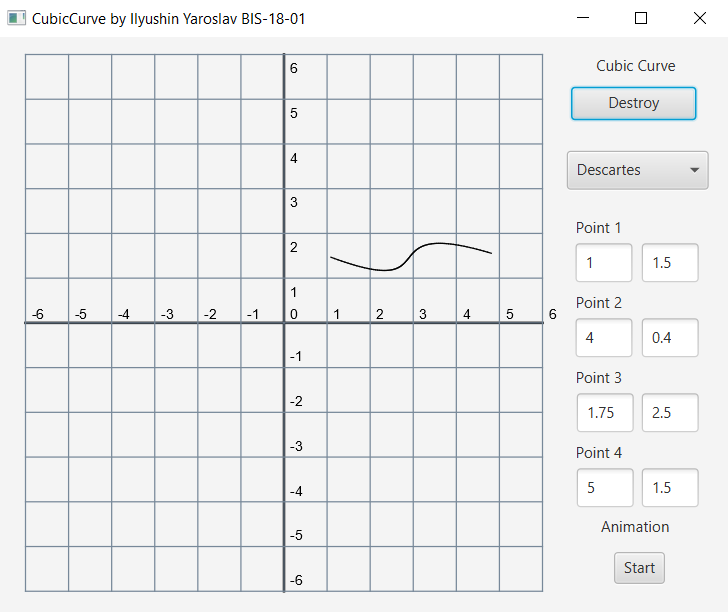


Рисунок 4.2 – Отрисованная кривая

После того как кривая будет построена на интерфейсе станет доступна кнопка «Start». Чтобы запустить примитив по траектории кривой необходимо нажать на нее, в качестве примитива используется закрашенный синий квадрат со стороной a>2 (рисунок 4.3).

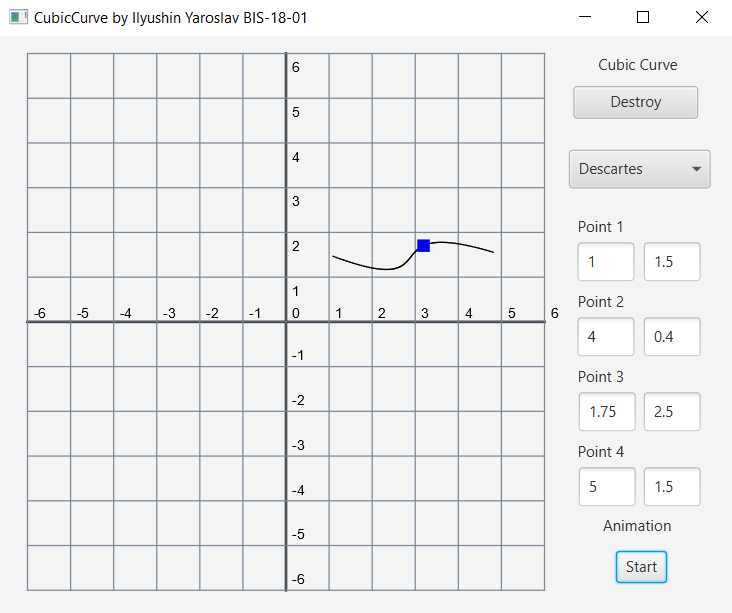


Рисунок 4.3 – Движущейся примитив по траектории кривой

При попытке нажатия на кнопку «Create» без заполнения некоторых полей данными, программа сбросит значения для всех TextBox на форме (рисунок 4.4).

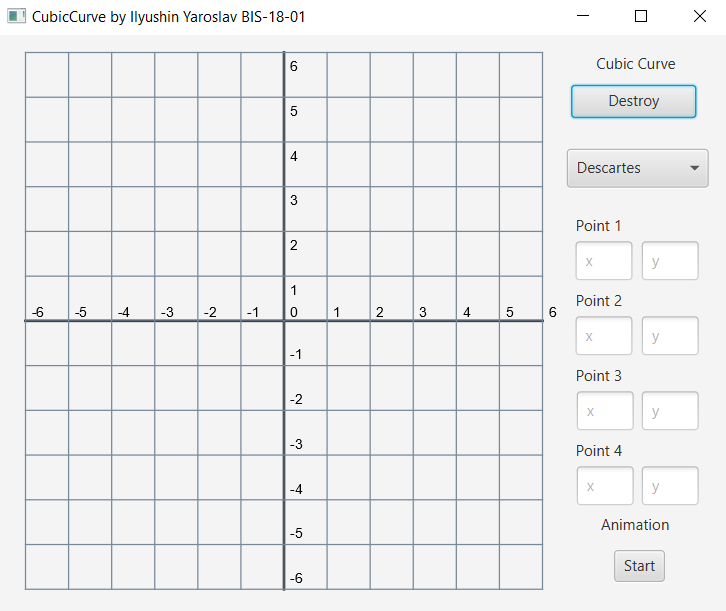


Рисунок 4.4 – Ошибка в случае не введённых данных

При попытки пользователем растянуть края формы приложения, все объекты, которые находятся на форме включая саму форму программы, будут масштабироваться в зависимости от размера формы (рисунок 4.5).

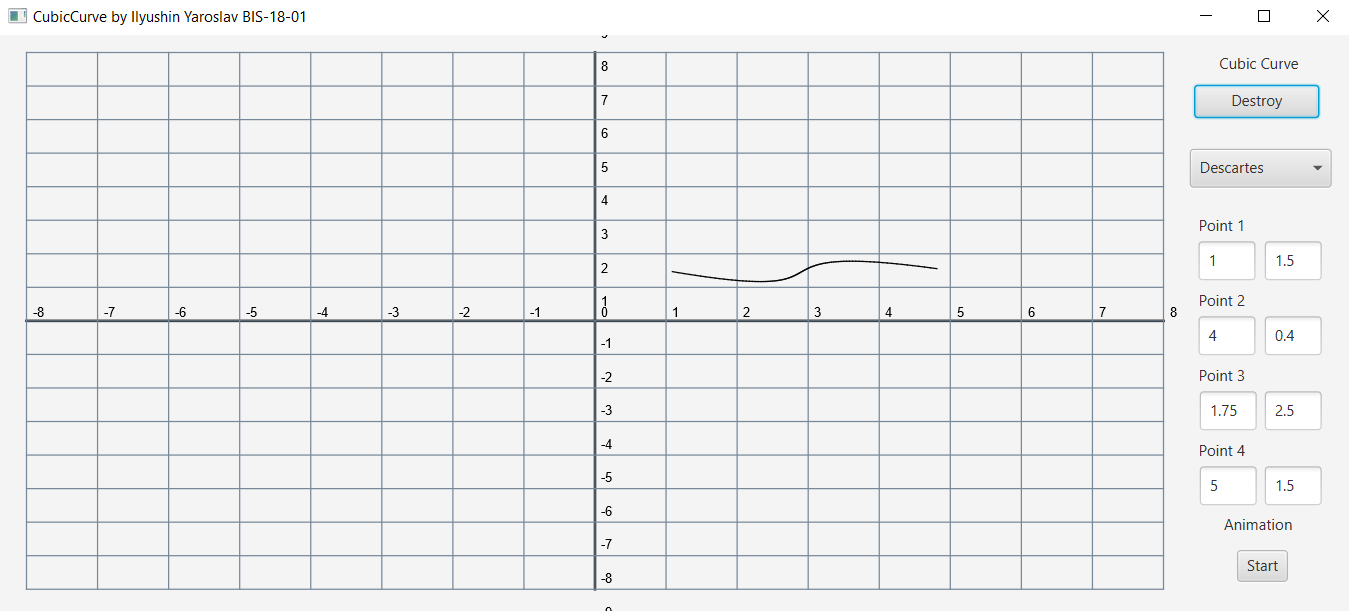


Рисунок 4.5 – Масштабирование программы

Для того, чтобы построить кривую в полярной системе координат, пользователю требуется выбрать элемент «Polar» в ComboBox, после чего координатная система изменится на полярную (рисунок 4.6).

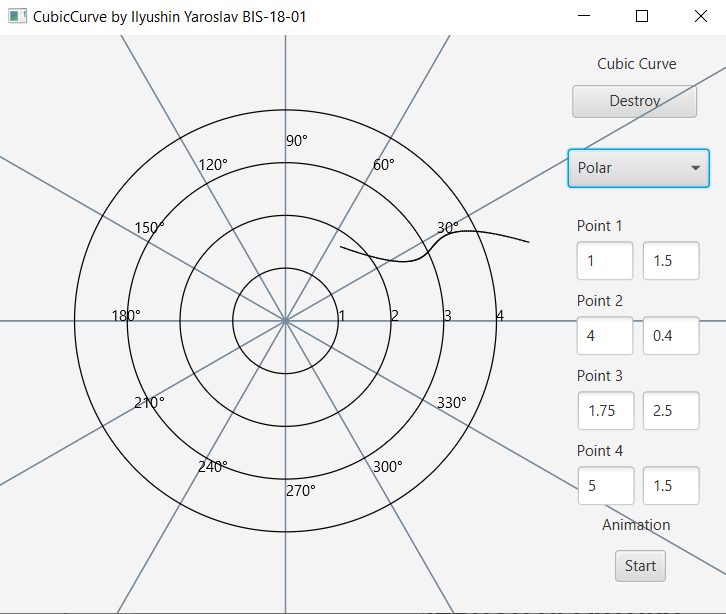


Рисунок 4.6 – Кривая в полярной системе координат

Результатом является построение полярной системы координат с отрисованным графиком кривой с заданным пользователем точками, либо просто просто полярная система с возможностью построить кривую.

5 Принцип работы кода в программе

В программе «Cubic Curve» присутствуют TextBox, которые нужны для ввода точек, с помощью которых строится кубическая кривая.

В коде программы отдельным переменным присваиваются значения из этих TextBox (рисунок 5.1).

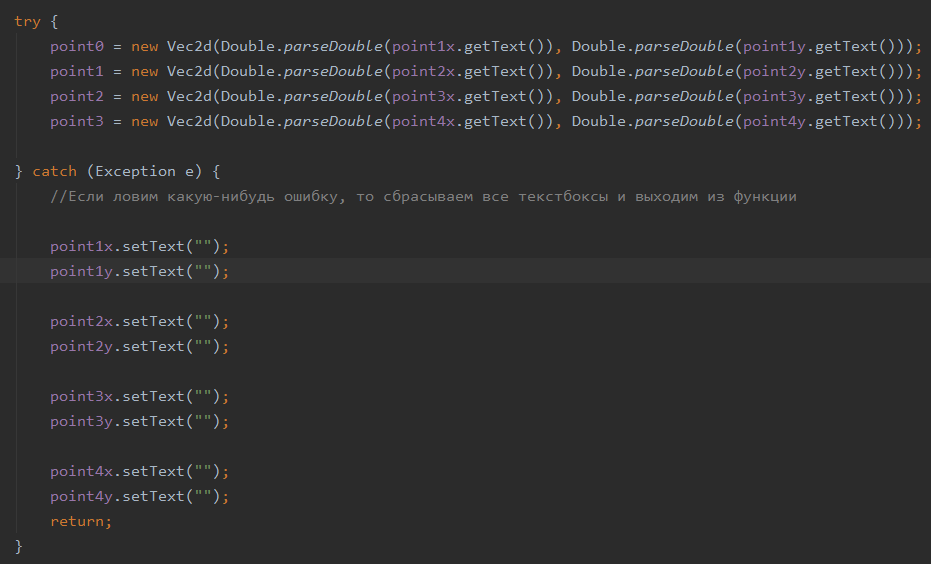


Рисунок 5.1 – Код для присваивания значений в переменные

Вызывается процедура, в которой анализируется какая сейчас система координат должна отражаться, вызывается процедура ее отрисовки, затем с помощью данных, которые пользователь ввел в необходимые для этого поля ввода, программа с помощью формулы (1), введенных пользователем данных и параметрических функций в отдельном классе «CubicCurve.java», в методе «getCubicCurveCoordinatesMass()» рассчитывает необходимые точки координат X и Y и заполняет ими Vec2 (рисунок 5.2), затем по этим координатам строится кривая (рисунок 5.3).

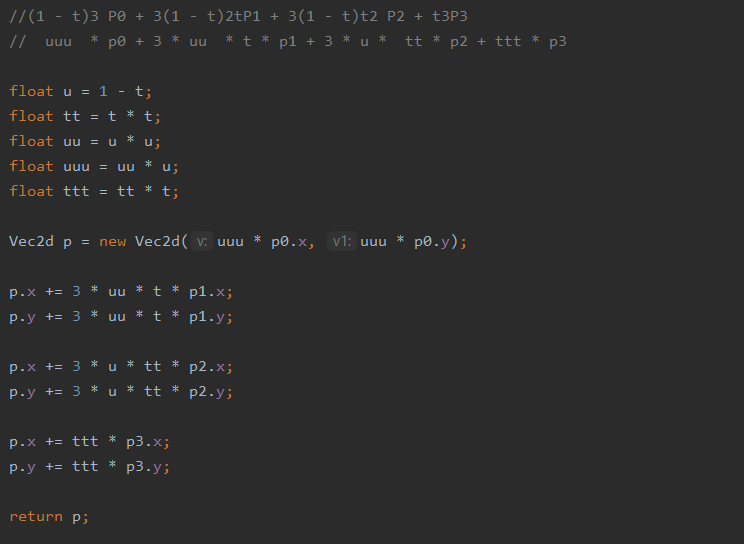


Рисунок 5.2 – Код расчёта координатных точек кривой

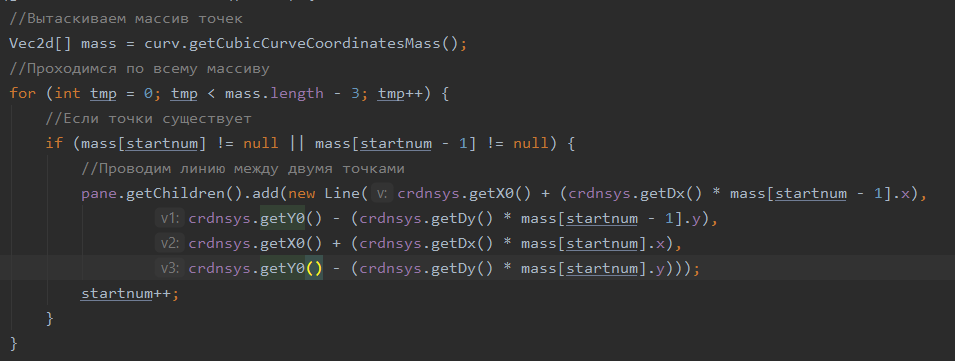


Рисунок 5.3 – Код отрисовки координатных точек кривой

Примитив отрисовывается с помощью таймера (AnimationTimer), который релаизован в коде. Для этого вызывается функция QuadDraw(), в которой вычисляются координаты точек кубической кривой, затем она рисует примитив на определенной вычисленной точке, которой и соответствует значение индекса. Сам же индекс увеличивается с каждым «тиком» таймера. Если значение индекса выходит за границы количества точек кривой, то таймер останавливается, переменная с индексом аннулируется, а Pane перерисовывается уже без квадрата. (рисунок 5.4; 5.5).

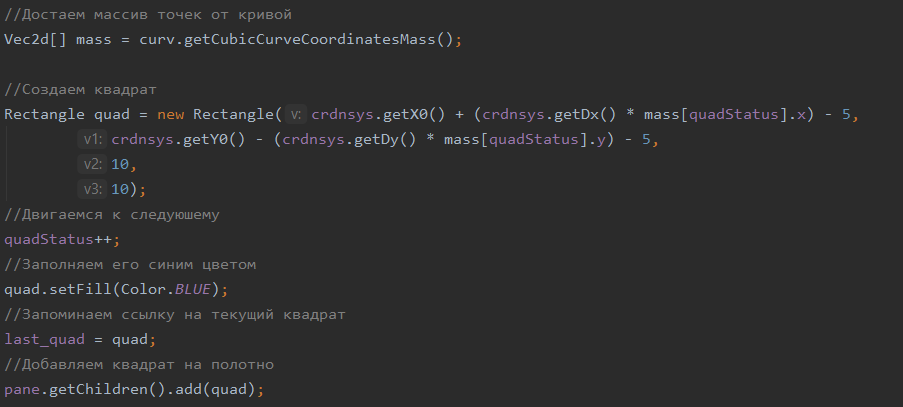


Рисунок 5.4 – Код, отвечающий за построение примитива

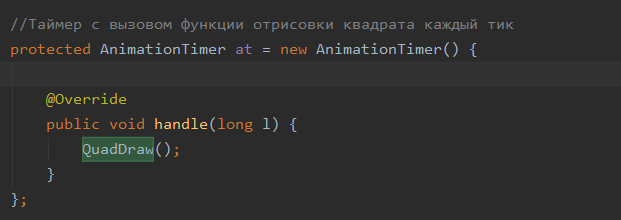


Рисунок 5.5 – Код работы таймера за «тик»

В методе «DrawDescartes()» в классе «CoordinateSystem.java» совершается прорисовка декартовой координатной сетки (рисунок 5.6).

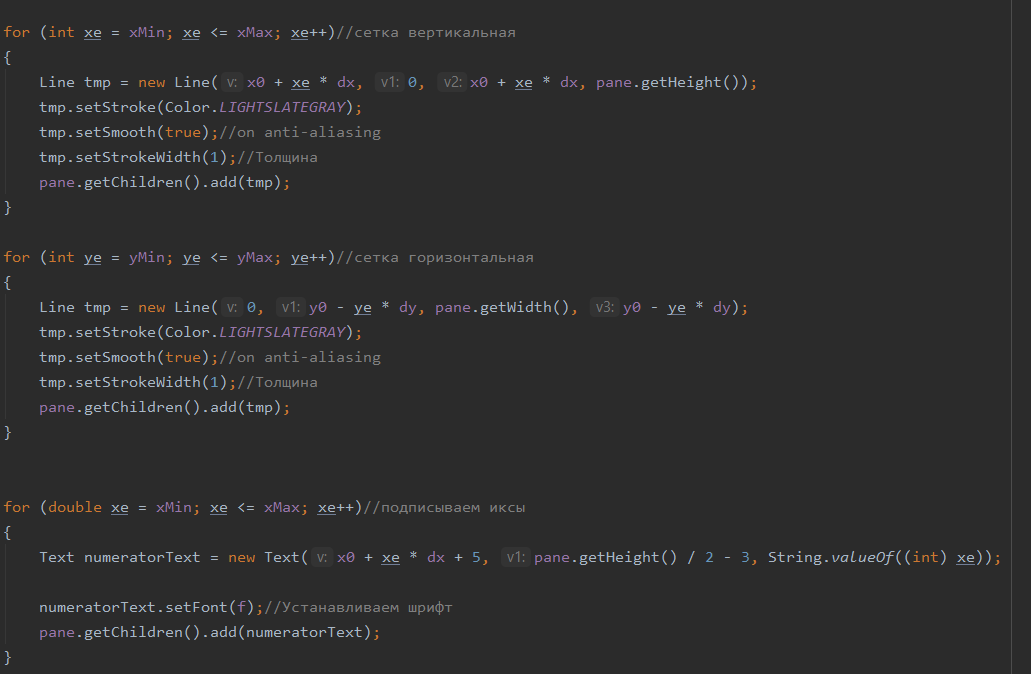


Рисунок 5.6 – Код отрисовки декартовой системы координат

Отрисовка координатной плоскости производится от левого края Pane до правых его границ.

В методе DrawPolar() совершается прорисовка полярной координатной сетки (рисунок 5.7).

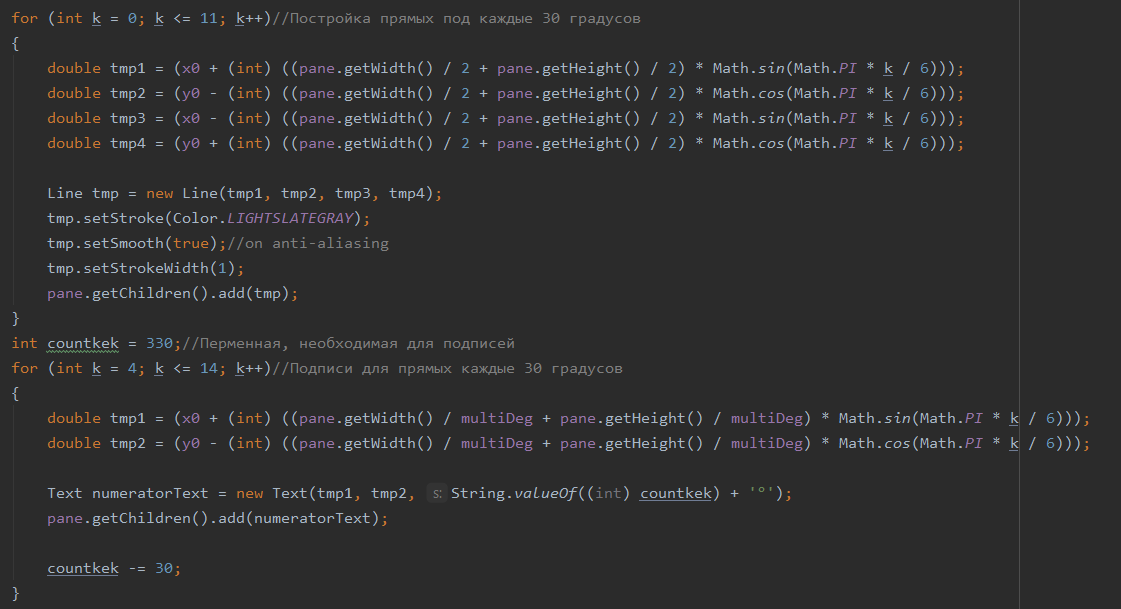


Рисунок 5.7 – Код отрисовки полярной системы координат

Произведем тестирование полученного программного кода.

6 Тестирование программы

6.1 Тестирование в нормальных условиях

Данный тест подразумевает, что пользователь не будет ошибаться при вводе данных в программе. Имеются в виду исключительно числа (в том числе и с плавающей запятой), без любых символов, кроме предусмотренных.

При корректных значениях при запуске программы, она работает в нормальном режиме (рисунок 6.1).

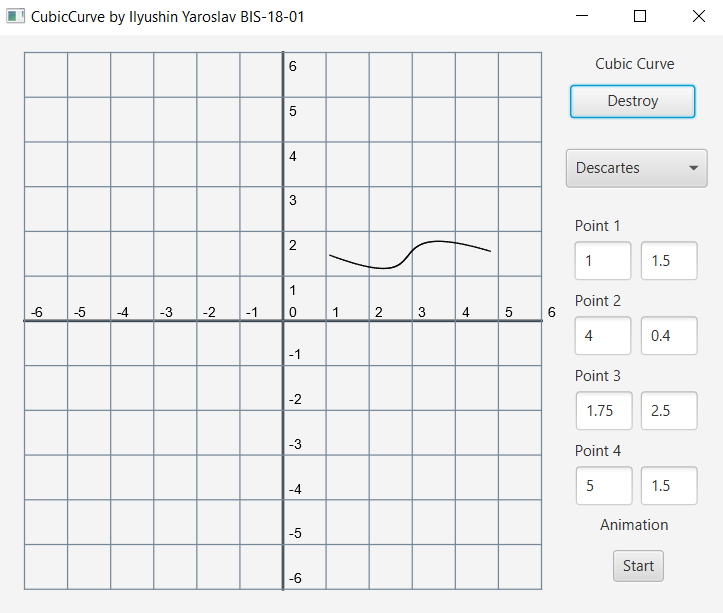


Рисунок 6.1 – Окно с отрисовкой графика кривой

Результатом работы программы в нормальном режиме является отрисованный график кривой.

6.2 Стресс**-**тестирование

Если при вводе будут допущены ошибки, то программа сбросит значения полей, введённых пользователем (рисунок 6.2).

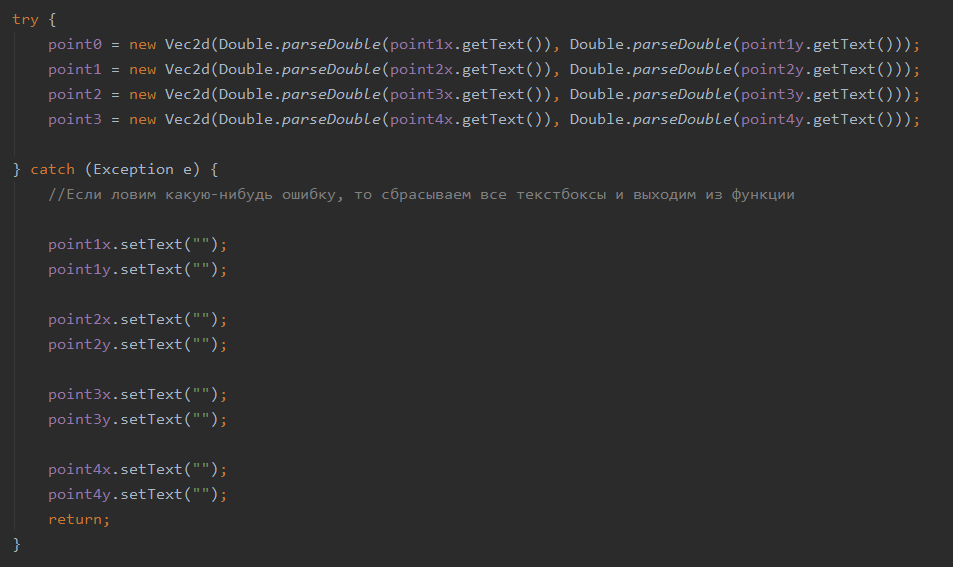


Рисунок 6.2 – Код обработки значений на корректность ввода

Результатом работы данного кода является обработка значений на корректность ввода. У пользователя не будет возможности аварийно завершить программу путем ввода некорректных значений.

7 Инструкция пользователя

7.1 Описаниеполей

Поля в программе расположены в соответствии с рисунком 4.1 и соответствуют ключевым четырём координатам точек.

7.2 Описаниекнопок

В программе имеются 2 кнопки и один ComboBox. Кнопка «Create» на форме программы отвечает за построение графика вместе с выбранной пользователем системой координат в ComboBox. Если же кривая отрисована, то кнопка изменит название на «Destroy» и при нажатии отрисует систему координат, выбранную пользователем, но уже без кривой. ComboBox предлагает пользователю выбрать систему координат, в каждой из которых можно отрисовать график кривой. Кнопка «Start» станет доступной в интерфейсе, если график вместе с системой координат отрисован на Pane. При нажатии запустится анимация примитива, движущегося по траектории построенной кривой.

Заключение

В процессе прохождения практики была разработана программа «Cubic Curve». Эта программа предназначена для построения графика кривой «Кубика», которая зависит от параметрических данных.

Также в процессе создания программы были освоены принципы работы с оконными приложениями, имеющими графический интерфейс, а также закреплены базовые принципы ООП.

Благодаря курсу учебной практики были освоены принципы объектно-ориентированного программирования, что востребовано при работе с массивными и сложными программами. А также стало понятно, как и в каких источниках нужно искать определенную информацию, связанной со специальностью.

Список использованных источников

1. JavaFX – 2D Shapes CubicCurve [Электронный ресурс] // TurotialsPoint. – Режим доступа: https://www.tutorialspoint.com/javafx/2dshapes\_cubic\_curve.htm. (Дата обращения – 22.07.2020).

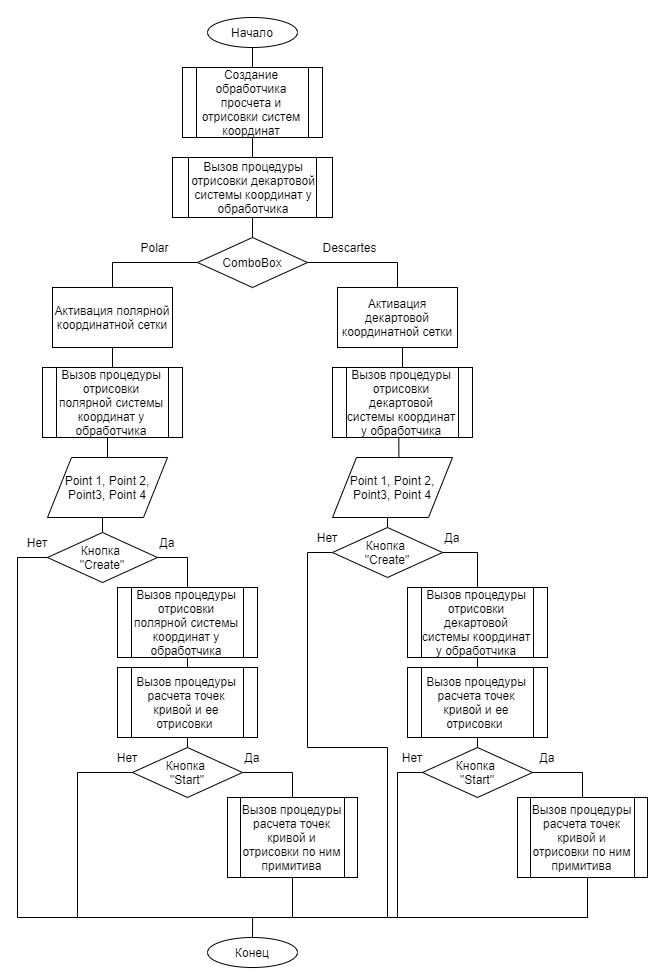
2. Кубика // Википедия – интернет-энциклопедия со свободным контентом [Электронный ресурс] // Wikipedia. – Режим доступа: https://ru.wikipedia.org/wiki/Кубика. (Дата обращения –22.07.2020).

3. JavaFX [Электронный ресурс] // JavaFX. – Режим доступа: https://openjfx.io/. (Дата обращения –22.07.2020).

4. СК-СТО-ТР-04-1.005-2015. Требования к оформлению текстовой части выпускаемых квалификационных работ, курсовых работ (проектов), рефератов, контрольных работ, отчетов по практикам, лабораторным работам [Электронный ресурс] // Библиотека ВГУЭС – Режим доступа: https://studfiles.net/preview/5919843/. (Дата обращения – 22.07.2020).

Приложение А

Блок-схема программы



1. если уравнение кривой в одной из систем координат отсутствует, то оно может быть заменено уравнением в параметрической форме [↑](#footnote-ref-1)