МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЖИТОМИРСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Кафедра інженерії програмного забезпечення

**КУРСОВИЙ ПРОЕКТ**

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: «Об’єктно-орієнтоване програмування»

на тему:

**«**Розробка програми побудови фракталів**»**

студента І курсу групи ІПЗ-20-4

спеціальності 121 «Інженерія програмного забезпечення»

Кормиша Романа Івановича

(прізвище, ім’я та по-батькові)

Керівник: доцент кафедри КН к.т.н., доцент Морозов А. В.

Дата захисту: " \_\_\_ " \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Л. Левківський .

(підпис) (прізвище та ініціали) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Г.В. Марчук .

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Чижмотря .

(підпис) (прізвище та ініціали)

Житомир – 2021

ЖИТОМИРСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет інформаційно-комп’ютерних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень: бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

Зав. кафедри

\_\_\_\_\_\_\_\_\_\_\_\_ А.В.Морозов

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_р.

ЗАВДАННЯ

НА КУРСОВИЙ ПРОЕКТ СТУДЕНТУ

Данилюку Віталію Анатолійовичу

1. Тема роботи: Розробка програми побудови фракталів.

керівник роботи: доцент кафедри комп’ютерних наук, к.т.н., доцент Морозов Андрій Васильович.

1. Строк подання студентом: “ ” травня 20\_\_р.
2. Вихідні дані до роботи: Розробити програму побудови фракталів, яка дозволить будувати фрактали відповідно до введенех користувачем даних та зберігати їх як зображення.
3. Зміст розрахунково-пояснювальної записки(перелік питань. Які підлягають розробці)
   * + 1. Постановка завдання
       2. Аналіз аналогічних розробок
       3. Алгоритми роботи програми

4. Опис роботи програми

5. Програмне дослідження

1. Перелік графічного матеріалу(з точним зазначенням обов’язкових креслень)

1.Презентація до КП\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Посилання на репозиторій: https://gitlab.com/2020-2024/ipz-20-4/kormysh-roman/coursework.git

1. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посади консультанта | Підпис, дата | |
| завдання  видав | завдання прийняв |
| 1,2 | Левківський В.Л., ст. викладач каф. КН |  |  |
| 1,2 | Марчук Г.В., ст. викладач каф. КН |  |  |
| 1,2 | Чижмотря О.В., ст. викладач каф. ІПЗ |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Дата видачі завдання “ 17 ” лютого 2021 р.

|  |  |  |  |
| --- | --- | --- | --- |
| № з/п | Назва етапів курсового проекту | Строк виконання етапів проекту | Примітки |
| 1 | Постановка задачі | 17.02.21 – 28.02.21 |  |
| 2 | Пошук, огляд та аналіз аналогічних розробок | 28.02.21 – 01.03.21 |  |
| 3 | Формулювання технічного завдання | 01.02.21 – 04.03.21 |  |
| 4 | Опрацювання літературних джерел | 04.02.21 – 05.03.21 |  |
| 5 | Проектування структури | 05.02.21 – 11.03.21 |  |
| 6 | Написання програмного коду | 12.03.21 –  26.03.21 |  |
| 7 | Відлагодження | 26.03.21 – |  |
| 8 | Написання пояснювальної записки |  |  |
| 9 | Захист |  |  |

**КАЛЕНДАРНИЙ ПЛАН**

**Студент** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кормиш Р.І.

(підпис) (прізвище та ініціали)

**Керівник проекту** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Власенко О.В.

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до курсового проекту на тему «Програма побудови фракталів» складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатку.

Текстова частина викладена на 33 сторінках друкованого тексту.

Пояснювальна записка має 66 сторінку додатків. Список використаних джерел містить 5 найменувань і займає 1 сторінку. В роботі наведено 17 рисунків. Загальний обсяг роботи – 102 сторінки.

У першому розділі було обґрунтовано створення програми для створення фракталів та збереження їх зображень в різних форматах.

У другому розділі проведено проектування і розробка програмного продукту.

У третьому розділі проведено тестування програмного продукту.

Висновок містить в собі результати виконаної роботи створення програми для зберігання, пошуку і відображення інформації про країни.

У додатку представлений лістинг розробленого програмного продукту.

Ключові слова: ООП, ФРАКТАЛИ, МНОЖИНА МАНДЕЛЬБРОТА, КРИВА ХАРТЕРА - ХЕЙТУЕЯ, ПАПОРОТНИК БАРНСЛІ, ФРАКТАЛЬНЕ ДЕРЕВО.

# 

**Зміст**

[ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 6](#_Toc70621616)

[ВСТУП 7](#_Toc70621617)

[РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 8](#_Toc70621618)

[1.1 Аналіз задачі, засобів та методів її вирішення 8](#_Toc70621619)

[1.2 Аналіз існуючого програмного забезпечення за тематикою курсової роботи. 10](#_Toc70621620)

[Висновки до першого розділу : 14](#_Toc70621621)

[РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 15](#_Toc70621622)

[2.1 Проектування загального алгоритму роботи програми 15](#_Toc70621623)

[Загальна схема роботи програми : 16](#_Toc70621624)

[2.2 Розробка функціональних алгоритмів роботи програми 17](#_Toc70621625)

# ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

VS – Visual Studio

ПЗ – Програмне забезпечення

ППФ – Програма побудови фракталів

# ВСТУП

У цій курсовій роботі буде наведено процес створення програми побудови фракталів.

Програми побудови фракталів — це комп’ютерна програма, яка будує фрактали опираючись на фрактальні формули. Подібні програми призначені для візуалізації фрактальних формул, створенню фрактальної графіки, детального огляду створених фракталів та їх збереження. Тому в роботі ППФ можна виділити два основних етапи: перший — побудова фракталу за формулою, другий — робота зі створеним фракталом та його збереження.

    ППФ широко застосовуються в науці. Найбільш корисним застосування фракталів є у комп'ютерній науці - фрактальне стиснення даних. Фрактали використовуються в нафтохімії при моделюванні пористих матеріалів, в біології для опису внутрішніх органів, в медицині для відображення биття серця, у фізиці при моделюванні нелінійних процесів. В природі фрактальними властивостями володіють безліч об'єктів - кора древ, хмари, сніжинки тощо. Фрактали використовують в комп'ютерних іграх, де рельєфна місцевість, зазвичай, є фрактальний зображенням. Слідуючи з цих тверджень, фрактали та фрактальна графіка потрібні усюди, тому створення ППФ є актуальним в наш час.

**Метою створення** даного курсового проекту є розробка програми для побудови фракталів з додатковими можливостями для їх дослідження.

**Об’єктом дослідження** є технології побудови фракталів.

**Предмет дослідження** – є вивчення можливостей інтерфейсу програмування додатків Windows Fomrs (WF) та використання їх для побудови фракталів.

# РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

## **Аналіз задачі, засобів та методів її вирішення**

Задача полягає в тому, що потрібно:

1. Визначити потреби звичайного користувача
2. Вияснити способи реалізації відмальовування фракталів на формі
3. Зрозуміти алгоритми побудови фракталів .
4. Продумати можливість приближення, віддалення , повернення до початку та управління програмою за допомогою “hot keys”.
5. Налагодити можливість збереження фракталу як зображення в різних форматах.
6. Передбачити можливість задання параметрів фракталу користувачем.
7. Реалізувати адаптивність форм.
8. Продумати дизайн та оформлення програми.
9. Реалізувати візуалізацію процесу побудови фракталу.

В процесі створення ППФ бралися різні аспект проекту, розроблялася найпростіша версія, а після того вдосконалювалася, чи повністю перероблювалася.

Для реалізації даного програмного продукту чудово підходить інтегроване середовище розробки програмного забезпечення Visual Studio, а саме платформа Windows Forms (WF), адже вона має велику кількість елементів управління та забезпечує високий рівень стабільності роботи програми. Також слід обрати платформу .Net Framework і мову програмування C#.

При реалізації треба використовувати об’єктно-орієнтований підхід, тому що тоді завдання спрощується і з’являється можливість оперувати більшими по об’єму та складнішими програмами. Також це допомагає уникнути деяких помилок. Також у подальшому, з легкістю, можна використовувати даний проект в іншому, або переробити при необхідності. За рахунок використання класів можна легко модифікувати існуючі елементи без зміни вже готових.

## **1.2 Аналіз існуючого програмного забезпечення за тематикою курсової роботи.**

При аналізі вже існуючого *ПЗ* за тематикою курсової роботи було виявлено декілька проектів. Всі вони схожі один на одного і мають схожу реалізацію, адже всі вони розроблені для побудови фракталів та створення фрактальної графіки.

Aналогічні програмні додатки за заданою темою:

1. Ultra Fractal.

Розробник: Frederik Slijkerman.

Мова програмування: Delphi.

Посилання на ПО: <https://www.ultrafractal.com/>

Розглянемо перший аналог програму Ultra Fractal:

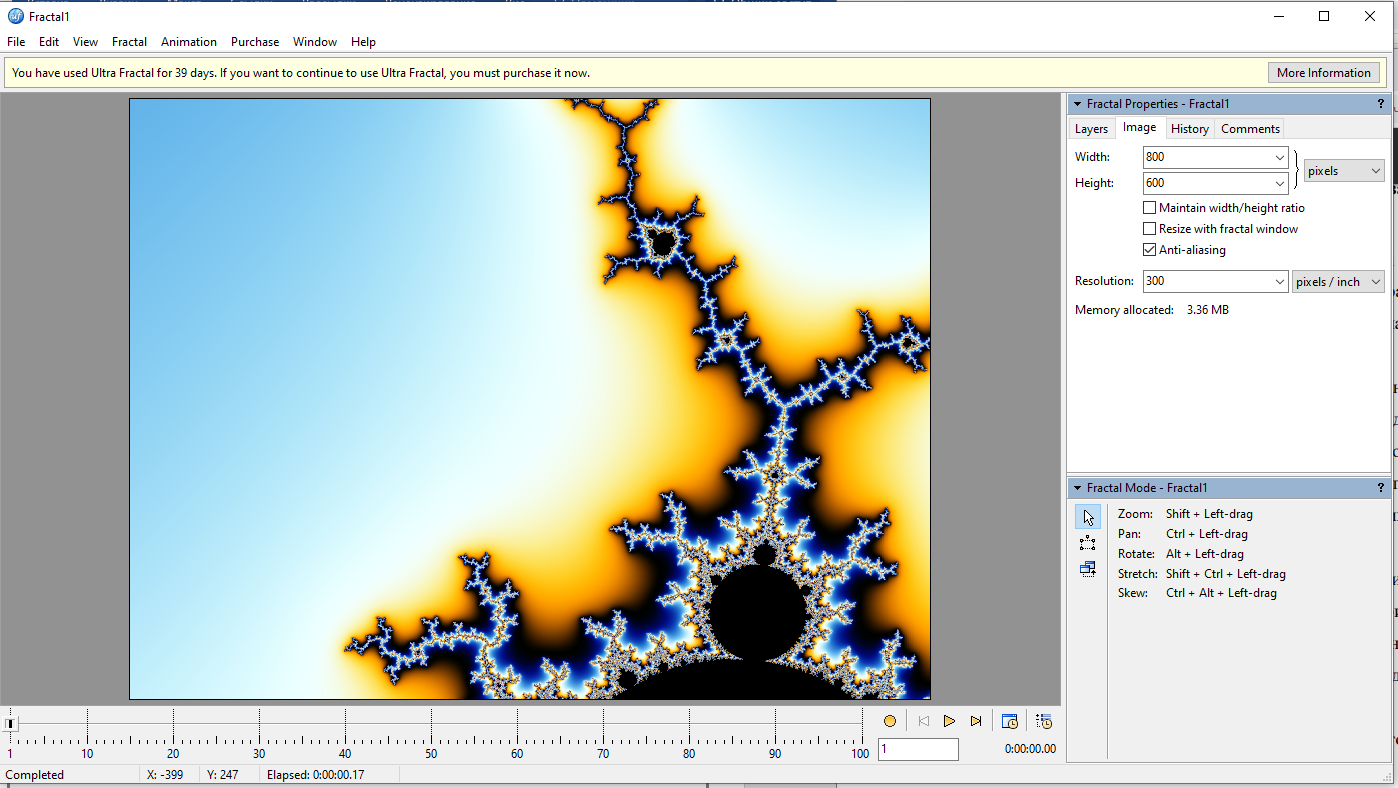


Рис 1.1 Ultra Fractal

Програма Ultra Fractal(Рис 1.1) є одним із лідерів по популярності серед ППФ, ажде вона має зручний користувацький інтерфейс, можливість збереження фракталу в кожен момент зміни та має можливість запису змін(“анімація”). А також плавний перехід (хоча із погіршення якості) в момент зміни масштабу і можливість задання власної формули. Але незважаючи на зручність інтерфейсу, він має один серйозний недолік –

надмірна кількість елементів управління, які можуть заважати роботі. А також не зручно реалізована механіка приближення. Також дана програма є комерційною, адже надається доступ до програми на безплатній основі на деякий період.

Основні плюси:

* Зручний користувацький інтерфейс
* Можливість задання власної формули
* Стабільність роботи програми

Основні мінуси:

* Надмірність елементів управління
* Комерційність
* Не зручна реалізація механіки приближення

1. Генератор фракталів від Nadin desing

Розробник: Надія Трубочкіна

Мова програмування: JScript.

Посилання на ОП: <http://nadin.miem.edu.ru/1111/index.html>

Розглянемо другий аналог Генератор фракталів від Nadin desing:

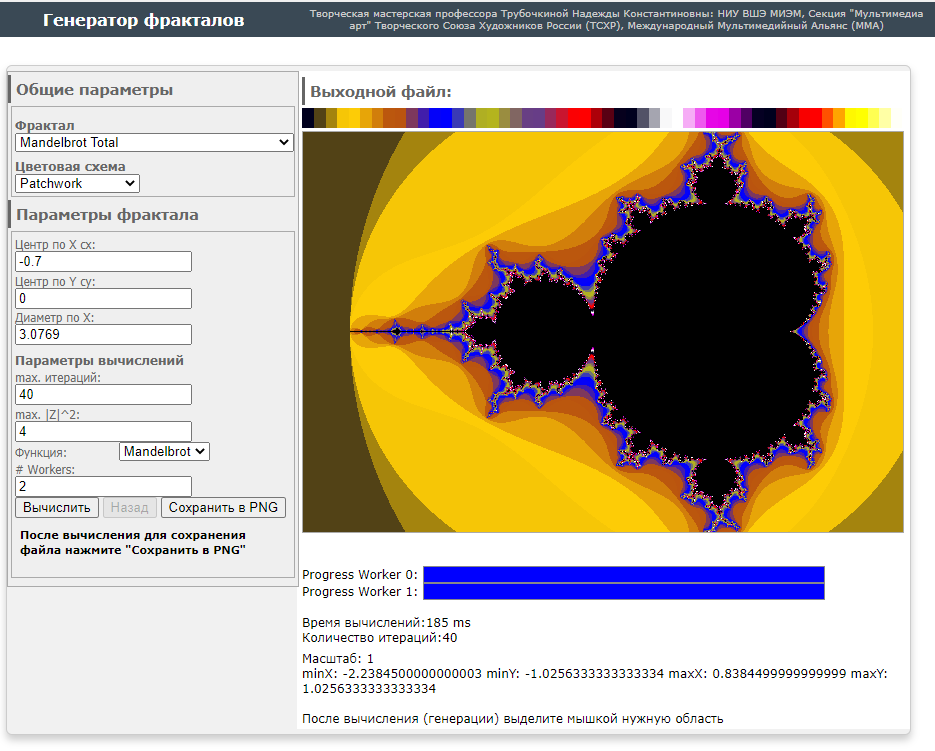


Рис 1.2 Генератор фракталів від Nadin desing.

Програма генератор фракталів від Nadin desing (Рис 1.2) є лідером серед мережевих ППФ. Вона має мінімалістичний користувацький інтерфейс, вибір в загальному тільки серед двох фракталів таких, як множина Мандельброта та множина Жулья. Також мінусом є те, що при великому значенні масштабування процес процес створення відбувається надто довго. Має можливість збереження зображення фракталу тільки в форматі PNG. Механіка масштабування краща ніж у Ultra Fractal.

Основні плюси:

* Зручний користувацький інтерфейс
* Стабільність роботи програми
* Механіка приближення реалізована зручно

Основні мінуси:

* Обмеженість вибору двома фракталами
* Довге виконання обчислень при великому приближенні
* Збереження зображення фракталу тільки в форматі PNG

1. Fractal Explorer

Розробник: Sirotinsky Arthur

Посилання: <http://www.fractal-explorer.com/>

Розглянемо третій аналог Fractal Explorer:

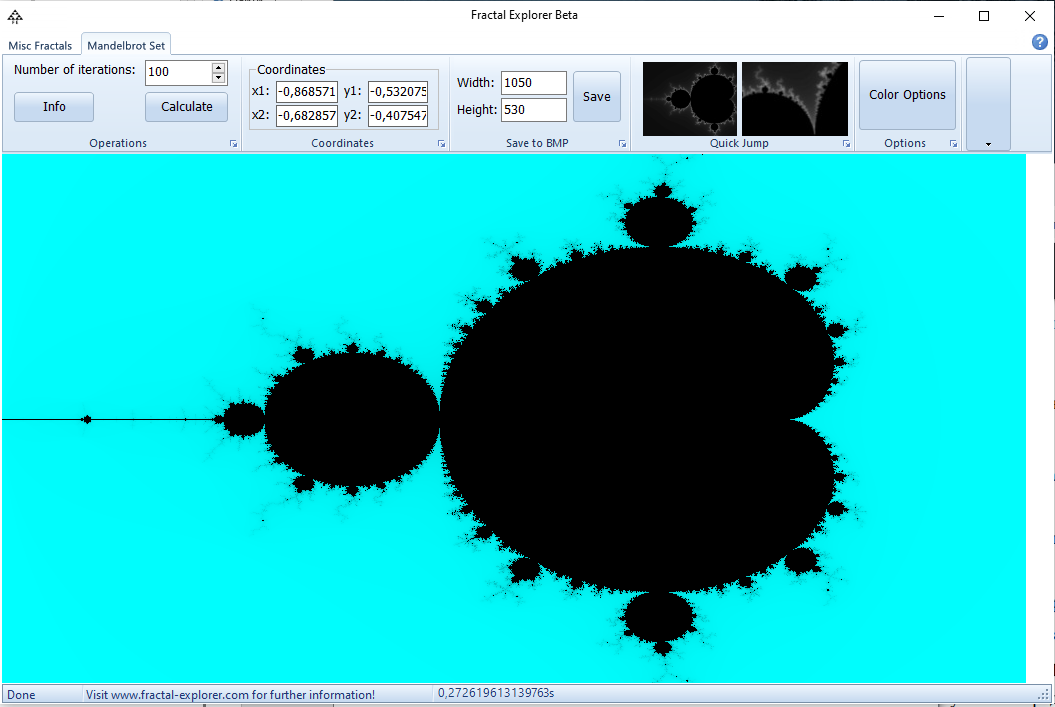


Рис 1.3 Fractal Explorer.

Програма Fractal Explorer (Рис 1.3) займає позиції біля середини списку по популярності. Вона має мінімальний користувацький інтерфейс і є застарілим в візуальному плані, вибір на побудову фракталів такий : множина Мандельброта, сніжинка та крива Коха і трикутник Серпинського.

Користувач може вибирати тільки розміри зображення і фрактал не масштабується при зміні розміру вікна. Дуже серйозним мінусом є те, що програма при вичисленнях стає неактивною, а також при обчисленнях може видати дуже багато помилок і закритися.

Основні плюси:

* Зручний користувацький інтерфейс
* Ширший вибір для побудови фракталів ніж в попередній розглянутій програмі

Основні мінуси:

* Застарілий в візуальному плані
* Не стабільний при використанні
* Деколи під час побудови фракталу програма стає неактивною і видає велику кількість помилок

Актуальним напрямком реалізації власного продукту є побудова фракталів методами схожими з методами Ultra Fractal , реалізувати користувацький інтерфейс, уникаючи надмірності Ultra Fractal, але послідувати основним аспектам інтерфейсу цього продукту та звертатися до методів продукту, приведеного на Рис 1.2, також забезпечити стабільність програми та можливість масштабування.

## **Висновки до першого розділу :**

У ході виконання першого розділу було поставлено завдання проекту, а саме який має бути функціонал , де буде застосовуватися і визначено актуальність проекту. Також обрано платформу та мову розробки, визначено актуальний напрямок розробки власного продукту. Розглянуто існуючі програмні продукти зі схожою тематикою та більш детально проаналізовано деякі із них, а саме : Ultra Fractal, Генератор фракталів від Nadin desing, та Fractal Explorer.

# РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## **2.1 Проектування загального алгоритму роботи програми**

На даному етапі розробки розглянемо загальний алгоритм роботи ППФ.

Загальна схема роботи ППФ приведена на рисунку 2.1.

Спочатку буде створено функції для створення фракталів насамперед в основні формі, але після створення бібліотек класів буде перероблено функції та перенесено їх до відповідної бібліотеки *FractalClasses*, яка міститиме фрактальні класи : *MBrotSet, FractalTree, BarnsleyFern* та *CurveDragon*.

Також буде створена друга бібліотека класів, яка містила допоміжні класи *HelperClasser*, яка міститиме клас для збереження кольорів конкретних пікселів *Pixel* та клас для комплексних чисел *Complex*.

Наступним кроком після побудови фракталів буде реалізація їх збереження. Що ми зможемо виконати за допомогою діалогових вікон. А для можливості збереження в різних форматах використано властивість *filter*.

Для можливості та зручності використання ППФ буде добавлено багато елементів управління та реалізовано “*hot keys*”.

Також для зручності задання кольорів фракталів буде реалізовано окреме вікно та додано можливості генерації градієту і завантаження користувачем відповідного шаблону кольорів.

Тож, ППФ матиме вигляд декількох вікон зв’язаних між собою, які зручний користувацький інтерфейс, де користувач зможе задавати параметри обраних фракталів.

При запуску програми користувач буде бачити перед собою побудовану множину Мандельброта і зможе вибрати інший фрактал в *ComboBox.* Далі вводяться початкові дані і перевіряються на коректність.

Для множини Мандельброта початковими даними є точка, відносно якої будується фрактал, квадрат максимального значення функції та кількості ітерацій. При введенні некоректних даних повинна виводитися іконка з попередженням.

Для фрактального дерева початковими даними є довжина гілки, мінімальна довжина гілки, кількість гілок, градуси відхилу гілок від кореневої гілки, початкові координати та ширина гілки. Кількість гілок в даній ППФ користувач може обирати за допомогою *ComboBox*, який має на вибір кількість гілок від 1 до 4. Далі користувач також може задати задній фон для фракталу.

Для папоротника Барнслі початковими даними є кількість точок, які будуть формувати фрактал, звуження по висоті і ширині, а такаж задання заднього фону фракталу.

Останнім фракталом на вибір є крива Хартера-Хемінтуея, початковими даними для якої є кількість кривих, координати точок початку і кінця цих кривих, ширина кісті, яка задає ширину лінії кривої, кількість ітерацій, яка задає кількість згинів кривої, а також задання заднього фону фракталу.

Після обрання фракталу та введення коректних даних користувач нажиматиме на кнопку побудови, яка розміщена в лівому нижньому кутку вікна.

Під час побудови фракталу під кнопкою буде заповнюватися progressBar, який відображатиме процес побудови заданого фракталу. Також є можливість зміни колірної палітри фракталу, за допомогою задання градієнту.

Далі користувач може зберегти створений фрактал та далі продовжити працювати з програмою.

## Загальна схема роботи програми :

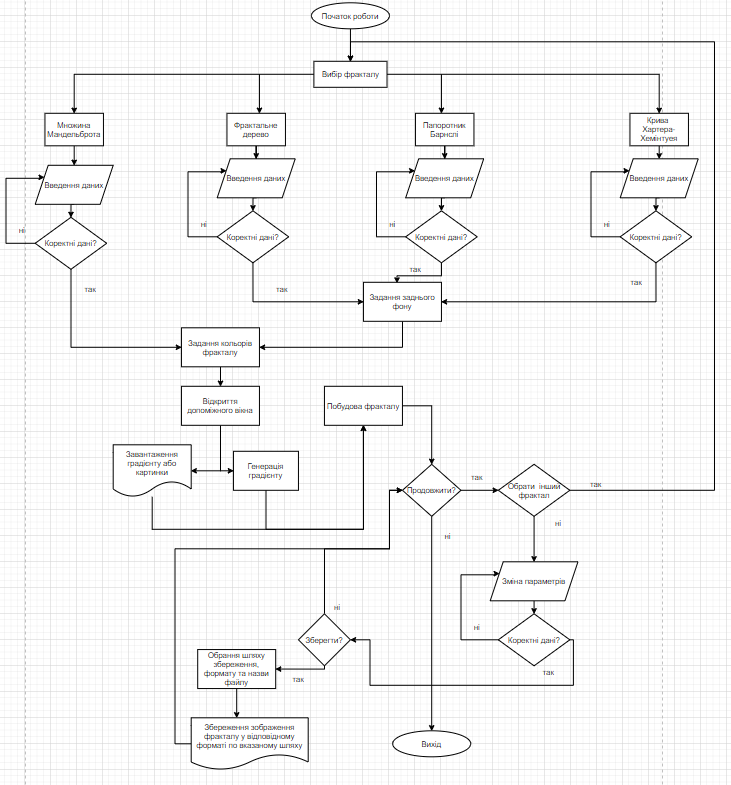


Рисунок 2.1 – Загальна схема роботи програми

## **2.2 Розробка функціональних алгоритмів роботи програми**

Залишилося розглянути детальніше алгоритми робити основних функцій.

Звичайно, основне призначення ППФ – побудова фракталів, тому розглянемо саме це. Для малювання на формі треба використовувати елемент класу Graphics. Для побудови кожного фракталу використовується відповідна йому фрактальна формула, після поітераційного обчислення цієї формули ми отримуємо зображення фракталу. Для зручності огляду будь-якого фракталу потрібно використовувати палітру кольорів(градієнт), що вирізняти різні його елементи.

При обранні у програмі фракталу, задаються початкові значення автоматично, які користувач може змінити. В залежності від того, який обирається фрактал, надаються різні поля вводу даних.

Сам алгоритм полягає в тому, що при нажатті на кнопку побудови фракталу, Створюється об’єкт з відповідним класом, який наслідується від базового класу FractalClasses, виконуються необхідні операції, а після їх виконання передається створене зображення фракталу в *PictureBox*.

Для виконання побудови множини Мандельброта використовуються об’єкти типу *Complex*, які відповідають за комплексні числа, адже формула цього фракталу побудована саме на них. Кольори для відображення кожного фракталу зберігаються в *List* типу *Pixel*.

При роботі зі збереженням та завантаженням даних в ППФ використовувався простий алгоритм перевірки на порожнечу та відповідність параметрам. При збереженні спочатку перевірялося чи зображення фракталу існує і чи існує картинка з заданою користувачем назвою, а потім виконувалося збереження.

При завантаженні градієнту виконується алгоритм перевірки на порожнечу, а Також для перевірки на відповідність параметрам було додано алгоритм

перевірки розрішення картинки. За умовою якого ширина картинки має становити не меньше за 482 пікселі. Якщо умова не виконується відображається відповідне попередження.

Для побудови фракталів було розроблено алгоритми генерації відповідних фракталів за допомогою опису відповідних фрактальних формул. При зміні параметрів запускаються відповідні алгоритми які залежать як від даних введених користувачем так і від розміру віконного додатку. Також передбачені алгоритми для уникнення виникнення накладок як програмного коду так і візуального відображення, які забезпечують зручність використання продукту та його стабільність.

## Діаграма класів та компонентів :

Рисунок 2.2 – Діаграма класів та компонентів

## **2.3 Розробка програмного забезпечення**

Структура рішення проекту :

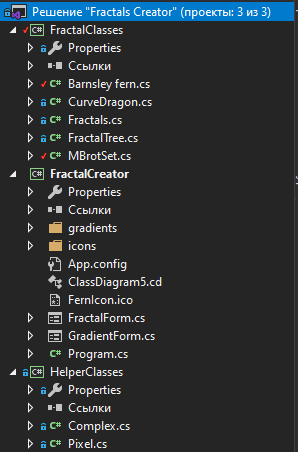


Рисунок 2.3 – Структура рішення проекту

*FractalClasses* – це бібліотека класів, яка містить , як не дивно, фрактальні класи. В ній містяться 5 класів: *BurnsleyFern, CurveDragon, FractalTree, MBrotSet* і базовий клас *Fractals.*

*HelperClasses -* це бібліотека класів, яка містить, допоміжні класи. В ній містяться 2 класи: *Сomplex* та *Pixel.*

Фрактальнікласи і є основними в даному проекті. Тож розглянемо їх детальніше.

Клас *MBrotSet* відповідає за побудову множини Мандельброта, яка описується за допомогою відповідної формули : *fc(z) = z2+c,* де *c* і *z* комплексні числа. Для для можливості виконання дій над комплексними числами було створено клас *Complex.*

Цей клас має такі методи, як *Sqr*, який забезпечує піднесення комплексного числа до квадрату,

public void Sqr()

{

double tmp = (Re \* Re) - (Im \* Im);

Im = 2.0d \* Re \* Im;

Re = tmp;

}

*Magn(Magnitude)*, який повертає модуль комплексного числа,

public double Magn()

{

return Math.Sqrt((Re \* Re) + (Im \* Im));

}

і метод *Add*, який додає два комплексних числа.

public void Add(Complex c)

{

Re += c.Re;

Im += c.Im;

}

Саме ці методи забезпечують можливісті виконання операцій над комплексними числами. Звичайно, тут описані не всі методи виконання дій над комплексними числами, але в даному проекті їх достатньо.

Якщо при роботі програми обрана множина Мандельброта, тоді викликається метод *CalculationMBrot*, який повертає об’єкт типу *Bitmap* з зображенням фракталу. Де основною частиною є розрахунок комплексного числа *z:*

do

{

it++;

z.Sqr();

z.Add(c);

if (z.Magn() > maxZ)

{

break;

}

} while (it < UserIt);

Саме цей цикл і є реалізацією фрактальної формули Мандельброта.

Також *Bitmap* поверає метод *DrawBransleyFern*, який будує папоротник Барнслі. Основним циклом якого є вираховування можливостей формул заданих коефіцієнтами:

double randomNum = random.NextDouble();

for (int j = 0; j < probability.Length; j++)

{

randomNum -= probability[j];

if (randomNum <= 0)

{

FunctionIndex = j;

break;

}

}

, де коефіцієнти можливостей задані масивом float[] probability = new float[4] { 0.01f, 0.06f, 0.08f, 0.85f };, а формули задані коефіцієнтами формул

float[,] Coefficient = new float[4, 6]

{

{0, 0, 0, 0.16f, 0, 0},

{-0.15f, 0.28f, 0.26f, 0.24f, 0, 0.44f},

{0.2f, -0.26f, 0.23f, 0.22f, 0, 1.6f},

{0.85f, 0.04f, -0.04f, 0.85f, 0, 1.6f}

};

, де кожна формула задана 6 коефіцієнтами, які задовільняють формулу

, де кожна літера відповідає індексації елементу в масиві, а початковим x і y беруться точки початку побудови.

Інші методи, а саме *DrawCurveDragon* та *DrawFractalTree* не повертають такого об’єкту, адже виконуються за допомогою рекурсивних функцій та мають ти *void*.

Основною частиною методу *DrawFractalTree* є розрахунок наступної точки для побудови лінії:

int x1, y1;

x1 = (int)(x + len \* Math.Sin((2 \* Math.PI \* angle) / 360.0));

y1 = (int)(y + len \* Math.Cos((2 \* Math.PI \* angle) / 360.0));

, де *angle* це кут нахилу гілки відносно попередньої гілки , ця дія виконується циклічно зі зміною довжини(*len*) та кута(*angle*) лінії(гілки) , що і забезпечує побудову фракталу в формі дерева.

for (int j = 0; j < angles.Length; j++)

{

DrawFractalTree(x1, y1, (int)(len / 1.5), angle + angles[j], progress);

}

Зазвичай будують фрактальне дерево з двома основними вітками, але в даному проекті є можливість спробувати побудувати як із однієї так із трьох чи чотирьох гілок. Швидкість виконання побудови залежить від кількості новостворених гілок, чим більша різниця між довжинами гілок та чим їх більше тим повільніще відбувається процес побудови.

Для побудови фрактала Хартера-Хейтуея використовується не складний алгоритм, який полягає в діленні відрізку навпіл під кутом 90о

NextX = (int)((x1 + x2) / 2 + (y2 - y1) / 2);

NextY = (int)((y1 + y2) / 2 - (x2 - x1) / 2);

, де *NextX* і *NextY -* це координати точки поділу лінії, в яких формулах яких перша частина це точка сережини відрізку, а друга координати точки відносно лінії під кутом 90о Ці операції виконуються циклічно, що і забезпечує самоподібність і оптичну складність даного фракталу.

Крім фрактальних методів, є також важливий метод отримання кольору пікселів з градієнту чи з картинки:

public List<Pixel> GetPixels(Bitmap bitmap)

{

List<Pixel> pixels = new List<Pixel>(bitmap.Width);

for (int x = 0; x < bitmap.Width; x++)

{

pixels.Add(new Pixel()

{

Color = bitmap.GetPixel(x, bitmap.Height / 2) // отримання кольору ряду пікселів з градієнту

});

}

return pixels;

}

Цей метод, можна так сказати, вирізає один ряд пікселів посередині картинки чи градієнту і передає його у вигляді списку кольорів. Потім цей список використовується як градієнт або набір кольорів для розфарбування фракталів.

## **Висновки до другого розділу:**

Спроектовано загальну схему програми, діаграму класів. Визначено та детально описано основні можливості програми. Пояснено алгоритми найважливіших методів ППФ та приведено код їх основних частин. Приведено приклади коду деяких методів. Роз’яснено деякі нюанси роботи програми. Показано структуру проекта.