Содержание

[Введение 4](#_Toc27865128)

[1. Описание метода решения задачи 5](#_Toc27865129)

[1.1. Интерполяция 5](#_Toc27865130)

[1.2. Способы интерполяции 5](#_Toc27865131)

[1.3. Кривая Безье. 5](#_Toc27865132)

[Разработка структур данных 7](#_Toc27865133)

[2. Разработка алгоритмов 9](#_Toc27865134)

[3. Тестирование программы 11](#_Toc27865135)

[4. Разработка документации 15](#_Toc27865136)

[5.1. Назначение и условия применения программы 15](#_Toc27865137)

[5.2 Характеристика программы 15](#_Toc27865138)

[5.3 Обращение к программе 15](#_Toc27865139)

[5.4 Входные и выходные данные 17](#_Toc27865140)

[Заключение 19](#_Toc27865141)

[Библиографический список 20](#_Toc27865142)

[Приложение А. Листинг программы 21](#_Toc27865143)

Введение

Целью курсового проекта является разработка программной системы для решения математической задачи методом сплайн-интерполяции (Кривая Безье).

Задачами курсового проектирования является:

* изучение метода решения задачи;
* разработка алгоритма и структур данных;
* разработка программы, позволяющей решить задачу уточнения корней уравнения;
* выполнение контрольного расчета в ручном режиме;
* тестирование работы программы на контрольном примере;
* тестирование работы программы на нескольких тестовых примерах;
* разработка документации к программе.

Пояснительная записка к курсовому проекту состоит из следующих разделов:

* введение;
* описание метода решения задачи;
* разработка структур данных;
* разработка алгоритмов;
* тестирование программы;
* разработка документации;
* заключение;
* библиографический список;
* приложение (листинг программных модулей).

# Описание метода решения задачи

## Интерполяция

Интерполяция - способ нахождения промежуточных значений величины по имеющемуся дискретному набору известных значений.

Рассмотрим систему несовпадающих точек , = из некоторой области D. Пусть значения функции известны только в этих точках:

, где

Задача интерполяции состоит в том, чтобы найти такую функцию F из данного класса функций, что:

* Точки называются узлами интерполяции, а их совокупность – интерполяционной сеткой;
* Пары называют точками данных или базовыми точками;
* Разность между «соседними» значениями – шаг интерполяционной сетки. Он может быть как переменным, так и постоянным;
* Функция – интерполирующая функция или интерполянт;

## Способы интерполяции

1. Интерполяция методом ближайшего соседа;
2. Интерполяция многочленами;
3. Обратное интерполирование (вычисление x при заданной y);
4. Интерполяция функции нескольких переменных;
5. Рациональная интерполяция;
6. Тригонометрическая интерполяция.

## Кривая Безье.

Кривая Безье является разновидностью сплайн-интерполяции, которая в свою очередь относится к интерполяции многочленами.

Кривые бывают линейные, квадратичные и высших степеней. Разберём в качестве примера кубическую кривую.

В параметрической форме кубическая Кривая Безье описывается следующим уравнением:

P0, P1, P2, P3 – опорные точки, заданные в 2-х или 3-х мерном пространстве – определяют форму кривой.

Линия берёт начало в точке Р0, направляется к Р1, и заканчивается в Р3, подходя к ней со стороны Р2. То есть кривая не проходит через сами точки Р1 и Р2, они используются для того, чтобы указать её направление. Длина отрезка между Р0 и Р1 определяет то, как скоро кривая повернёт к Р3.

# Разработка структур данных

Для решения задачи необходимо определить типы данных, которые будут использоваться для хранения исходных и промежуточных данных, а также результатов вычислений.

Язык *Free Pascal* предоставляет возможность использования следующих целочисленных типов (Таблица 1):

Таблица 1 – Целые типы данных

|  |  |  |
| --- | --- | --- |
| **Тип** | **Диапазон значений** | **Размер, байт** |
| Byte | 0 .. 255 | 1 |
| Shortint | -128 .. 127 | 1 |
| Smallint | -32768 .. 32767 | 2 |
| Word | 0 .. 65535 | 2 |
| Integer | smallint или longint | 2 или 4 |
| Cardinal | longword | 4 |
| Longint | -2147483648 .. 2147483647 | 4 |
| Longword | 0..4294967295 | 4 |
| Int64 | -9223372036854775808 .. 9223372036854775807 | 8 |
| QWord | 0 .. 18446744073709551615 | 8 |

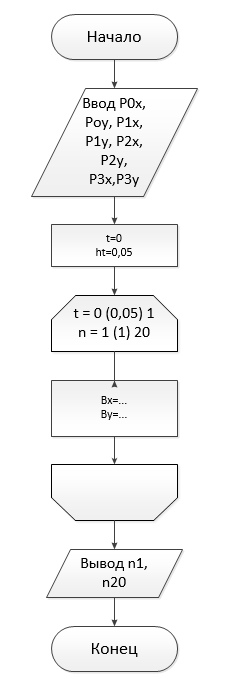
Язык *Free Pascal* предоставляет возможность использования вещественных типов данных:

*Таблица 2 – Вещественные типы данных*

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип** | **Диапазон значений** | **Значимых разрядов** | **Размер, байт** |
| Real | Зависит от платформы | Зависит от платформы | 4 или 8 |
| Single | 1.5E-45 .. 3.4E38 | 7-8 | 4 |
| Double | 5.0E-324 .. 1.7E308 | 15-16 | 8 |
| Extended | 1.9E-4932 .. 1.1E4932 | 19-20 | 10 |
| Comp | -2E64+1 .. 2E63-1 | 19-20 | 8 |
| Currency | -922337203685477.5808 | 922337203685477.5807 | 8 |

Тип данных *Real* зависит от платформы, и использовать его нежелательно. Тип *Currency* предназначен для операций с числами с фиксированной точностью (операции с денежными величинами). Тип *Comp* трактуется как вещественное число без дробной части.

Точность данных типа *Single* недостаточна для решения задач вычислительной математики, поэтому предпочтение следует отдать типам *Double* или *Extended*.

  
*Рисунок 1 – Алгоритм нахождения коэффициентов Кривой Безье*

# Разработка алгоритмов

Для построения Кривой Безье вводными данными алгоритма будут:

* Bx, By – координаты значений функции;
* P0x, P0y, P1x, P1y, P2x, P2y P3x, P3y – координаты опорных точек;
* n – счётчик точек;
* t1, t2, t3 – введённые замены переменных

Тело цикла будет включать в себя следующие действия:

* вычисление координат точек функции;

Условием завершение цикла является:

* *t* принимает значение 1;

На рисунке (Рисунок 3.2) представлен алгоритм уточнения корня методом половинного деления.

Общий алгоритм (Рисунок 3.3) будет заключаться в следующем:

* ввод исходных данных (P0x, P0y, P1x, P1y, P2x, P2y P3x, P3y);
* выполнение координат;
* вывод результатов.

# Тестирование программы

Тестирование программного обеспечения — процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом

В [6] дается следующее определение тестирования: тестирование – это процесс выполнения программы с целью нахождения ошибок.

Для тестирования программы построения кривой Безье необходимо:

* Ввести простые координаты опорных точек;
* Построить полученную кривую любым удобным способом;
* Меняя опорные точки проверять корректность построения разных Кривых.

Проведем тестирование программы для точек P0 = (0 , 0); P1 = (-2 , 6); P2 = (6 , 6) P3 = (0 , 6)

Зададим указанные выше координаты и выполним тестовый запуск программы:

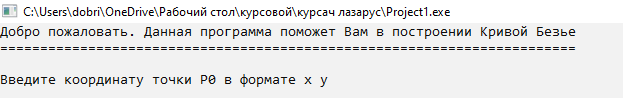


Рисунок 4.1 – Запуск программы

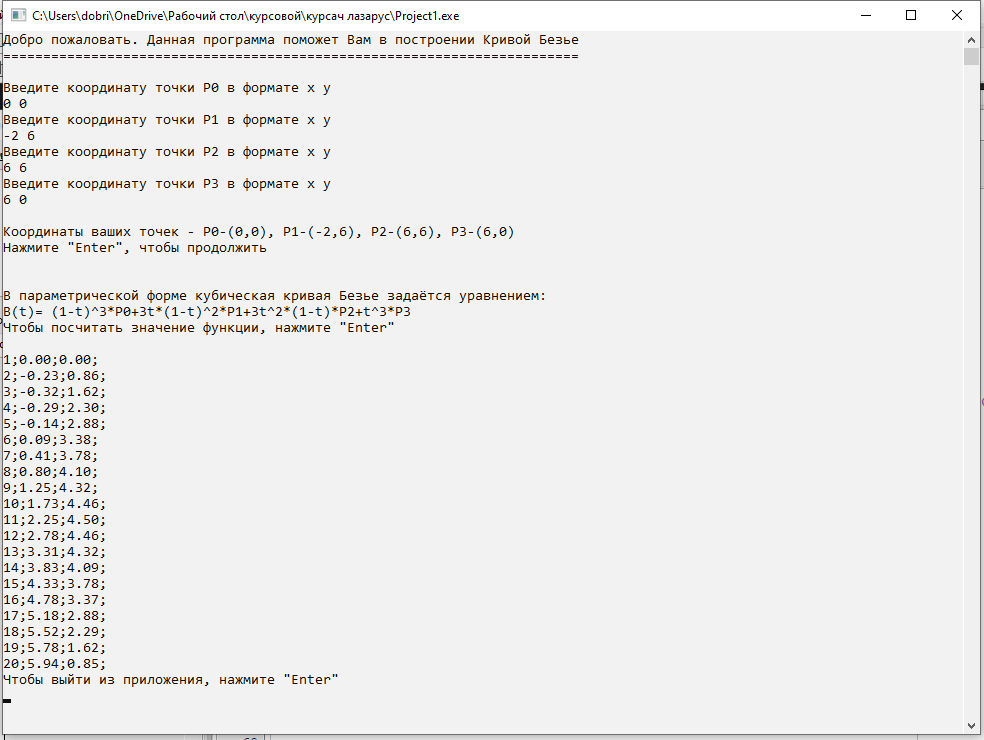
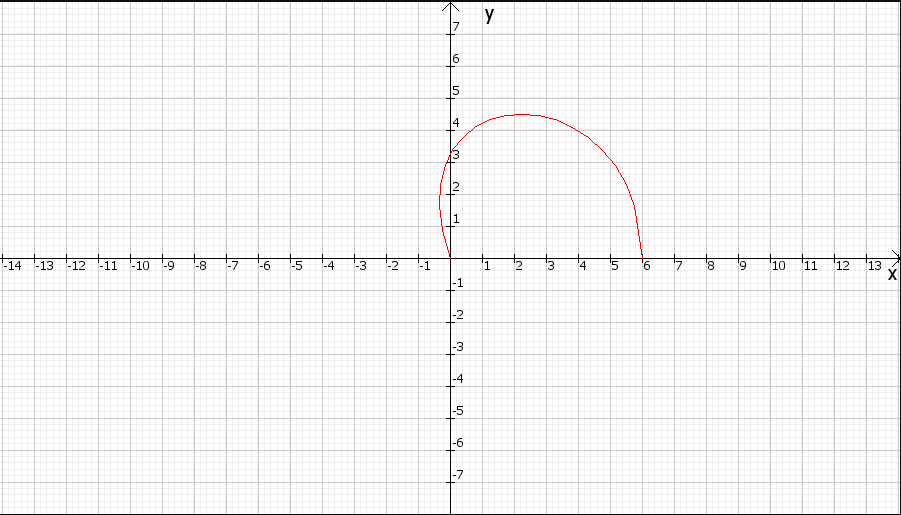


Рисунок 4.2 – Результаты выполнения программы

В результате запуска были вычислены требуемые координаты, по которым и строится Кривая Безье.

Теперь по имеющимся точкам построим кривую



*Рисунок 4.3 – Кривая, полученная в результате работы программы*

# Разработка документации

В данном разделе приведено руководство программиста по использованию программы применения Кривой Безье

## 5.1. Назначение и условия применения программы

Программа *HalfDivision* предназначена для нахождения координат построения Кривой Безье. Программа позволяет задавать опорные точки, по которым в последствии ищутся координаты, из которых и строится Кривая.

Для выполнения программы необходимо наличие операционной системы *Windows*. Особые требования к составу периферийных устройств не предъявляются.

Персонал, использующий программу (программист) должен обладать практическими навыками работы с текстовым пользовательским интерфейсом.

## 5.2 Характеристика программы

Программа *HalfDivision* является независимым программным обеспечением и обеспечивает решение задач вычислительной математики – уточнение корня уравнения методом половинного деления.

Работа программы осуществляется в консольном режиме.

Размер исполняемого файла *HalfDivision*.*exe* – 92 889байт.

## 5.3 Обращение к программе

Запуск программы осуществляется стандартным для операционной системы способом – двойным щелчком левой кнопки мыши на исполняемом файле *HalfDivision*.*exe* или выполнением команды в *HalfDivision* в командной строке в каталоге программы.

После запуска в окне консоли (Рисунок 5.1) появляется следующая информация:

* назначение программы;
* окно ожидания ввода данных.

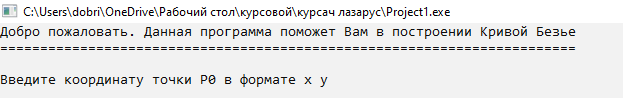


Рисунок 5.1 – Результат запуска программы HalfDivision

Пользователь должен ввести по порядку все опорные точки, после чего программа выведет их на экран для уточнения значений.

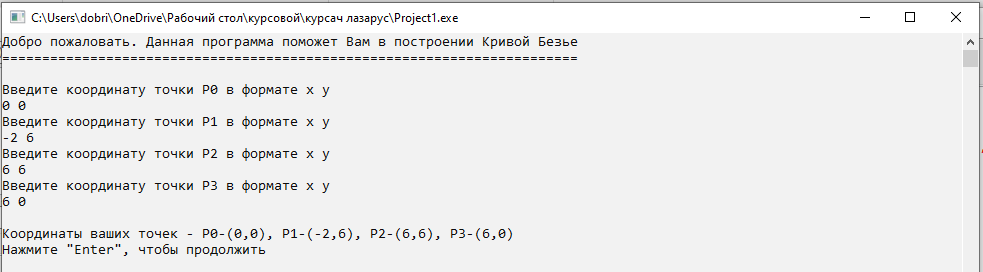
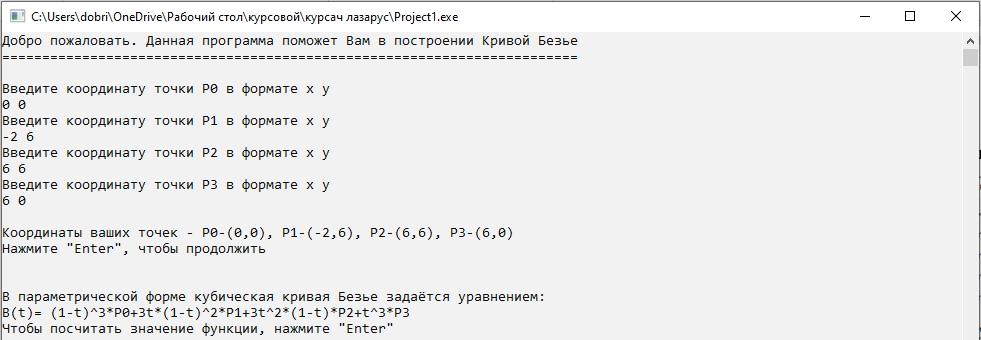


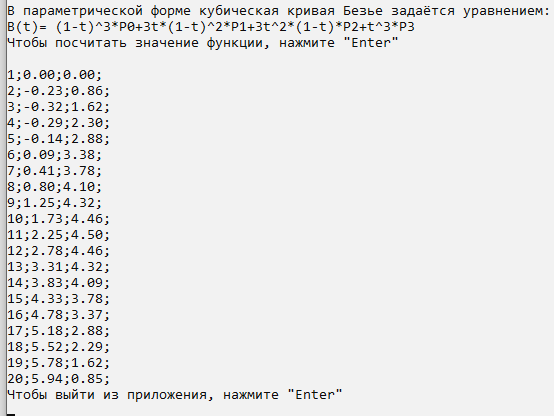
Рисунок 5.2 – Результат запуска программы HalfDivision

После ввода данных программа показывает пользователю формулу, по которой будут высчитываться координаты точек, по которым в последствии построится кривая.



*Рисунок 5.3 – Результат запуска программы HalfDivision*

Далее осуществляется расчет и вывод итоговых координат



*Рисунок 5.4 – окончательный результат работы программы*

## 5.4 Входные и выходные данные

Входными данными программы являются:

* опорные точки функции

Выходными данными программы являются:

* координаты точек, по которым строится кривая Безье;

Заключение

В данном курсовом проекте решена задача построения Кривой Безье. В процессе создания курсового проекта разработан алгоритм решения поставленной задачи. По этому алгоритму составлена и отлажена программа, в которой вычисляются координаты точек, составляющих Кривую Безье 3 порядка.

Библиографический список

1. ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов.
2. ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов.
3. ГОСТ 19.104-78\* ЕСПД. Основные надписи.
4. ГОСТ 19.105-78\* ЕСПД. Общие требования к программным документам.
5. ГОСТ 19.106-78\* ЕСПД. Общие требования к программным документам, выполненным печатным способом.
6. ГОСТ 19.504-79\* ЕСПД. Руководство программиста. Требования к содержанию и оформлению.
7. ГОСТ 19.604-78\* ЕСПД. Правила внесения изменения в программные документы, выполненные печатным способом.
8. Самарский А.А., Гулин А.В. Численные методы. - М.: 1989.
9. Калиткин Н.Н. Численные методы. - М.: Наука, 1978.
10. Демидович Б.П., Марон И.А. Основы вычислительной математики. - М.: 1970.
11. Копченова Н.В., Марон И.А. Вычислительная математика в примерах и задачах . - М.: Наука, 1972.
12. Мудров А.Е. Численные методы для ПЭВМ на языках Бейсик, Фортран, Паскаль. – Томск: МП «Раско», 1991.
13. Майерс Г.[en] Надежность программного обеспечения. М: Мир, 1980

Приложение А. Листинг программы

{$codepage utf8}

{\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*}

{ }

{ Проект HalfDivision }

{ Copyright (c) 2019 ФГБОУ РГРТУ им. В.Ф. Уткина }

{ ФВТ/кафедра КТ }

{ }

{ Разработчик: ст. гр. 848 Сапунов Д.Р. }

{ Выполнен: 22.12.2019 }

{ }

{\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*}

program Project1;

var

Bx,By, //координаты точек кривой

t, ht, //шаг

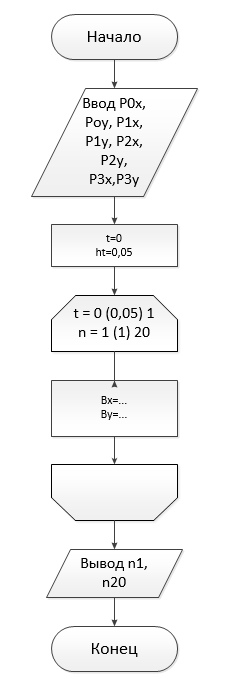
P0x,P0y,P1x,P1y,P2x,P2y,P3x,P3y : extended; //координаты опорных точек

n: integer; //номер точки

t1, t2, t3: extended; //замены выраженый

begin

writeln('Добро пожаловать. Данная программа поможет Вам в построении Кривой Безье');



writeln('========================================================================');

writeln();

writeln('Введите координату точки P0 в формате x y');

readln(P0x,P0y);

writeln('Введите координату точки P1 в формате x y');

readln(P1x,P1y);

writeln('Введите координату точки P2 в формате x y');

readln(P2x,P2y);

writeln('Введите координату точки P3 в формате x y');

readln(P3x,P3y);

writeln();

writeln('Координаты ваших точек - P0-(',P0x:0:0,',',P0y:0:0,'), P1-(',P1x:0:0,',',P1y:0:0,'), P2-(',P2x:0:0,',',P2y:0:0,'), P3-(',P3x:0:0,',',P3y:0:0,')');

writeln('Нажмите "Enter", чтобы продолжить');

writeln();

readln();

writeln('В параметрической форме кубическая кривая Безье задаётся уравнением:');

writeln('B(t)= (1-t)^3\*P0+3t\*(1-t)^2\*P1+3t^2\*(1-t)\*P2+t^3\*P3');

writeln('Чтобы посчитать значение функции, нажмите "Enter"');

readln();

t := 0; //задаём начальный параметр t

ht := 0.05; //задаём шаг параметра t

n := 1;

while t <= 1 do

begin

t1 := 1 - t; //делаем замены для упрощения выражения

t2 := t1 \* t1;

t3 := t2 \* t1;

Bx:= t3\*P0x+3\*t\*t2\*P1x+3\*t\*t\*t1\*P2x+t\*t\*t\*P3x; //считаем координату точки по иксу

By:= t3\*P0y+3\*t\*t2\*P1y+3\*t\*t\*t1\*P2y+t\*t\*t\*P3y; //считаем координату точки по игреку

writeln(n, ';', Bx:0:2, ';', By:0:2, ';');

t:= t + ht;

n := n + 1;

end;

writeln('Чтобы выйти из приложения, нажмите "Enter"');

readln();

end.