МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ КОЗЬМЫ МИНИНА»

Факультет Информационных технологий

Кафедра Прикладной информатики и информационных технологий в образовании

Направление подготовки (специальность): 09.03.02 - Информационные системы и технологии

Профиль (специализация): Информационные системы и технологии

**К У Р С О В О Й П Р О Е К Т**

на тему: Линейная аппроксимация функции

ОБУЧАЮЩИЙСЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Брунов Д. Д.

(личная подпись) (инициалы, фамилия)

РУКОВОДИТЕЛЬ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_к.п.н., доцент Круподерова Е.П.

(личная подпись) (ученая степень, звание, инициалы, фамилия)

Нижний Новгород - 2023 г.

Содержание

[Введение 2](#_Toc104297901)

[Постановка задачи 5](#_Toc104297902)

[Описание метода решения задачи 8](#_Toc104297903)

[Разработка алгоритма решения задачи 10](#_Toc104297904)

[Выбор систем программирования 12](#_Toc104297905)

[Описание программного комплекса 16](#_Toc104297906)

[Анализ результатов 25](#_Toc104297907)

[Заключение 30](#_Toc104297908)

[Список литературы 31](#_Toc104297909)

[Приложение 36](#_Toc104297910)

# Введение

Аппроксимация (от лат. *proxima* - ближайшая) - это научный метод, заключающийся в нахождении выражения, приближенного к исходным каким-нибудь математическим объектам (например, чисел или функций) посредством нахождения более упрощенных и удобных в использовании или более известных. В научных целях аппроксимация используется для анализирования, описания функций, обобщения и дальнейшего использования эмпирических результатов.

Известно, что между различными величинами возможна точная или, правильнее сказать, функциональная связь. Функциональная связь - это определенная взаимосвязь явлений, где изменение одного из явлений приводит к преобразованию другого. Также существует менее точная корреляционная связь. В корреляции одному значению аргумента соответствует другое, приближенное значение или даже несколько значений функции (множество), в разной мере приближенных друг к другу. При проведении исследований в области науки, обработке результатов наблюдения или опыта нам, как правило, приходится обращаться ко второму варианту. При исследовании количественных зависимостей результатов показателей, значения которых находятся опытным путем, обычно присутствует некая их изменчивость. От части значения неустойчивы из-за переменчивости самих исследуемых объектов неживой или, тем более, живой природы, также это частично зависит от погрешностей наблюдения и количественной обработке материалов. К сожалению, несмотря на повсеместное развитие технологий, люди до сих пор не всегда могут исключить погрешности исследования полностью, а могут лишь уменьшить ее за счет тщательного выбора метода, которым будет проводиться исследование, и благодаря аккуратной работе. Как следствие, если необходимо заняться научно-исследовательской работой, то сразу возникает проблема, связанная с выявлением истинного характера зависимостей изучаемых показателей, в разной мере выполнении скрытых неучтенных изменчивых значений. Для того, чтобы выявить закономерность показателей или упростить вычисление функций, связывающих различные точки (значения), и применяется аппроксимация – приблизительное описание корреляционной связи переменных, используемых в уравнении функции и передающих тенденцию зависимости, называемую трендом.

Выбирая аппроксимацию, следует опираться на цель исследования. Как правило, чем проще уравнение, используемое для аппроксимации, тем менее точные получаются результаты исследования. Поэтому важно сразу определиться, насколько критичны отклонения результатов, полученных после аппроксимации, выявления тренда, от реальных значений. При использовании более сложных аппроксимирующих уравнений можно добиться большей точности по отношению к эмпирическим значениям. Но надо учитывать, что нет никакого смысла стремиться к максимальной точности, ведь никто не может учесть все случайные отклонения величин данных, полученных опытным путем. В абсолютном большинстве случаев намного более важно уловить закономерность изменения значений, которые в конкретном случае наиболее логично и с неплохой точностью выражаются именно при помощи уравнения степенной функции.

Аппроксимация называется точечной (дискретной) в случае, если она строится на дискретном наборе значений, в ином случае аппроксимация считается непрерывной (интегральной).

Если имеются эмпирически полученные данные, заданные в таблице, то задача линейной аппроксимации заключается в нахождении прямой, которая опишет максимально точную зависимость, которой подчиняются все изначальные точки.

Метод наименьших квадратов (МНК) – заключается в проведении аппроксимирующей прямой таким образом, чтобы сумма квадратов ее отклонений от изначальных данных была наименьшей.

Подытоживая введение курсового проекта, следует отметить, что, выбирая метод аппроксимации, исследователь всегда идет на компромисс, отдавая предпочтение, где это уместно, неточности показателей, для обобщенного выражения зависимости сопоставляемых переменных. В попытках найти общие закономерности, придется опустить случайные отклонения эмпирических данных. Аппроксимация, в частности линейная, помогает при решении других задач, таких как нахождение неизвестных, по тем или иным причинам, переменных, используя интерполяцию или экстраполяцию; формализует найденную зависимость.

Цель проекта: разработать программу на языке программирования С# для нахождения аппроксимирующей функции при помощи метода наименьших квадратов.

Задачи проекта:

1. изучить справочную и учебную литературу по данной теме;

2. составить алгоритм и написать программу для нахождения аппроксимирующей функции при помощи метода наименьших квадратов;

3. выполнить проектирование формы, подобрать компоненты для реализации программы;

4. провести отладку программы;

5. выполнить анализ результатов.

# Постановка задачи

В современной жизни все постепенно сводится к упрощению. Все технологии постепенно становятся более «дружелюбными» для пользователей. Упрощение использования особенно заметно в программах, операционных системах.

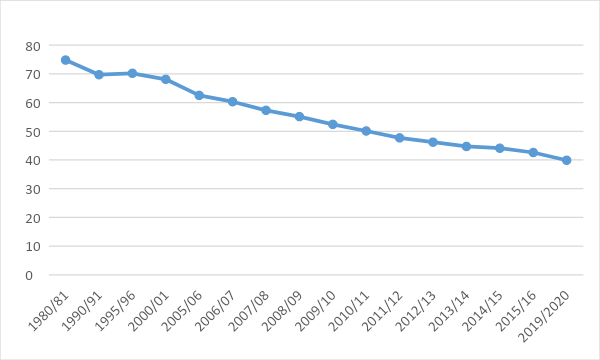
В научных исследованиях тоже используется упрощение. Одним из способов облегчить анализ как раз и является аппроксимация. Она позволяет изучить различные значения и качественные свойства объекта, сводя задачу к исследованию наиболее простых или наиболее удобных объектов.

2023 год объявлен годом педагога и наставника в России, а также в этом году отмечается 200-летие со дня рождения одного из основателей российской педагогики – Константина Дмитриевича Ушинского. В связи с этим я обратил внимание на статистику о российских школах. С сайтов Минпросвещения России [5] и РБК [4] были собраны данные о количестве школ в России с 1980 по 2019 год и отображены в виде таблицы (таблица 1).

*Таблица 1. Изменение кол-ва школ на территории Российской федерации*

|  |  |
| --- | --- |
| Год | Кол-во школ (тыс.) |
| 1980/1981 | 74,8 |
| 1990/1991 | 69,7 |
| 1995/1996 | 70,2 |
| 2000/2001 | 68,1 |
| 2005/2006 | 62,5 |
| 2006/2007 | 60,3 |
| 2007/2008 | 57,3 |
| 2008/2009 | 55,1 |
| 2009/2010 | 52,4 |
| 2010/2011 | 50,1 |
| 2011/2012 | 47,7 |
| 2012/2013 | 46,2 |
| 2013/2014 | 44,7 |
| 2014/2015 | 44,1 |
| 2015/2016 | 42,6 |
| 2019/2020 | 39,9 |

Полученные данные представлены в виде графика при помощи Microsoft Excel (рисунок 1):



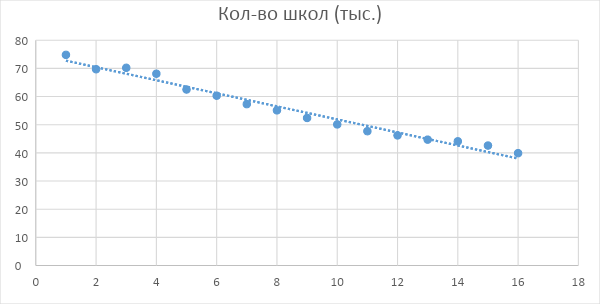
*Рисунок 1. График изменения кол-ва школ со временем в России*

График, который достаточно точно описывает изменение количества школ с 1980/1981 по 2019/2020. Однако на практике он может оказаться малопригодным. Для упрощения анализа следует найти аппроксимирующую прямую g(x) для исходной функции f(x).

Линейная аппроксимирующая функция находится по следующей формуле:

g(x) = A\*x+B

В Microsoft Excel есть функция нахождения аппроксимирующей прямой, результат ее работы вы можете увидеть на графике(рисунок 2) в виде линии тренда.



*Рисунок 2. Линейная аппроксимация g(x)*

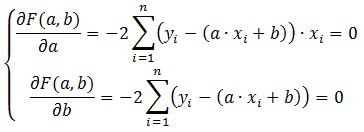
# Описание метода решения задачи

Любую функцию линейного типа можно описать при помощи уравнения: .

Суть аппроксимации заключается в нахождении в уравнении таких коэффициентов a и b, чтобы все точки, полученные опытным путем, находились как можно ближе к аппроксимирующей прямой.

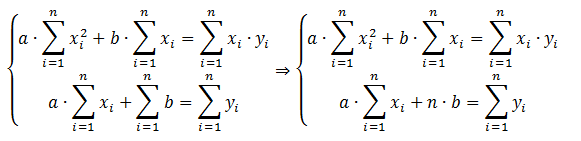
На практике для нахождения этих коэффициентов используется метод наименьших квадратов, далее МНК. Его суть в поиске суммы квадратов отклонений значений точек от точек, полученных из аппроксимирующей прямой.

Для применения МНК нужно сначала найти экстремум заданной функции двух переменных. Для этого нужна формула , позволяющая вычислить частные производные функции по коэффициентам a и b. Далее приравниваются полученные функции к нулю (формула 1).



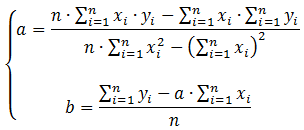
(1)

Следующий шаг: решить полученную систему уравнений:



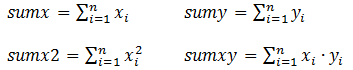
(2)

Далее определяются значения коэффициентов по формуле 3:



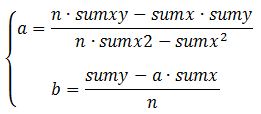
(3)

Чтобы вычислить коэффициенты нужно найти суммы всех x, y, x\*y и x^2 (формула 4):



(4)

В результате значения искомых коэффициентов будут определены как на формуле 5:



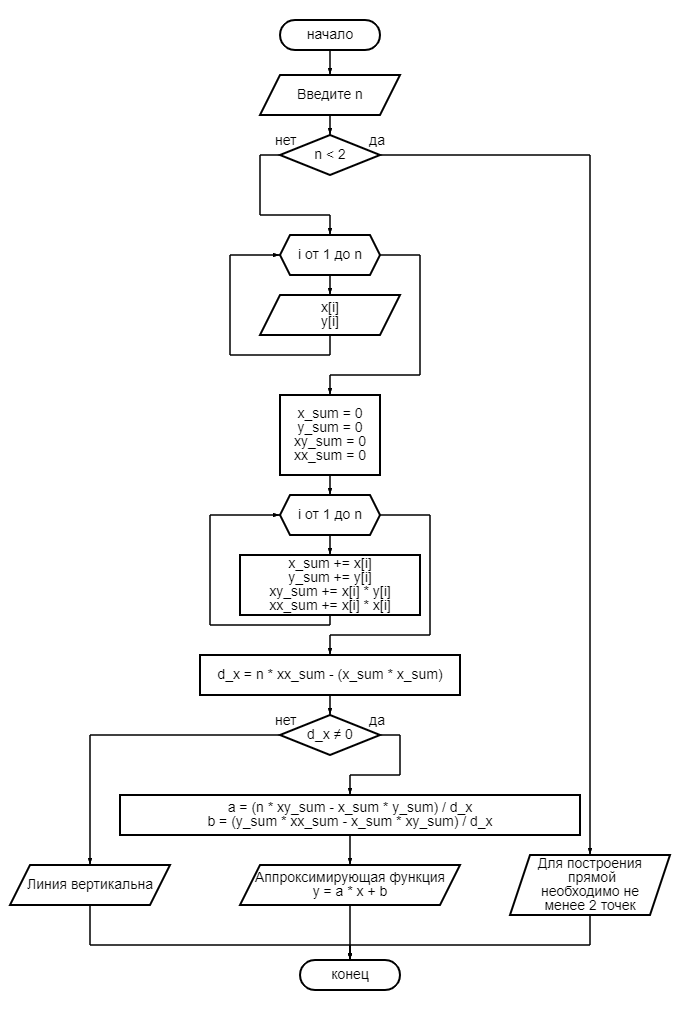
(5)

Найденные значения a и b – коэффициенты уравнения функции g(x) (), являющейся аппроксимирующей к исходной по заданным точкам.

# Разработка алгоритма решения задачи

Данный алгоритм реализуется путем нахождения оптимальных коэффициентов a и b аппроксимирующей прямой (). Для их вычисления необходимы следующие исходные данные: количество точек (n) и их координаты. Если точек меньше двух, тогда прямую невозможно найти. Если больше двух, то понадобится найти суммы: , , , и (Вывод этих функций можно проследить из пунктов 2.2 – 2.4). Если d\_x = 0, тогда прямая вертикальна. В другом случае вычисляются коэффициенты a и b по формуле 5. Полученные коэффициенты подставляются в формулу прямой .

Так находится линейная аппроксимирующая функция. Все эти шаги отображены для наглядности в блок-схеме (рисунок 3).



*Рисунок 3. Блок-схема алгоритма нахождения коэффициентов аппроксимирующей функции, применяя МНК*

# Выбор систем программирования

Система программирования - программная система, предназначенная для разработки программ на конкретном языке программирования.

Ядро системы программирования составляет язык. Были сравнены самые популярные из ныне существующих языков программирования, которые используют профессионалы.[2]

Java — основной язык, используемый для разработки Android-приложений для смартфонов и планшетов. Популярность Java у разработчиков связана с простотой и надежностью языка, который обеспечивает долгосрочную совместимость написанных на нём продуктов. Программы на Java транслируются в байт-код, выполняемый виртуальной машиной Java. Достоинством подобного способа выполнения программ является полная независимость байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина.[9]

Python — высокоуровневый язык программирования, который часто считается самым легким языком благодаря своей простоте, читаемости и синтаксису. Созданный на основе более ранних языков, он впитал в себя все их наработки и является более совершенным. Основные плюсы компилятора – минимализм, многофункциональность и простота. Но в свою очередь, за минимализм приходится платить низким быстродействием и наличием множества ошибок в системном коде.[7]

C# является основным языком для разработки на платформах и сервисах Microsoft. Язык С# позволяет разрабатывать практически любые приложения, которые связаны с Visual Studio IDE. Кроме того, это и один из основных языков для разработки игр в Unity.

Почти любой современный сайт использует элементы JavaScript — этого прототипно-ориентированного сценарного языка программирования. JavaScript — это язык программирования, выполняемый в клиентском браузере, который обрабатывает команды на компьютере конечного пользователя, а не сервера, что приводит к снижению нагрузки на сервер и увеличению скорости работы приложения.[20]

C++ — идеальный выбор для разработки мощного «настольного» программного обеспечения, игр с функцией аппаратного ускорения, а также приложений для ПК, консолей и мобильных устройств, требующих большого объема памяти для работы. Основным его плюсом является полная универсальность – на С++ можно писать всё. Также можно отметить сравнительную простоту компилятора – если вы уже овладели С, Python или Java, то работать с С++ для вас не составит труда. Минусы – это неудобный синтаксис и то, что иногда результатом работы с С++ является слишком длинный программный код, что влечёт за собой некоторые неудобства в дальнейшей работе с программой.[6]

В данной работе программа будет написана на языке программирования C# в среде объектно-ориентированного программирования Visual Studio 2019.

Microsoft Visual Studio — это программная среда по разработке приложений для ОС Windows, как консольных, так и с графическим интерфейсом. C# (произносится как "си шарп") — современный объектно-ориентированный и типобезопасный язык программирования. C# относится к широко известному семейству языков C. C# является объектно-ориентированным языком, но поддерживает также и компонентно-ориентированное программирование. Разработка современных приложений все больше тяготеет к созданию программных компонентов в форме автономных и самоописательных пакетов, реализующих отдельные функциональные возможности. Главная особенность таких компонентов в том, что они представляют собой модель программирования со свойствами, методами и событиями. У них есть атрибуты, предоставляющие декларативные сведения о компоненте. Они включают в себя собственную документацию. C# предоставляет языковые конструкции, непосредственно поддерживающие такую концепцию работы. Благодаря этому C# подходит для создания и применения программных компонентов.[2]

Вот лишь несколько функций языка C#, обеспечивающих надежность и устойчивость приложений. Сборка мусора автоматически освобождает память, занятую недостижимыми неиспользуемыми объектами. Обработка исключений предоставляет структурированный и расширяемый подход к обнаружению ошибок и их восстановлению. Типобезопасная структура языка делает невозможным чтение из неинициализированных переменных, индексацию массивов за пределами их границ или выполнение непроверенных приведений типов.

В C# существует единая система типов. Все типы C#, включая типы-примитивы, такие как int и double, наследуют от одного корневого типа object. Таким образом, все типы используют общий набор операций, и значения любого типа можно хранить, передавать и обрабатывать схожим образом. Кроме того, C# поддерживает пользовательские ссылочные типы и типы значений, позволяя как динамически выделять память для объектов, так и хранить упрощенные структуры в стеке.

Современные системы программирования обычно предоставляют пользователям мощные и удобные средства разработки программ. В них входят:

· компилятор или интерпретатор;

· интегрированная среда разработки;

· средства создания и редактирования текстов программ;

· обширные библиотеки стандартных программ и функций;

· отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе;

· "дружественная" к пользователю диалоговая среда;

· многооконный режим работы;

· мощные графические библиотеки; утилиты для работы с библиотеками

· встроенный ассемблер;

· встроенная справочная служба;

· другие специфические особенности.[7]

# Описание программного комплекса

Программа написана на языке C# (C Sharp) для платформы .NET.

При запуске программы мы видим следующий интерфейс (масштаб увеличен), представленный на рисунке 4.

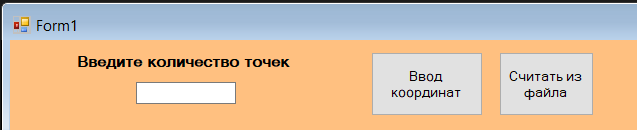


Рисунок 4. Интерфейс при запуске программы

Для начала выполнения программы необходимо ввести целое число - количество точек. За функцию ввода отвечает объект TextBox1 класса TextBox. В объекте Label1 класса Label присутствует текст с указанием того, что необходимо сделать пользователю, а именно «Введите количество точек». Далее у пользователя есть выбор: ввести данные вручную или считать их из текстового файла(объекты класса Button, «Ввод координат»(Button1) и «Считать из файла»(Button4), соответственно).

После нажатия любой из кнопок («Ввод координат» или «Считтать из файла») в коде программы изменяется свойство Size до размеров 470, 420, а также, при помощи метода show() становятся видны и доступны следующие объекты, представленные на рисунке 5: DataGridView1 класса DataGridView, Label2 класса Label и Button2 класса Button.

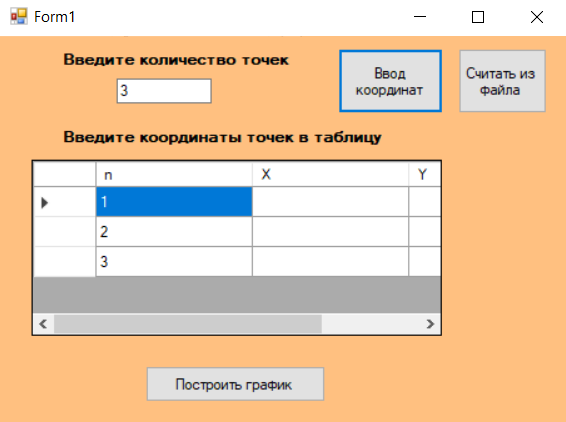


Рисунок 5. Второй этап ввода данных в программу

В зависимости от того, сколько точек мы ввели в TextBox1 на первом этапе ввода данных, в Label2 могут быть выведены следующие слова:

1. Если введено количество точек – 1, то «Нельзя найти аппроксимирующую функцию по одной точке» (рисунок 6)

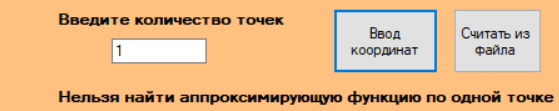


Рисунок 6. Значение Label2, если кол-во точек равно 1

1. Если введено число меньше 1, дробное число, символ, слово или пользователь не ввел число, то значение Label2 изменяется на «Введено некорректное количество точек» (рисунок 7).

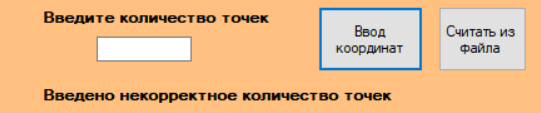


Рисунок 7. Значение Label2, если кол-во точек меньше 1 или не введено

1. Если пользователь ввел число больше 1, то «Введите координаты точек в таблицу», поясняющее, что нужно делать пользователю на данном этапе (рисунок 8).

В DataGridView1 автоматически добавляется нумерация ячеек, равная количеству введенных точек в TextBox1, и сами ячейки для ввода координат. При нажатии на кнопку «Ввод координат» на этом этапе пользователю необходимо ввести данные в эти ячейки. Все это отображено на рисунке 8.

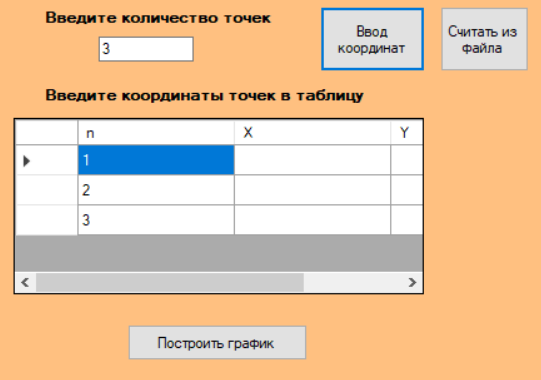


Рисунок 8. Значение Label2, если кол-во точек больше 1, DataGridView1 для ввода координат точек

При выборе кнопки Button4 «Считать из файла» у пользователя открывается окно с выбором текстового файла(рисунок 9).

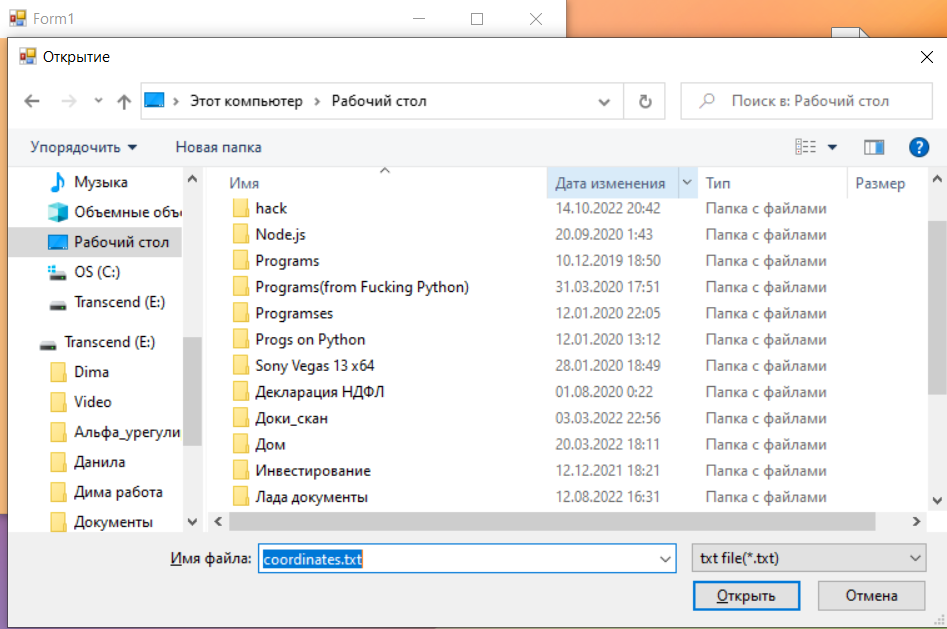


Рисунок 9. Выбор текстового файла для чтения

После выбора файла и нажатия кнопки «Открыть» данные автоматически подставляются в DataGridView1. Пример текстового файла с данными представлен на рисунке 10. Результат ввода данных отображен на рисунке 11.

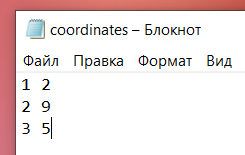


Рисунок 10. Пример текстового файла с данными

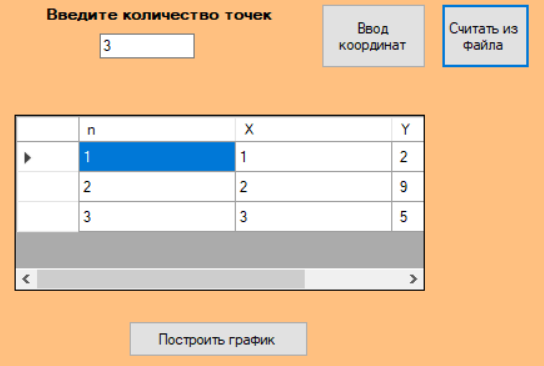


Рисунок 11. Результат ввода данных из файла

Компонент класса Button с подписью «Построить график» - это кнопка, при нажатии на которую, во-первых, изменяется размер приложения при помощи свойства Size с присвоенным значением 470, 646, во-вторых, при помощи метода Show() становятся видны следующие объекты, отображенные на рисунке 12: Chart1 класса Chart и Button3 класса Button.

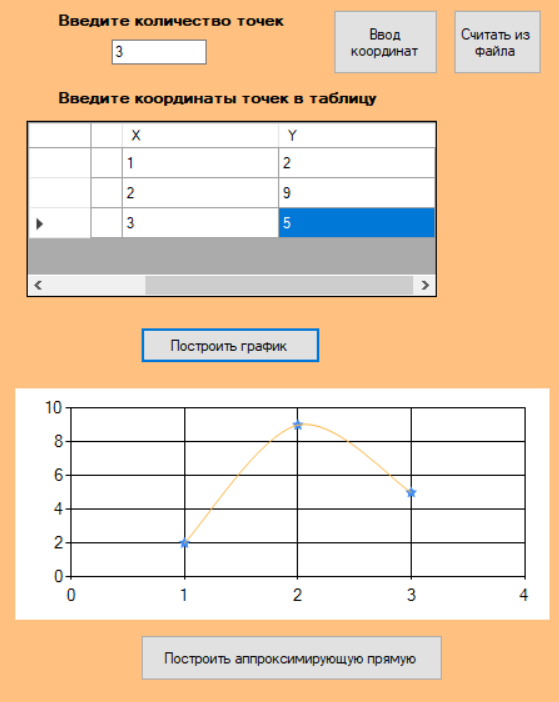


Рисунок 12. Программа отображает исходный данные в виде точек на графике, соединенных линиями

Если пользователь не ввел данные в DataGridView1, то программа обработает исключение и выдаст следующее сообщение, показанное на рисунке 13 (масштаб увеличен), в Label3: «Вы не ввели данные». Если же данные были введены, то Label3 остается скрытым от пользователя.

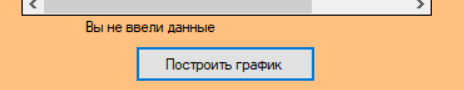


Рисунок 13. Значение Label3, если пользователем не были введены данные в DataGridView1

В Chart1 строится график по координатам, заданным пользователем в DataGridView1. Для этого программа выполняет цикл, в котором конвертирует данные из DataGridView1 сначала в тип данных String, а затем в Double, чтобы записать полученные данные в два массива класса Double и использовать полученные данные при построении точек и линий, их соединяющих на графике.

Для наглядности используются два графика: Series1 и Series2. У первого свойство ChartType с присвоенным значением «Point», которое помогает отобразить точки на графике, чтобы пользователю было видно расположение координат. У второго свойство ChartType с присвоенным значением «Line», чтобы пользователь смог увидеть изменчивость графика измерений. Примерный результат построенных графиков виден на рисунке 14 с увеличенным масштабом.

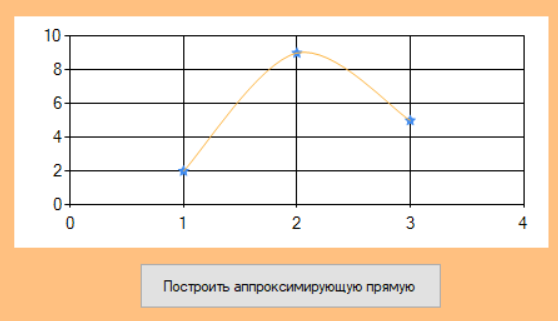


Рисунок 14. Примерный результат построенных в Chart1 графиков

При нажатии на кнопку «Получить аппроксимирующую прямую» происходит переход в новую форму(рисунок 15).

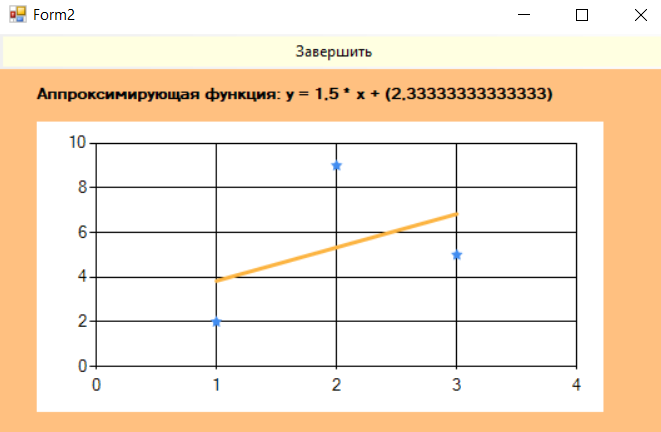


Рисунок 15. Форма 2 программы по нахождению аппроксимирующей прямой

На следующей форме выводится объект Label1 класса Label и Chart1 класса Chart, а также MenuStrip1 класса MenuStrip.

В Label1 в зависимости от того, какие данные были введены в DataGridView1 первой формы, могут быть следующие надписи:

1. «Линия вертикальна», если в результате вычислений по формулам, представленным в разделе 3 курсового проекта, программа посчитала d\_x и его значение равно 0. И, соответственно, аппроксимирующая прямая будет равна вертикальной линии. Такой вариант представлен на рисунке 16.

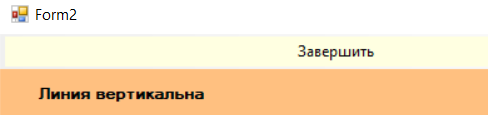


Рисунок 16. Значение "Линия вертикальна" в Label1 (2 форма)

1. «Аппроксимирующая функция: y = » коэффициент a, затем « \* x + (» и коэффициент b. Программа, в соответствии с формулами из раздела 3, нашла коэффициенты a и b аппроксимирующей прямой и подставила их в Label1. Такое значение Label можно увидеть на рисунке 15, при нашей подстановке ранее.

MenuStrip1 – это меню, в котором есть один вариант выбора – Завершить. Нажав на этот элемент меню, пользователь сможет выйти из программы. Этот элемент увеличен и отображен на рисунке 17.

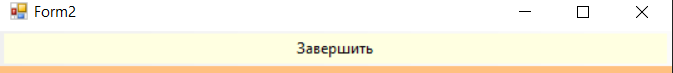


Рисунок 17. MenuStrip1 с вариантом выбора "Завершить"

В Chart1 строится график по координатам, полученным из аппроксимирующей прямой. Для этого программа берет координаты x из DataGridView1, конвертируя тип данных сначала в String, а потом в Double и записывая результат массив класса Double, и получает y из, полученного в результате выполнения программы, аппроксимирующей прямой. Полученные данные используются при построении точек и линий, соединяющих их на графике.

Для наглядности используются два графика: Series1 и Series2. У первого свойство ChartType с присвоенным значением «Point», которое помогает отобразить точки на графике, чтобы пользователю было видно расположение изначальных координат, введенных пользователем. У второго свойство ChartType с присвоенным значением «Line», чтобы пользователь смог увидеть непосредственно саму аппроксимирующую прямую. Примерный результат построенных графиков виден на рисунке 18 (Масштаб интерфейса увеличен для наглядности).

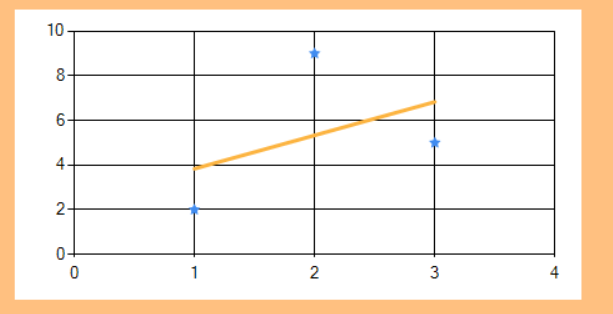


Рисунок 18. Пример графика в Chart1 (2 форма) с построенной аппроксимирующей прямой и точками – координатами заданными пользователем в DataGridView1

# Анализ результатов

Для доказательства работоспособности программы легче всего использовать Microsoft Excel. В таблицу нужно внести данные из пункта 1. Постановка задачи. Полученные данные можно посмотреть на рисунке 19.

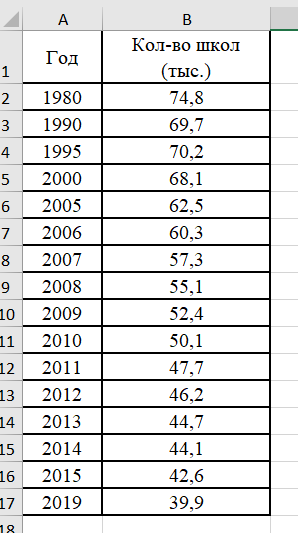


Рисунок 19. Изменение кол-ва школ на территории Российской федерации, оформленное в таблице Excel.

Чтобы найти аппроксимирующую прямую в Excel необходимо создать график по табличным данным. Для этого сначала необходимо выделить область данных, по которым планируется построить диаграмму. Выделить данные можно наведя курсор на начальную ячейку, нажать левую кнопку мыши, удерживая ее довести до конечной ячейки таблицы. Также можно быстро выделить первую ячейку, а затем нажать Shift + Стрелка Вправо, а затем Shift + Стрелка Вниз. После выделения диапазона с данными можно построить диаграмму, нажав Alt + F1. На некоторых клавиатурах функциональные клавиши могут быть объединены с другими. В этом случае потребуется использовать сочетание Alt+Fn+F1.

После построения диаграммы нужно добавить линию тренда (аппроксимирующую прямую). Сделать это можно нажав на «+» рядом с графиком и выбрать «Линия тренда», а затем «Дополнительные параметры…». Пример добавления линии тренда показан на рисунке 20.

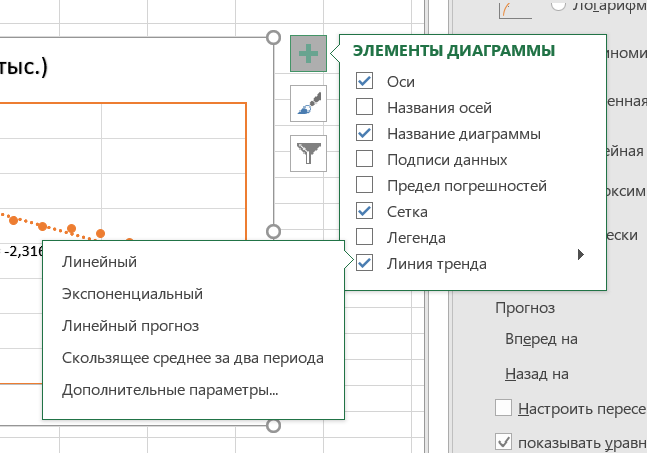


Рисунок 20. Пример добавления линии тренда на диаграмму в Excel

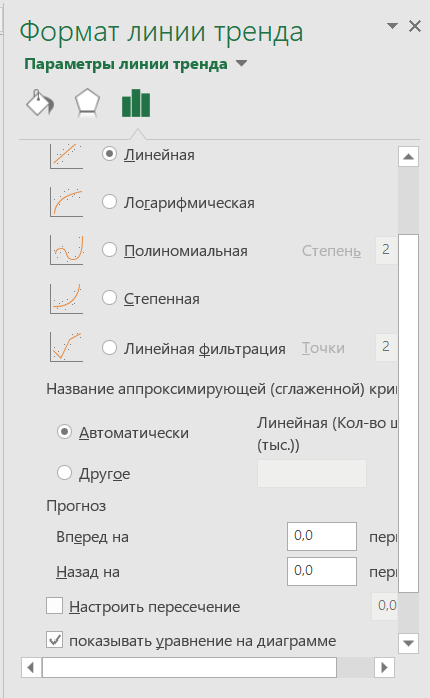
Выбрав дополнительные параметры, открывается окно «Формат линии тренда». В пункте «Параметры линии тренда» следует выбрать «Линейная», так как курсовой проект по линейной аппроксимации функции, а, следовательно, нам нужна прямая. Также, чтобы увидеть само уравнение прямой, а не только ее вид на диаграмме, необходимо выбрать пункт «показывать уравнение на диаграмме». На рисунке 21 указаны пункты, какие необходимо выбрать.

Рисунок 21. Формат линии тренда для аппроксимирующей прямой

В итоге, по данным, приведенным на рисунке 19, и выбрав все пункты, описанные выше, в Excel строится график аналогичный рисунку 22.

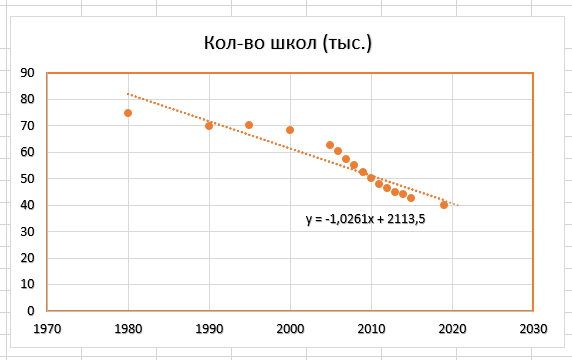
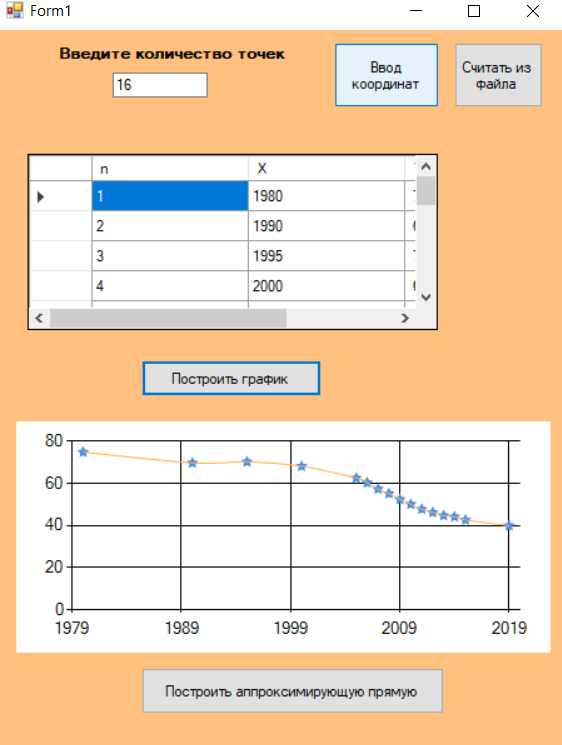


Рисунок 22. График изменения кол-ва школ в России в зависимости от года

На графике есть полученное уравнение аппроксимирующей прямой, а именно: .

Теперь нужно сравнить полученный результат с результатом работы написанной на C# программы. Для этого необходимо ввести количество точек (16) в textBox1. Затем, перейдя к следующему шагу заполнить таблицу в DataGridView1 аналогично с данными в Excel (рисунок 19).

Рисунок 23. Ввод значений из таблицы в программу



Нажав на кнопку «Получить аппроксимирующую прямую», программа открывает вторую форму, где в Label выводится значение аппроксимирующей прямой, как на рисунке 24. Так как в программе не было указано максимальное количество знаков после запятой, то результат получен более подробный, нежели в Excel, а именно: . Если сравнить его с результатом из Excel (), то можно сделать вывод, что программа работает верно. Сами по себе графики аппроксимирующей прямой приблизительно равны, как на диаграмме Excel, так и в Chart программы на C#.

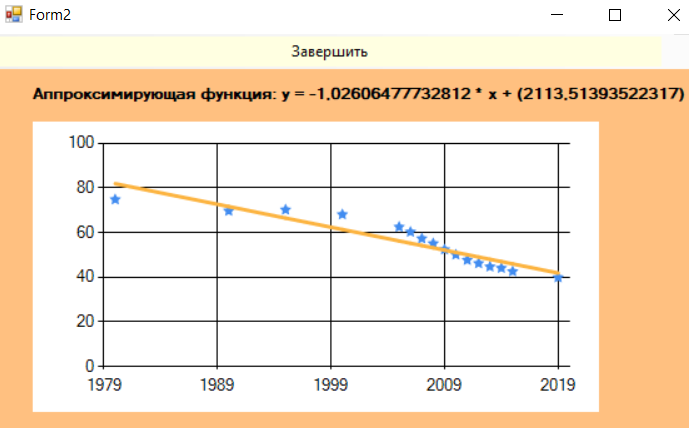


Рисунок 24. Результат работы программы на C# по введенным данным из таблицы (1)

# Заключение

В ходе выполнения данной работы было изучено больше количество математического материала. После чего был составлен алгоритм, и написана программа по нахождению аппроксимирующей функции с помощью наиболее интересного и легкого в реализации метода.

С точки зрения разработчика было потрачено около 20 часов на математический анализ поставленной задачи, 10 часов на разработку алгоритма, 10 часов на создание программного кода и рисование интерфейса и 7 часов на отладку.

Также для выбора языка программирования был проведен анализ существующих языков со всеми вытекающими плюсами и минусами. Были созданы 2 формы для работы. Кроме этого, использовались методы работы с ООП, которым нашлось применение в данной работе, так как был использован язык C#.

Корректность работы программы проверена построением графика аппроксимирующей функции с помощью Microsoft Excel. В результате проверки результат программы и построения графика в Microsoft Excel совпали.

# Список литературы

1. Малышева Т.А. Численные методы и компьютерное моделирование. Лабораторный практикум по аппроксимации функций: Учеб. -метод. пособие. СПб.: Университет ИТМО, 2016. 33 с
2. Аппроксимация функций [Электронный ресурс]: методические указания к лабораторным работам по дисциплине «Информатика» / сост. М.М. Степанов, Н.Н. Потапова, Т.В. Ерещенко; М-во образования и науки Росс. Федерации, Волгогр. гос. архит.-строит. ун-т. — Электрон. текстовые и граф, дан. (2,5 Мбайт). — Волгоград: ВолгГАСУ, 2017. — Учебное электронное издание комбинированного распространения: 1 CD-диск. — Систем. требования: PC 486 DX-33; Microsoft Windows XP; 2-скоростной дисковод CD-ROM; Adobe Reader 6.0. — Официальный сайт Волгоградского государственного архитектурностроительного университета. — Режим доступа: http://www.vgasu.ru/publishing/on-line/ — Загл. с титул. экрана.
3. Скиена С. С42 Алгоритмы. Руководство по разработке. — 2-е изд.: Пер. с англ. — СПб.: БХВ-Петербург, 2011. — 720 с.: ил. ISBN 978-5-9775-0560-4
4. Статья Евгения Калюкова «Счетная палата сообщила о резком сокращении числа школ в России» на сайте РБК URL: https://www.rbc.ru/society/28/06/2019/5d16366a9a7947d218d79f3a (дата обращения: 10.04.2022).
5. Информация об общем образовании на сайте Минпросвещения России URL: https://edu.gov.ru/activity/statistics/general\_edu (дата обращения: 10.04.2022).
6. Дроздов С.Н. Структуры и алгоритмы обработки данных: учебное пособие. Таганрог: Издательство Южного федерального университета. 2016. 228 с. URL: http://biblioclub.ru/index.php?page=book&id=493032.
7. Кнут Д. Искусство программирования. Том 1. Основные алгоритмы. М.: Вильямс. 2017. 720 с.
8. Кнут Д. Искусство программирования. Том 2. Получисленные алгоритмы. М.: Вильямс. 2017. 832 с.
9. Кнут Д. Искусство программирования. Том 3. Сортировка и поиск. М.: Вильямс. 2017. 824 с.
10. Колдаев В.Д. Численные методы и программирование. Учебное пособие. М.: Форум: Инфра-М. 2016. 336 с.
11. Пирумов У., Гидаспов В., Иванов И., Ревизников Д., Формалев В., Стрельцов В. Численные методы. Учебник и практикум. М.: Юрайт. 2017. 421 с.
12. Романенко В.В. Объектно-ориентированное программирование: учебное пособие. Томск: Томский гос. университет систем управления и радиоэлектроники. 2014. 475 с. URL: https://biblioclub.ru/index.php?page=book\_red&id=480517.
13. Скит Д. C# для профессионалов. Тонкости программирования. М.: Вильямс. 2017.608 с.
14. Гуськова, О. И. Объектно ориентированное программирование в Java : учебное пособие : [16+] / О. И. Гуськова. – Москва : Московский педагогический государственный университет (МПГУ), 2018. – 240 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=500355 (дата обращения: 01.04.2023). – Библиогр. в кн. – ISBN 978-5-4263-0648-6. – Текст : электронный.
15. Белоцерковская И.Е., Галина Н.В., Катаева Л.Ю. Алгоритмизация. Введение в язык программирования С++. М.: Национальный Открытый Университет «ИНТУИТ». 2016. 197 с.
16. Шелудько, В. М. Основы программирования на языке высокого уровня Python : учебное пособие : [16+] / В. М. Шелудько. – Ростов-на-Дону ; Таганрог : Южный федеральный университет, 2017. – 147 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=500056 (дата обращения: 01.04.2023). – Библиогр. в кн. – ISBN 978-5-9275-2649-9. – Текст : электронный.
17. Дубровин, В. В. Программирование на C# : учебное пособие : в 2 частях : [16+] / В. В. Дубровин. – Тамбов : Тамбовский государственный технический университет (ТГТУ), 2017. – Часть 1. – 81 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=499439 (дата обращения: 01.04.2023). – Библиогр.: с. 77. – ISBN 978-5-8265-1830-4. – Текст : электронный.
18. Биллиг, В. А. Основы программирования на C#3.0: ядро языка : учебное пособие : [16+] / В. А. Биллиг. – 2-е изд., испр. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 411 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=428947 (дата обращения: 01.04.2023). – ISBN 978-5-9963-0259-8. – Текст : электронный.
19. Сафонов, В. О. Возможности Visual Studio 2013 и их использование для облачных вычислений : учебное пособие : [16+] / В. О. Сафонов. – 2-е изд., испр. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 380 с. : схем., ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=429144 (дата обращения: 01.04.2023). – Библиогр. в кн. – Текст : электронный.
20. Суханов, М. В. Основы Microsoft .NET Framework и языка программирования C# : учебное пособие : [16+] / М. В. Суханов, И. В. Бачурин, И. С. Майоров ; Северный (Арктический) федеральный университет им. М. В. Ломоносова. – Архангельск : Северный (Арктический) федеральный университет (САФУ), 2014. – 97 с. : схем., табл., ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=312313 (дата обращения: 01.04.2023). – Библиогр. в кн. – ISBN 978-5-261-00934-4. – Текст : электронный.
21. Абрамян, М. Э. Введение в стандартную библиотеку шаблонов C++. Описание, примеры использования, учебные задачи: учебник по курсу «Стандартная библиотека C++» для студентов направления 02.03.02 «Фундаментальная информатика и информационные технологии» (бакалавриат) : [16+] / М. Э. Абрамян ; Южный федеральный университет. – Ростов-на-Дону ; Таганрог : Южный федеральный университет, 2017. – 179 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=499454 (дата обращения: 01.04.2023). – Библиогр. в кн. – ISBN 978-5-9275-2374-0. – Текст : электронный.
22. Николаев, Е. И. Объектно-ориентированное программирование : учебное пособие : [16+] / Е. И. Николаев ; Северо-Кавказский федеральный университет. – Ставрополь : Северо-Кавказский Федеральный университет (СКФУ), 2015. – 225 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=458133 (дата обращения: 01.04.2023). – Библиогр. в кн. – Текст : электронный.
23. Объектно-ориентированное программирование: лабораторный практикум : практикум : в 2 частях : [16+] / авт.-сост. Е. И. Николаев ; Северо-Кавказский федеральный университет. – Ставрополь : Северо-Кавказский Федеральный университет (СКФУ), 2015. – Часть 1. – 183 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=458134 (дата обращения: 01.04.2023). – Библиогр.: с. 179. – Текст : электронный.
24. Объектно-ориентированное программирование: лабораторный практикум : практикум : в 2 частях : [16+] / авт.-сост. Е. И. Николаев ; Северо-Кавказский федеральный университет. – Ставрополь : Северо-Кавказский Федеральный университет (СКФУ), 2015. – Часть 2. – 156 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=458135 (дата обращения: 01.04.2023). – Библиогр. в кн. – Текст : электронный.
25. Публичная платформа, создающая окончательную коллекцию вопросов и ответов по кодированию на сайте StackOverflow URL: https://stackoverflow.com/ (дата обращения: 01.04.2023).

# Приложение

**Код программы «Линейная аппроксимация функции»**

Форма 1. Начальный интерфейс программы и ввод данных.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Net.Mime.MediaTypeNames;

namespace Kyrsovaya

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

label2.Text = "";

label3.Text = "";

dataGridView1.Hide();

button2.Hide();

button3.Hide();

chart1.Hide();

Size = new Size(470, 130);

}

Form2 f2 = new Form2();

private void button1\_Click(object sender, EventArgs e)

{

Size = new Size(470, 420);

dataGridView1.Show();

button2.Show();

dataGridView1.Rows.Clear();

dataGridView1.Refresh();

int i = 1;

int n = 0;

try

{

n = Convert.ToInt32(textBox1.Text);

}

catch

{

label2.Text = "Введено некорректное количество точек";

}

if (n >= 2)

{

label2.Text = "Введите координаты точек в таблицу";

for (i = 1; i <= n; i++)

{

dataGridView1.Rows.Add(String.Format("{0:0}", i));

}

}

if (n == 1) { label2.Text = "Нельзя найти аппроксимирующую функцию по одной точке"; }

}

private void button2\_Click(object sender, EventArgs e)

{

label3.Text = "";

double n = 0, l, m;

try

{

n = Convert.ToInt32(textBox1.Text);

}

catch { }

for (int x = 1; x < 3; x++)

{

for (int i = 0; i < n; i++)

{

if (dataGridView1.Rows[i].Cells[x].Value == null)

{

label3.Text = "Вы не ввели данные";

}

}

}

if (label3.Text != "Вы не ввели данные")

{

Size = new Size(470, 646);

button3.Show();

chart1.Show();

chart1.Series[0].Points.Clear();

chart1.Series[1].Points.Clear();

for (int y = 0; y < n; y++)

{

l = Convert.ToDouble(dataGridView1.Rows[y].Cells[1].Value.ToString());

m = Convert.ToDouble(dataGridView1.Rows[y].Cells[2].Value.ToString());

chart1.Series[0].Points.AddXY(l, m);

chart1.Series[1].Points.AddXY(l, m);

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

f2.Show();

Hide();

double x\_sum = 0, y\_sum = 0, xy\_sum = 0, xx\_sum = 0, d\_x, a = 0, b = 0, x, y;

int n = Convert.ToInt32(textBox1.Text);

for (int i = 0; i < n; i++)

{

x\_sum += Convert.ToDouble(dataGridView1.Rows[i].Cells[1].Value.ToString());

y\_sum += Convert.ToDouble(dataGridView1.Rows[i].Cells[2].Value.ToString());

xy\_sum += Convert.ToDouble(dataGridView1.Rows[i].Cells[1].Value.ToString()) \* Convert.ToDouble(dataGridView1.Rows[i].Cells[2].Value.ToString());

xx\_sum += Convert.ToDouble(dataGridView1.Rows[i].Cells[1].Value.ToString()) \* Convert.ToDouble(dataGridView1.Rows[i].Cells[1].Value.ToString());

}

d\_x = n \* xx\_sum - Math.Pow(x\_sum, 2);

if (d\_x != 0)

{

a = (n \* xy\_sum - x\_sum \* y\_sum) / d\_x;

b = (y\_sum \* xx\_sum - x\_sum \* xy\_sum) / d\_x;

f2.label1.Text = "Аппроксимирующая функция: y = " + a + " \* x + (" + b + ")";

}

else f2.label1.Text = "Линия вертикальна";

f2.chart1.Series[1].Points.Clear();

for (int i = 0; i < n; i++)

{

x = Convert.ToDouble(dataGridView1.Rows[i].Cells[1].Value.ToString());

y = a \* x + b;

f2.chart1.Series[1].Points.AddXY(x, y);

}

double d, k;

for (int m = 0; m < n; m++)

{

d = Convert.ToDouble(dataGridView1.Rows[m].Cells[1].Value.ToString());

k = Convert.ToDouble(dataGridView1.Rows[m].Cells[2].Value.ToString());

f2.chart1.Series[0].Points.AddXY(d, k);

}

}

private void button4\_Click(object sender, EventArgs e)

{

Size = new Size(470, 420);

dataGridView1.Show();

button2.Show();

dataGridView1.Rows.Clear();

dataGridView1.Refresh();

StreamReader sr;

Stream stream;

String filename = "";

String str;

int nc = 0;

OpenFileDialog openFileDialog1 = new OpenFileDialog();

openFileDialog1.Filter = " txt file(\*.txt)| \*.txt | All files(\*.\*) | \*.\* ";

dataGridView1.Rows.Clear();

chart1.Series[0].Points.Clear();

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

if ((stream = openFileDialog1.OpenFile()) != null)

{

filename = openFileDialog1.FileName;

stream.Close();

}

else

{

MessageBox.Show("No FileName", "dataGridChart", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

}

else

{

MessageBox.Show("No SaveFuleDialog", "dataGridChart", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

sr = new StreamReader(filename);

int i = 0;

while (!sr.EndOfStream)

{

str = sr.ReadLine();

nc = str.IndexOf(' ');

dataGridView1.Rows.Add();

dataGridView1.Rows[i].SetValues((i+1), str.Substring(0, nc), str.Substring(nc));

i++;

}

}

}

}

Форма 2. Форма с выводом полученной аппроксимирующей прямой.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Kyrsovaya

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

}

private void назадToolStripMenuItem\_Click(object sender, EventArgs e)

{

Application.OpenForms[0].Close();

}

}

}