МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

|  |  |
| --- | --- |
| Институт (факультет) | Институт информационных технологий |
| Кафедра | МПО ЭВМ |

КУРСОВАЯ РАБОТА

|  |  |
| --- | --- |
| по дисциплине | Структурное программирование |

|  |  |
| --- | --- |
| на тему | Программирование на языке высокого уровня |

|  |  |
| --- | --- |
|  | Выполнил студент группы 1ПИб-02-2оп-22 |
|  | *группа* |
|  | Направления подготовки (специальности) |
|  | 09.03.04 Программная инженерия |
|  | *шифр, наименование* |
|  | Симаньков Александр Евгеньевич |
|  | *фамилия, имя, отчество* |
|  |  |
|  | Руководитель |
|  | Пышницкий Константин Михайлович |
|  | *фамилия, имя, отчество* |
|  | Старший преподаватель |
|  | *должность* |
|  | Дата представления работы |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023 г. |
|  |  |
|  | Заключение о допуске к защите |
|  |  |
|  |  |
|  | Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | *количество баллов* |
|  | Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

2023 год

Оглавление

[Введение 3](#_Toc138153714)

[1.Описание предметной области 4](#_Toc138153715)

[2.Описание классов Graphics, Pen и Brush 5](#_Toc138153716)

[2.1. Класс Graphics 5](#_Toc138153717)

[2.2. Класс Pen 6](#_Toc138153718)

[2.3. Класс Brush 6](#_Toc138153719)

[3.Описание созданного приложения 7](#_Toc138153720)

[3.1. Постановка задачи 7](#_Toc138153721)

[3.2. Логическое проектирование 8](#_Toc138153722)

[3.3. Физическое проектирование 9](#_Toc138153723)

[3.4. Тестирование 11](#_Toc138153724)

[3.5. Результаты работы 12](#_Toc138153725)

[Заключение 12](#_Toc138153726)

[Источники 13](#_Toc138153727)

[Приложение 1. Техническое задание 14](#_Toc138153728)

[Приложение 2. Руководство пользователя 20](#_Toc138153729)

[Приложение 3. Программный код 25](#_Toc138153730)

Введение

Программирование играет большую роль в наше время, так как пришла эпоха информационного общества, в котором каждый день создаются все новые программы, приложения и так далее. Языков программирования существует довольно много, поэтому, выбор языка программирования является непростой задачей в наше время. Различают языки высокого и низкого уровня. Высокоуровневые языки программирования отличаются от низкоуровневых тем, что он может использовать более “естественный” язык, что делает его более понятным и удобным. Также высокоуровневый язык программирования может автоматизировать значительные области вычисления систем, таких как, управление памятью, что делает его более приятным для использования, в отличии от языков низкого уровня. Также, портативность языка играет важную роль в удобстве его использования, поэтому высокоуровневые языки получили большую поддержку от программистов, нежели языки низкого уровня [1].

Язык С++ является одним из самых популярных высокоуровневых языков программирования. Он является развитием языка С и добавляет возможности объектно-ориентированного программирования, что делает его мощным инструментом для создания сложных программных систем [2].

Задача курсовой работы заключается в том, чтобы разработать программу, которая способна генерировать графическое изображение, а именно – “Спираль Архимеда”. Также программа должна иметь возможность изменять размер и цвет изображения, а также сохранять изображение через соответствующую кнопку.

1.Описание предметной области

Предметная область данной курсовой работы связана с разработкой программы для создания Архимедовой спирали в Microsoft Forms. Спираль играют важную роль в геометрии, физике, инженерии, дизайне и искусстве.

1. Геометрия: Спираль Архимеда является примером плоской кривой, у  
которой расстояние между оборотами равномерно увеличивается.

2. Физика: Спираль Архимеда встречается в некоторых физических процессах. Например, электроны в магнитных полях описывают спиральные траектории, похожие на спираль Архимеда.

3. Инженерия и дизайн: Спираль Архимеда часто используется в инженерии и дизайне. Например, ее форма может быть применена в конструкции винтовых лестниц, витражей, пружин, катушек и других механизмов.

4. Искусство: Спираль Архимеда обладает гармоничной и эстетически приятной формой, поэтому она широко используется в художественных произведениях, архитектуре, дизайне и украшениях[5].

Windows Forms — это платформа пользовательского интерфейса для создания классических приложений Windows. Она обеспечивает один из самых эффективных способов создания классических приложений с помощью визуального конструктора в Visual Studio. Такие функции, как размещение визуальных элементов управления путем перетаскивания, упрощают создание классических приложений [3].

Предметной областью является спираль Архимеда (рис.1).

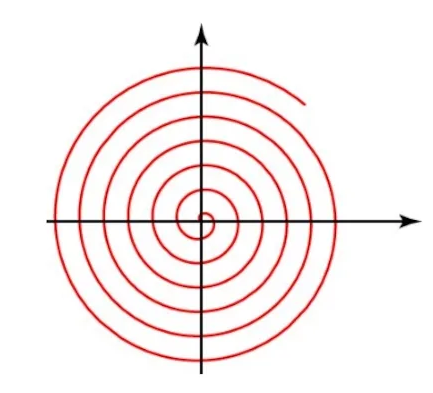


Рис.1. Спираль Архимеда

2.Описание классов Graphics, Pen и Brush

Рисование фигур является одним из важных аспектов в разработке графических приложений. В Windows Forms это может быть реализовано с помощью классов Graphics, Pen и Brush в языке программирования C++. Пространство имен System.Drawing (Рисование) обеспечивает доступ к функциональным возможностям графического интерфейса GDI+ , используя около 50 классов, в том числе класс Graphics, Pen и Brush.

2.1. Класс Graphics

Класс Graphics предоставляет методы рисования на устройстве отображения. Данный класс входит в пространство имен System.Drawing, как и большинство классов для работы с графикой. Класс Graphics определяет набор методов для вывода текста, изображений и геометрических фигур [4].

Методов в этом классе довольно много, поэтому рассмотрим некоторые из них в табл. 1:

Таблица 1

Методы класса Graphics

|  |  |
| --- | --- |
| Имя | Описание |
| AddMetafileComment | Добавляет комментарий к текущему объекту Metafile. |
| BeginContainer() | Сохраняет графический контейнер, содержащий текущее состояние данного объекта Graphics, а затем открывает и использует новый графический контейнер. |
| Clear | Очищает всю поверхность рисования и выполняет заливку поверхности указанным цветом фона. |
| CopyFromScreen(Int32, Int32, Int32, Int32, Size) | Выполняет передачу данных о цвете, соответствующих прямоугольной области пикселей, блоками битов с экрана на поверхность рисования объекта Graphics. |
| Dispose | Освобождает все ресурсы, используемые данным объектом Graphics. |
| DrawArc(Pen, Int32, Int32, Int32, Int32, Int32, Int32) | Рисует дугу, которая является частью эллипса, заданного парой координат, шириной и высотой. |
| DrawBezier(Pen, Point, Point, Point, Point) | Рисует кривую Безье, определяемую четырьмя структурами Point. |
| DrawLine(Pen, Int32, Int32, Int32, Int32) | Проводит линию, соединяющую две точки, задаваемые парами координат. |
| FillClosedCurve(Brush, Point[], FillMode) | Заполняет внутреннюю часть замкнутой фундаментальной сплайновой кривой, определяемой массивом структур Point, используя указанный режим заливки. |
| FillEllipse(Brush, Int32, Int32, Int32, Int32) | Заполняет внутреннюю часть эллипса, определяемого ограничивающим прямоугольником, заданным с помощью пары координат, ширины и высоты. |

2.2. Класс Pen

Класс Pen определяет объект, используемый для рисования прямых линий и кривых [4].

Методы представлены в табл. 2:

Таблица 2

Методы класса Pen

|  |  |
| --- | --- |
| Имя | Описание |
| Pen(Color) | Инициализирует новый экземпляр класса Pen с указанным цветом. |
| Pen(Color, Single) | Инициализирует новый экземпляр класса Pen с указанными свойствами Color и Width. (Width - устанавливает ширину пера Pen, в единицах объекта Graphics, используемого для рисования) |

2.3. Класс Brush

Класс Brush определяет объекты, которые используются для заливки внутри графических фигур, таких как прямоугольники, эллипсы, круги, многоугольники и дорожки.

Это абстрактный базовый класс, который не может быть реализован. Для создания объекта "кисть" используются классы, производные от Brush, такие как SolidBrush, TextureBrush и LinearGradientBrush [4].

3.Описание созданного приложения

В данном разделе описаны основные этапы разработки приложения для построения Архимедовой спирали.

3.1. Постановка задачи

Для корректной работы программы требуется выполнить следующие задачи:

* Программа должна создавать графическое изображение спирали при нажатии соответствующей кнопки;
* программа должна позволять выбирать цвет заднего фона;
* программа должна позволять менять масштаб, толщину линии и цвет спирали;
* программа должна позволять сохранять в файловом виде изображение при нажатии соответствующей кнопки;
* программа должна иметь возможность повторно создать графическое изображение при изменении масштаба, цвета и толщины линии.

3.2. Логическое проектирование

Для построения спирали (рис. 1) нам понадобятся 2 функции, которые мы обозначим функциями «А» и «Б». Функция «А» рисует саму спираль по заданным переменным, а функция «Б» рисует сетку, по которой строится сама спираль. Обе эти функции будут вызываться общей функцией, которая предназначена для рисования спирали.  
 Функция «А» будет рисовать саму спираль, а также координатную ось, в точке пересечения линий которых будет начало спирали. Для построения самой спирали нам понадобятся переменные, с помощью которых будет строиться спираль, а также цикл for, с помощью которого будет происходить само построение. Переменная «r» будет отвечать за расстояние между витками, а переменные x, y, x1 и y1 за местоположение спирали на рисунке. Сама спираль строиться по формулам: x1 = 600 - sin(alpha / 180.0 \* 3.14) \* r; y1 = 300 + cos(alpha / 180.0 \* 3.14) \* r;. Где r – расстояние между витками, x1, x2 – положение точек на рисунке, а alpha – угол поворота спирали. Также переменные size и tol нужны для изменения масштаба и толщины линии спирали соответственно.

Функция «Б» отвечает за построение сетки, по которой легче понять, как строить саму спираль. Чтобы ее изобразить нам понадобятся несколько переменных. Переменная r отвечает за расстояние между линиями, x1, y1 отвечает за положение точек на рисунке, alpha также отвечает за угол наклона. Сама сетка строится по формулам: x1 = 600 - sin(alpha / 180.0 \* 3.14) \* r; y1 = 300 + cos(alpha / 180.0 \* 3.14) \* r;. Переменная size нужна для изменения масштаба.

Алгоритм работы программы представлен на блок-схеме (рис. 2):

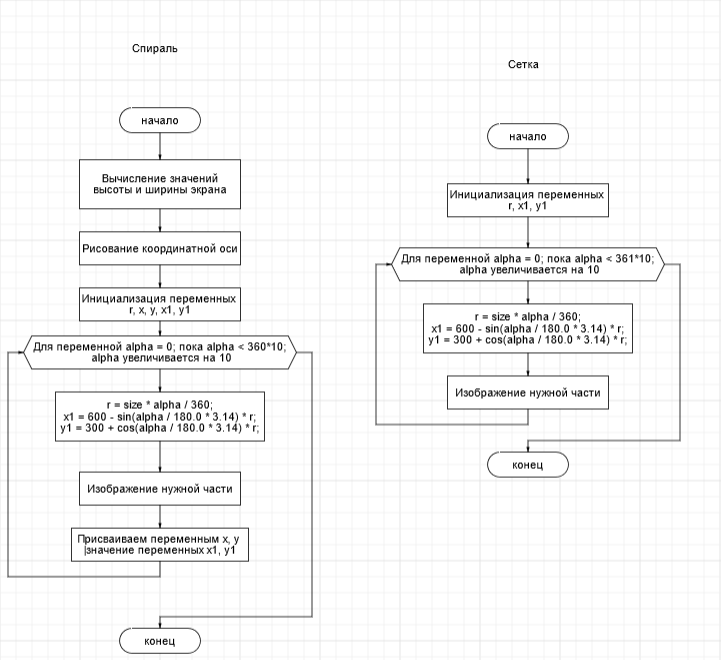


Рис. 2. Алгоритм работы программы в блок-схеме

3.3. Физическое проектирование

Переменные, примененные для реализации программы, представлены в табл. 3:

Таблица 3

Переменные

|  |  |  |
| --- | --- | --- |
| Наименование | Обозначение | Тип данных |
| Значение размера спирали | size | int |
| Ширина прямоугольника для рисования | width | int |
| Высота прямоугольника для рисования | height | int |
| Координата х | x | float |
| Координата у | y | float |
| Координата X1 | x1 | float |
| Координата Y1 | y1 | float |
| Значение угла | alpha | int |
| Значение радиуса | r | float |
| Значение толщины линии спирали | tol | int |
| Цвет линии спирали и заднего фона | color | Color |
| Количество вариантов изменения ползунка | value | int |

Спецификация функций представлена в табл. 4:

Таблица 4

Спецификация функций

|  |  |  |  |
| --- | --- | --- | --- |
| Имя модуля | Заголовок процедуры или функции | Формальные параметры | Выполняемое действие |
| Kursach.sln | button1\_Click | System::Object^ sender, System::EventArgs^ e | Функция выполняет вызов функции построения спирали и сетки. |
| Kursach.sln | SaveButton\_Click | System::Object^ sender, System::EventArgs^ e | Функция сохраняет полученное изображение. |
| Kursach.sln | trackBar1\_Scroll | System::Object^ sender, System::EventArgs^ e | Функция задаёт масштаб спирали. |
| Kursach.sln | trackBar2\_Scroll | System::Object^ sender, System::EventArgs^ e | Функция задаёт толщину линии спирали. |
| Kursach.sln | colorButton\_Click | System::Object^ sender, System::EventArgs^ e | Функция вызывает диалоговое окно для выбора пользователю цвета спирали. |
| Kursach.sln | button2\_Click | System::Object^ sender, System::EventArgs^ e | Функция вызывает диалоговое окно для выбора пользователю цвета заднего фона. |
| Kursach.sln | SpiralDraw | int size, Color color | Функция рисует спираль и координатные оси. |
| Kursach.sln | SpiralDraw | int size, Color color | Функция рисует сетку |

3.4. Тестирование

Тестовые данные и результаты тестирования представлены в табл. 5, 6:

Таблица 5

Тестовые данные

|  |  |  |
| --- | --- | --- |
| Исходные данные | Тестируемая функция | Ожидаемый результат |
| Нажатие кнопки «Архимедова спираль» | button1\_click | Построение графика |
| Нажатие кнопки «Сохранить» | SaveButton\_Click | Сохранение изображения |
| Изменение толщины линии | trackBar2\_Scroll | Построение графика с другой толщиной линии (от 1 до 5) |
| Изменение масштаба | trackBar1\_Scroll | Построение графика с другим масштабом (от 1 до 5) |
| Изменение цвета фона | button2\_Click | Изменение цвета фона на выбранный |
| Изменение цвета спирали | colorButton\_Click | Изменение цвета спирали на выбранный |

Таблица 6

Результаты тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Дата тестирования | Тестируемый модуль | Кто проводил тестирование | Описание теста | Результат тестирования |
| 16.06.2023 | Kursach.sln | Овчинников М.В. | Проверка работы кнопки построения спирали | Успех |
| 16.06.2023 | Kursach.sln | Овчинников М.В. | Проверка работы кнопки сохранения изображения | Успех |
| 16.06.2023 | Kursach.sln | Милавин Д.А. | Изменение толщины линии | Успех |
| 16.06.2023 | Kursach.sln | Милавин Д.А | Изменение масштаба | Успех |
| 16.06.2023 | Kursach.sln | Гончаров Е.Д. | Изменение цвета фона | Успех |
| 16.06.2023 | Kursach.sln | Гончаров Е.Г. | Изменение цвета спирали | Успех |

3.5. Результаты работы

В результате работы было создано приложение для построения Архимедовой спирали в Windows Forms на языке С++. Спираль строится по изначальным значениям, присутствует возможность изменения масштаба, толщины линии, задания цвета спирали и цвета заднего фона, а также возможность сохранения изображения.

Интерфейс понятен, удобен и представлен на рис. 3.



Рис. 3. Интерфейс программы

Заключение

В рамках данного проекта была разработана программа в Windows Forms на языке С++, реализующая графическое изображение “Архимедова спираль”.

В ходе работы, были получены навыки в работе с Windows Forms, освоены возможности классов Graphics, Pen и Brush.

Источники

* 1. Разница между языком высокого и низкого уровня [электронный ресурс]: https://ru.strephonsays.com/high-level-language-and-vs-low-level-language-14521 Дата доступа: 02.06.2023
  2. Язык программирования С++ [электронный ресурс]: https://metanit.com/cpp/tutorial/1.1.php Дата доступа: 02.06.2023
  3. Windows Forms [электронный ресурс]: https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-6.0 Дата доступа: 15.06.2023
  4. Основы библиотеки System.Drawing [электронный ресурс]: https://c-sharp.pro/общие-замечания-классы-graphics-pen-и-brush/. Дата доступа: 15.06.2023
  5. Назначение Архимедовой спирали [электронный ресурс]: https://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D0%BC%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0\_%D1%81%D0%BF%D0%B8%D1%80%D0%B0%D0%BB%D1%8C. Дата доступа: 16.06.2023

Приложение 1. Техническое задание

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт информационных технологий |
| наименование института (факультета) |
|  |
| Кафедра математического и программного обеспечения ЭВМ |
| наименование кафедры |
| Структурное программирование |
| наименование дисциплины в соответствии с учебным планом |

УТВЕРЖДАЮ

Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_

д.т.н., профессор\_\_\_Ершов Е.В.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_2023г.

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ ВЫСОКОГО УРОВНЯ

Техническое задание на курсовую работу

Листов 6

Руководитель: старший преподаватель

Пышницкий К.М.

Исполнитель: студент гр. 1ПИб-02-2оп-22

Симаньков А.Е.

2023 год

Введение

Программа для построения Архимедовой спирали в Windows Forms рисует изображение и сохраняет полученный результат.

Данный проект предназначен для формирования знаний и навыков по дисциплине «Структурное программирование».

1. Основания для разработки

Основанием для разработки является задание на курсовую работу по дисциплине «Структурное программирование», выданное на кафедре МПО ЭВМ ИИТ ЧГУ.

Наименование темы разработки: программирование на языке высокого уровня.

2. Назначение разработки

Назначение разработки этой программы состоит в создании приложения для построения Архимедовой спирали. Программа предоставляет пользователю возможность изменения масштаба спирали, толщины ее линий, заднего фона и цвета самой спирали. Также программа позволяет сохранить полученный результат в файл для дальнейшего использования в форматах JPEG,PNG,BMP. Программа может использоваться для создания Архимедовой спирали и дальнейшей эксплуатации в своих целях.

3. Требования к программе

3.1. Требования к функциональным характеристикам

Требования к функциональным характеристикам программы:

* Генерация спирали: Программа должна позволять пользователю генерировать Архимедову спираль. Сгенерированная спираль должна быть наглядно отображена на экране, чтобы пользователь мог визуально оценить результат.
* Изменение параметров спирали: Пользователь должен иметь возможность изменять параметры спирали, такие как толщина линии, цвет и масштаб, и видеть соответствующие изменения визуализации.
* Сохранение спирали: Пользователь должен иметь возможность сохранить сгенерированную спираль в файле, чтобы использовать его в дальнейшем для своих целей.
* Понятный интерфейс: Интерфейс программы должен быть интуитивно понятным и удобным в использовании, чтобы пользователь мог легко настраивать параметры спирали и видеть результаты своих действий.

3.2. Требования к надежности

Программа должна работать корректно и не вызывать непредвиденных ошибок. Она должна обрабатывать любые возможные ситуации и ошибки. Также программа должна корректно изображать спираль.

3.3. Условия эксплуатации

Программа должна быть совместима с операционной системой пользователя. Условия эксплуатации программы зависят от условий эксплуатации персонального компьютера.

3.4. Требования к составу и параметрам технических средств

Минимальные требования к техническим характеристикам компьютера для работы программы:

* процессор с тактовой частотой 1,6 ГГц;
* оперативная память объемом не менее 2 ГБ;
* свободное место на жестком диске от 2 ГБ;
* монитор;
* мышь, клавиатура;
* клавиатура.

3.5. Требования к информационной и программной совместимости

Для обеспечения программной совместимости необходим Visual Studio 2022 или новее на ОС Windows 7 или выше.

3.6. Требования к маркировке и упаковке

Требования к маркировке и упаковке программы, как правило, не являются применимыми, так как программа представляет собой цифровой продукт, который распространяется в электронном формате.

3.7. Требования к транспортированию и хранению

Файлы, требуемые для корректной работы программы, необходимо расположить на флеш-носителе либо памяти компьютера.

3.8. Специальные требования

Программное обеспечение должно иметь дружественный интерфейс, рассчитанный на пользователя средней квалификации. Ограничение возрастного порога с 12 лет.

4. Требования к программной документации

4.1. Содержание расчетно-пояснительной записки

Программная документация должна содержать расчётно-пояснительную записку с содержанием:

Титульный лист

Оглавление

Введение

Описание предметной области

Описание классов Graphics, Pen и Brush

Описание созданного приложения

1. Постановка задачи

2. Логическое проектирование

3. Физическое проектирование

4. Тестирование

5. Результаты работы

Заключение

Источники

Приложения

4.2. Требования к оформлению

Программная документация должна удовлетворять следующему оформлению (табл. П1.1):

Таблица П1.1

Требования к оформлению

|  |  |
| --- | --- |
| Документ | Печать на отдельных листах формата А4 (210х297 мм); оборотная сторона не заполняется; листы нумеруются. Печать возможна ч/б.  Файлы предъявляются на компакт-диске: РПЗ с ТЗ; программный код.  Листы и диск в конверте вложены в пластиковую папку скоросшивателя. |
| Страницы | Ориентация – книжная; отдельные страницы, при необходимости, альбомная.  Поля: верхнее, нижнее – по 2 см, левое – 3 см , правое – 1 см. |
| Абзацы | Межстрочный интервал – 1, перед и после абзаца – 0. |
| Шрифты | Кегль – 14. В таблицах шрифт 12. Шрифт листинга – 10 (возможно в 2 колонки). |
| Рисунки | Подписывается под ним по центру: Рис.Х. Название  В приложениях: Рис.П1.3. Название |
| Таблицы | Подписывается: над таблицей, выравнивание по правому: «Таблица Х».  В следующей строке по центру Название  Надписи в «шапке» (имена столбцов, полей) – по центру.  В теле таблицы (записи) текстовые значения – выровнены по левому краю, числа, даты – по правому. |

5. Технико-экономические показатели

Технико-экономические показатели к данной программе не предъявляются.

6. Стадии и этапы разработки

Стадии и этапы разработки представлены в табл. П1.2:

Таблица П1.2

Стадии и этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа разработки | Сроки разработки | Результат выполнения | Отметка о выполнении |
| Разработка ТЗ | до 23.05.2023 | Разработанное ТЗ |  |
| Разработка программы | 27.05.2023-15.06.2023 | Разработанная программа |  |
| Разработка руководства пользователя | 01.06.2023-20.06.2023 | Разработанное рук. пользователя |  |
| Разработка РПЗ | 01.06.2023-20.06.2023 | Разработанная РПЗ |  |

7. Порядок контроля и приемки

Порядок контроля и приемки представлен в табл. П1.3:

Таблица П1.3

Порядок контроля и приемки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование  контрольного этапа выполнения курсовой работы | Сроки  контроля | Результат выполнения | Отметка о приемке  результата  контрольного этапа |
| Оформление ТЗ | 23.05.2023 | Оформленное ТЗ |  |
| Разработка программы | 10.06.2023 | Неконечная версия программы |  |
| Оформление руководство пользователя | 20.06.2023 | Оформленное руководство пользователя |  |
| Доработка программы | 15.06.2023 | Конечная версия программы |  |
| Оформление РПЗ | 20.06.2023 | Оформленная РПЗ |  |
| Сдача РПЗ | 21.06.2023 | Оценка за курсовую работу |  |

Приложение 2. Руководство пользователя

1. Общие сведения о программе

Программа предназначена для графического изображения спирали Архимеда (рис. П2.1).

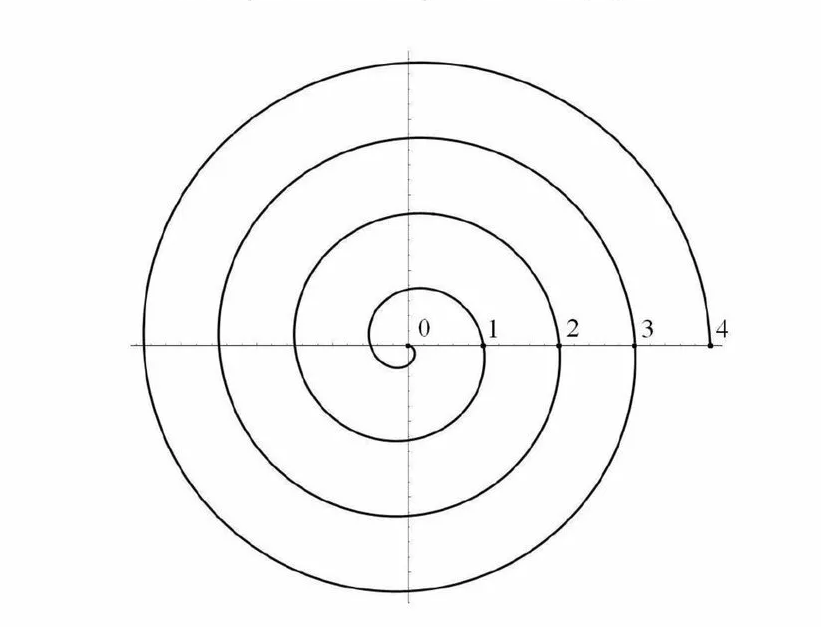


Рис. П2.1. Спираль Архимеда

Программа создает спираль по параметрам: масштаб, толщина линии и цвет изображения.

2. Описание установки

Установка программы не требуется. Чтобы запустить программу, нужно иметь доступ к файлу программы и Visual Studio 2022 или выше.

3. Описание запуска

1. Открыть папку, в которой находится файл программы.

2. Открыть программу.

3. Появляется окно программы, в котором можно работать (рис. П2.2).



Рис. П2.2. Окно программы

4. Инструкция по работе

Параметры построения спирали задаются с помощью элементов интерфейса управления (рис. П2.3).



Рис. П2.3. Элементы управления

1. Построение начальной спирали, цвет по стандарту синий, размер зависит от значения поля размера.
2. Выбор толщины линии спирали. Для выбора толщины линии нужно менять значение ползунка (от 1 до 5).
3. Выбор масштаба. Для выбора масштаба нужно менять значение ползунка (от 1 до 5).
4. Выбор заднего фона. При нажатии на соответствующую кнопку появляется диалоговое окно, в котором необходимо выбрать цвет, в который будет окрашен задний фон.
5. Выбор цвета спирали. При нажатии на соответствующую кнопку появляется диалоговое окно, в котором необходимо выбрать цвет, в который будет окрашена спираль.
6. Сохранение изображения. Для сохранения изображения нужно нажать на соответствующую кнопку. Появится диалоговое окно, где пользователь может выбрать путь и формат сохранения файла.

После изменения масштаба, толщины линии и цвета программа построит спираль (рис. П2.4) и сохранит его изображение (рис. П2.5).

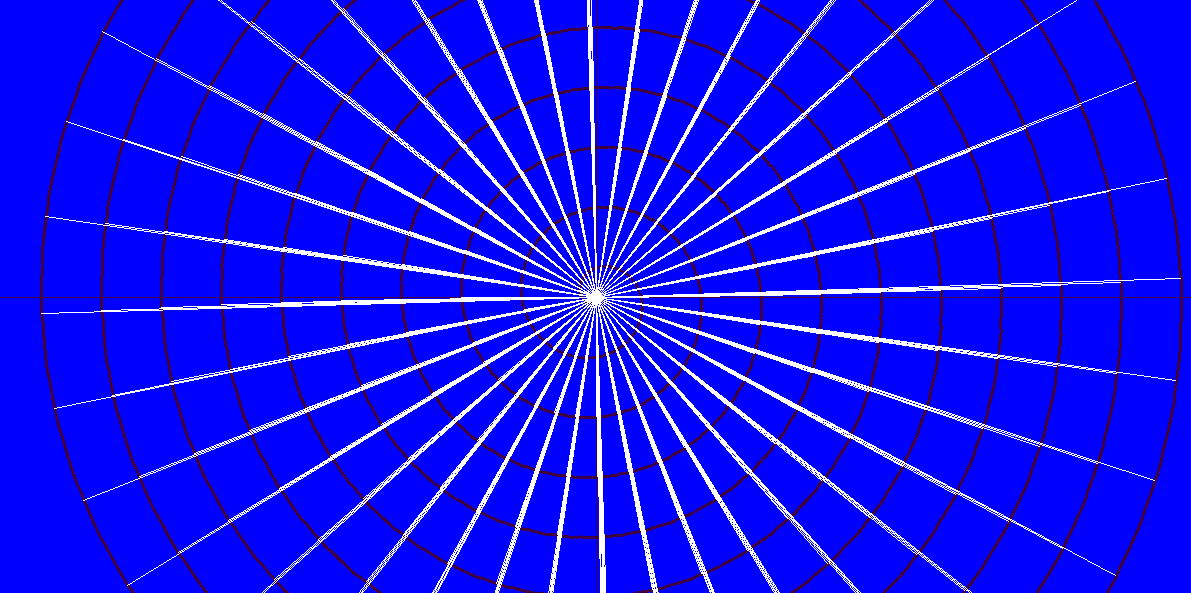


Рис. П2.4. Построенная спираль

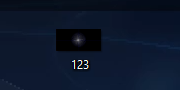


Рис. П2.5. Сохраненное изображение

5. Сообщения пользователю

При нажатии на кнопку изменить цвет спирали появится диалоговое окно, представленное на рис. П2.6:

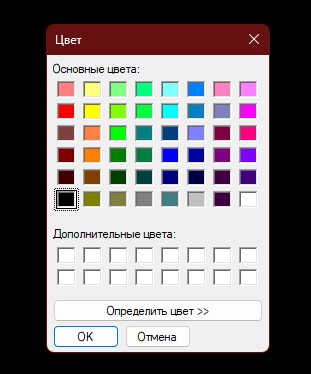


Рис. П2.6. Диалоговое окно выбора цвета

При нажатии на кнопку сохранить появится диалоговое окно, представленное на рис. П2.7:

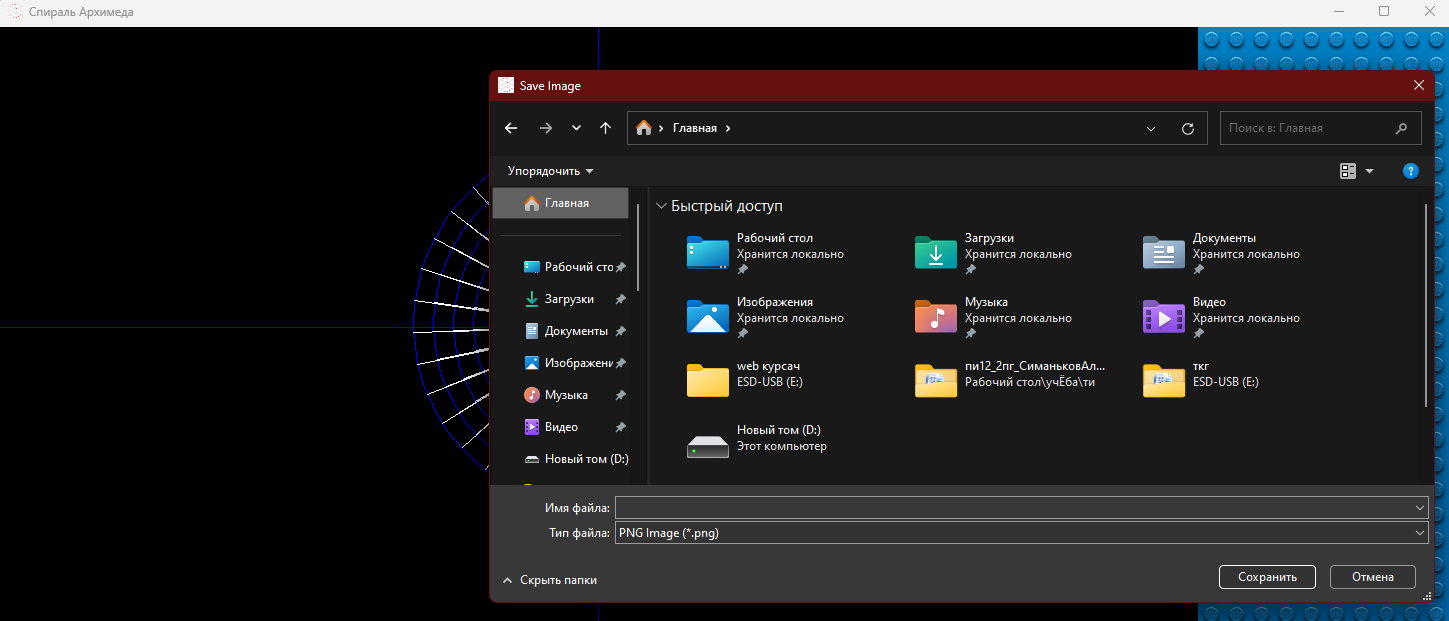


Рис. П2.7. Диалоговое окно сохранения изображения

Приложение 3. Программный код

Код MyForm.cpp:

#include "MyForm.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThread]

int main(array<String^>^ args) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Kursach::MyForm form;

Application::Run(% form);

return 0;

}

Код MyForm.h:

#pragma once

#include <ctime>

#include <cstdlib>

namespace Kursach {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::PictureBox^ pictureBox1;

protected:

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Button^ SaveButton;

private: System::Windows::Forms::SaveFileDialog^ saveFileDialog1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Button^ colorButton;

private: System::Windows::Forms::ColorDialog^ colorDialog1;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::TrackBar^ trackBar1;

private: System::Windows::Forms::TrackBar^ trackBar2;

private: System::Windows::Forms::Label^ label2;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MyForm::typeid));

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

this->button2 = (gcnew System::Windows::Forms::Button());

this->colorButton = (gcnew System::Windows::Forms::Button());

this->label1 = (gcnew System::Windows::Forms::Label());

this->SaveButton = (gcnew System::Windows::Forms::Button());

this->button1 = (gcnew System::Windows::Forms::Button());

this->saveFileDialog1 = (gcnew System::Windows::Forms::SaveFileDialog());

this->colorDialog1 = (gcnew System::Windows::Forms::ColorDialog());

this->trackBar1 = (gcnew System::Windows::Forms::TrackBar());

this->trackBar2 = (gcnew System::Windows::Forms::TrackBar());

this->label2 = (gcnew System::Windows::Forms::Label());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar2))->BeginInit();

this->SuspendLayout();

//

// pictureBox1

//

this->pictureBox1->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->pictureBox1->BackColor = System::Drawing::Color::Black;

this->pictureBox1->Location = System::Drawing::Point(-1, 0);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(1200, 600);

this->pictureBox1->TabIndex = 0;

this->pictureBox1->TabStop = false;

this->pictureBox1->Click += gcnew System::EventHandler(this, &MyForm::pictureBox1\_Click);

//

// button2

//

this->button2->BackColor = System::Drawing::SystemColors::Desktop;

this->button2->ForeColor = System::Drawing::Color::Blue;

this->button2->Location = System::Drawing::Point(1266, 316);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(140, 35);

this->button2->TabIndex = 6;

this->button2->Text = L"Изменить задний фон";

this->button2->UseVisualStyleBackColor = false;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// colorButton

//

this->colorButton->BackColor = System::Drawing::SystemColors::Desktop;

this->colorButton->ForeColor = System::Drawing::Color::Blue;

this->colorButton->Location = System::Drawing::Point(1266, 394);

this->colorButton->Name = L"colorButton";

this->colorButton->Size = System::Drawing::Size(140, 35);

this->colorButton->TabIndex = 5;

this->colorButton->Text = L"Изменить цвет спирали";

this->colorButton->UseVisualStyleBackColor = false;

this->colorButton->Click += gcnew System::EventHandler(this, &MyForm::colorButton\_Click);

//

// label1

//

this->label1->AutoSize = true;

this->label1->BackColor = System::Drawing::SystemColors::HotTrack;

this->label1->ForeColor = System::Drawing::Color::Blue;

this->label1->Location = System::Drawing::Point(1280, 232);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(56, 13);

this->label1->TabIndex = 4;

this->label1->Text = L"Масштаб:";

this->label1->Click += gcnew System::EventHandler(this, &MyForm::label1\_Click);

//

// SaveButton

//

this->SaveButton->BackColor = System::Drawing::SystemColors::ActiveCaptionText;

this->SaveButton->ForeColor = System::Drawing::Color::Blue;

this->SaveButton->Location = System::Drawing::Point(1266, 477);

this->SaveButton->Name = L"SaveButton";

this->SaveButton->Size = System::Drawing::Size(140, 31);

this->SaveButton->TabIndex = 1;

this->SaveButton->Text = L"Сохранить";

this->SaveButton->UseVisualStyleBackColor = false;

this->SaveButton->Click += gcnew System::EventHandler(this, &MyForm::SaveButton\_Click);

//

// button1

//

this->button1->BackColor = System::Drawing::SystemColors::ActiveCaptionText;

this->button1->ForeColor = System::Drawing::SystemColors::MenuHighlight;

this->button1->Location = System::Drawing::Point(1266, 52);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(140, 33);

this->button1->TabIndex = 0;

this->button1->Text = L"Спираль Архимеда";

this->button1->UseVisualStyleBackColor = false;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click\_1);

//

// trackBar1

//

this->trackBar1->BackColor = System::Drawing::SystemColors::MenuHighlight;

this->trackBar1->Location = System::Drawing::Point(1266, 248);

this->trackBar1->Maximum = 5;

this->trackBar1->Minimum = 1;

this->trackBar1->Name = L"trackBar1";

this->trackBar1->Size = System::Drawing::Size(140, 45);

this->trackBar1->TabIndex = 7;

this->trackBar1->Value = 1;

this->trackBar1->Scroll += gcnew System::EventHandler(this, &MyForm::trackBar1\_Scroll);

//

// trackBar2

//

this->trackBar2->BackColor = System::Drawing::SystemColors::MenuHighlight;

this->trackBar2->Location = System::Drawing::Point(1266, 171);

this->trackBar2->Maximum = 5;

this->trackBar2->Minimum = 1;

this->trackBar2->Name = L"trackBar2";

this->trackBar2->Size = System::Drawing::Size(140, 45);

this->trackBar2->TabIndex = 8;

this->trackBar2->Value = 1;

this->trackBar2->Scroll += gcnew System::EventHandler(this, &MyForm::trackBar2\_Scroll);

//

// label2

//

this->label2->AutoSize = true;

this->label2->BackColor = System::Drawing::SystemColors::HotTrack;

this->label2->Location = System::Drawing::Point(1280, 145);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(86, 13);

this->label2->TabIndex = 9;

this->label2->Text = L"Толщина линии";

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::SystemColors::GradientActiveCaption;

this->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"$this.BackgroundImage")));

this->ClientSize = System::Drawing::Size(1454, 599);

this->Controls->Add(this->label2);

this->Controls->Add(this->trackBar2);

this->Controls->Add(this->trackBar1);

this->Controls->Add(this->colorButton);

this->Controls->Add(this->button2);

this->Controls->Add(this->SaveButton);

this->Controls->Add(this->pictureBox1);

this->Controls->Add(this->label1);

this->Controls->Add(this->button1);

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"MyForm";

this->RightToLeft = System::Windows::Forms::RightToLeft::No;

this->Text = L"Спираль Архимеда";

this->Load += gcnew System::EventHandler(this, &MyForm::MyForm\_Load);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar2))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private:

//Переменные, необходимые для реализации программы

int spiralSize;

int spiralColor;

int size = 20;

Color color = Color::Blue;

int tol = 1;

private: System::Void button1\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

// Очищаем box

pictureBox1->Refresh();

SpiralDraw(size, color);

SetkaDraw(size, Color::White);

}

private: System::Void SaveButton\_Click(System::Object^ sender, System::EventArgs^ e) {

// Определяем область, которую хотим сохранить (в данном случае - область pictureBox)

Rectangle rect = pictureBox1->Bounds;

// Создаем Bitmap с размерами области

Bitmap^ bitmap = gcnew Bitmap(rect.Width, rect.Height);

// Создаем Graphics для Bitmap

Graphics^ graphics = Graphics::FromImage(bitmap);

// Копируем область pictureBox на Graphics

Point location = pictureBox1->PointToScreen(Point(0, 0));

graphics->CopyFromScreen(location, Point(0, 0), rect.Size);

// Освобождаем ресурсы Graphics

delete graphics;

// Открываем диалоговое окно для выбора места сохранения файла

SaveFileDialog^ saveFileDialog = gcnew SaveFileDialog();

saveFileDialog->Filter = "PNG Image (\*.png)|\*.png|JPEG Image(\*.jpg)|\*.jpg|Bitmap Image(\*.bmp)|\*.bmp";

saveFileDialog->Title = "Save Image";

if (saveFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

// Если пользователь выбрал место сохранения файла

if (saveFileDialog->FileName != "") {

// Сохраняем Bitmap в выбранном месте

bitmap->Save(saveFileDialog->FileName, Imaging::ImageFormat::Png);

}

}

// Освобождаем ресурсы Bitmap

delete bitmap;

}

void SpiralDraw(int size, Color color) {

Graphics^ graphics = pictureBox1->CreateGraphics();

Pen^ pen1 = gcnew Pen(color);

Pen^ pen2 = gcnew Pen(color);

//размеры picture box

int width = pictureBox1->Width;

int height = pictureBox1->Height;

//Изображаем координатную ось

graphics->DrawLine(pen2, 0, height / 2, width, height / 2);

graphics->DrawLine(pen2, width / 2, 0, width / 2, height);

pen1->Width = tol;

float r = 0;

float x = 600;

float y = 300;

float x1, y1;

//Цикл, для изображения спирали

for (int alpha = 0; alpha < 360 \* 10; alpha +=10) {

r = size \* alpha / 360;

x1 = 600 - sin(alpha / 180.0 \* 3.14) \* r;

y1 = 300 + cos(alpha / 180.0 \* 3.14) \* r;

graphics->DrawLine(pen1, x, y, x1, y1);

x = x1;

y = y1;

}

}

void SetkaDraw(int size, Color color) {

Graphics^ graphics = pictureBox1->CreateGraphics();

Pen^ pen1 = gcnew Pen(color);

float r = 0;

float x1, y1;

//цикл, для изображения сетки

for (int alpha = 0; alpha < 361 \* 10; alpha += 10) {

r = size \* alpha / 360;

x1 = 600 - sin(alpha / 180.0 \* 3.14) \* r;

y1 = 300 + cos(alpha / 180.0 \* 3.14) \* r;

graphics->DrawLine(pen1, 600, 300, 600 - sin(alpha / 180.0 \* 3.14) \* r, 300 + cos(alpha / 180.0 \* 3.14) \* r);

}

}

private: System::Void colorButton\_Click(System::Object^ sender, System::EventArgs^ e) {

//Изменение цвета спирали

ColorDialog^ colorDialog1 = gcnew ColorDialog();

if (colorDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

color = colorDialog1->Color;

}

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e) {

//Изменение заднего фона

ColorDialog^ colorDialog2 = gcnew ColorDialog();

if (colorDialog2->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

Color selectedcolor = colorDialog2->Color;

pictureBox1->BackColor = selectedcolor;

}

}

private: System::Void label1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void MyForm\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void pictureBox1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void trackBar1\_Scroll(System::Object^ sender, System::EventArgs^ e) {

//Масштаб спирали

pictureBox1->Refresh();

int value = trackBar1->Value;

size = value\*20;

}

private: System::Void trackBar2\_Scroll(System::Object^ sender, System::EventArgs^ e) {

//толщина линии

pictureBox1->Refresh();

tol = trackBar2->Value;

}

};

}