МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

Кафедра

інформатики та програмної інженерії

(повна назва кафедри, циклової комісії)

**КУРСОВА РОБОТА**

з \_\_основ програмування\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва дисципліни)

на тему: \_\_8-puzzle\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студента (ки, ів) 1 курсу, групи ІП-14

Тітова Романа Едуардовича

Спеціальності 121 «Інженерія програмного забезпечення»

Керівник старший викладач Головченко М. М. \_

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

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

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Члени комісії |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |
|  |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |

Київ - 2022 рік

КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

(назва вищого навчального закладу)

Кафедра інформатики та програмної інженерії

Дисципліна Основи програмування

Напрям "ІПЗ"

Курс 1 Група ІП-14 Семестр 2

**ЗАВДАННЯ**

на курсову роботу студента

Тітова Романа Едуардовича

(прізвище, ім’я, по батькові)

1. Тема роботи Задача “8-puzzle”

2. Строк здачі студентом закінченої роботи 12.06.2022

3. Вихідні дані до роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

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

6. Дата видачі завдання 10.02.2022

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Підписи керівника, студента |
| 1. | Отримання теми курсової роботи | 10.02.2022 |  |
| 2. | Підготовка ТЗ | 15.03.2022 |  |
| 3. | Пошук та вивчення літератури з питань курсової роботи | 13.04.2022 |  |
| 4. | Розробка сценарію роботи програми | 15.04.2022 |  |
| 5. | Узгодження сценарію роботи програми з керівником | 17.04.2022 |  |
| 6. | Розробка (вибір) алгоритму рішення задачі | 17.04.2022 |  |
| 7. | Узгодження алгоритму з керівником | 18.04.2022 |  |
| 8. | Узгодження з керівником інтерфейсу користувача | 20.04.2022 |  |
| 9. | Розробка програмного забезпечення | 05.05.2022 |  |
| 10. | Налагодження розрахункової частини програми | 17.05.2022 |  |
| 11. | Розробка та налагодження інтерфейсної частини програми | 20.05.2022 |  |
| 12. | Узгодження з керівником набору тестів для контрольного прикладу | 05.06.2022 |  |
| 13. | Тестування програми | 07.06.2022 |  |
| 14. | Підготовка пояснювальної записки | 11.06.2022 |  |
| 15. | Здача курсової роботи на перевірку | 12.06.2022 |  |
| 16. | Захист курсової роботи | 18.06.2022 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Студент

(підпис)

Керівник Головченко М. М.

(підпис) (прізвище, ім’я, по батькові)

"\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ р.

**АНОТАЦІЯ**

Пояснювальна записка до курсової роботи: 107 сторінок, 30 рисунків, 13 таблиць, 3 посилання.

Об’єкт дослідження: задача “8-puzzle”.

Мета роботи: дослідження задачі 8-puzzle і методів(LDFS та BFS) для її вирішення, розробка програмного забезпечення з графічним інтерфейсом для вирішення задачі 8-puzzle.

Досліджено методи LDFS та BFS для вирішення задачі 8-puzzle та описано алгоритми їх роботи.

Виконана програмна реалізація вирішення задачі “8-puzzle” з використанням графічного інтерфейсу і методів пошуку в дереві LDFS та BFS.

**ЗМІСТ**

[**ВСТУП** 7](#_Toc105883958)

[**1 ПОСТАНОВКА ЗАДАЧІ** 8](#_Toc105883959)

[**2 ТЕОРЕТИЧНІ ВІДОМОСТІ** 9](#_Toc105883960)

[2.1 BFS 9](#_Toc105883961)

[2.2 LDFS 10](#_Toc105883962)

[**3** **ОПИС АЛГОРИТМІВ** 11](#_Toc105883963)

[3.1 Загальний алгоритм 12](#_Toc105883964)

[3.2 Алгоритм BFS 12](#_Toc105883965)

[3.3 Алгоритм LDFS 13](#_Toc105883966)

[**4** **ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ** 15](#_Toc105883967)

[4.2 Діаграма класів програмного забезпечення 15](#_Toc105883968)

[4.2 Опис методів частин програмного забезпечення 15](#_Toc105883969)

[4.2.1 Стандартні методи 15](#_Toc105883970)

[4.2.2 Користувацькі методи 23](#_Toc105883971)

[**5** **ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ** 37](#_Toc105883972)

[5.1 План тестування 37](#_Toc105883973)

[5.2 Приклади тестування 37](#_Toc105883974)

[**6** **ІНСТРУКЦІЯ КОРИСТУВАЧА** 46](#_Toc105883975)

[6.1 Робота з програмою 46](#_Toc105883976)

[6.2 Формат вхідних та вихідних даних 50](#_Toc105883977)

[6.3 Системні вимоги 50](#_Toc105883978)

[**7** **АНАЛІЗ І УЗАГАЛЬНЕННЯ РЕЗУЛЬТАТІВ** 52](#_Toc105883979)

[**ВИСНОВКИ** 59](#_Toc105883980)

[**ПЕРЕЛІК ПОСИЛАНЬ** 60](#_Toc105883981)

[**ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ** 61](#_Toc105883982)

[**ДОДАТОК Б ТЕКСТИ ПРОГРАМНОГО КОДУ** **Ошибка! Закладка не определена.**](#_Toc105883983)

[Info.h 65](#_Toc105883984)

[Info.cpp 65](#_Toc105883985)

[Form1.h 66](#_Toc105883986)

[Form1.cpp 88](#_Toc105883987)

[Header.h 92](#_Toc105883988)

[Func.cpp 93](#_Toc105883989)

[Puzzle.h 102](#_Toc105883990)

[Puzzle.cpp 102](#_Toc105883991)

[pch.h 105](#_Toc105883992)

[pch.cpp 106](#_Toc105883993)

[CppCLR\_WinFormsProject.cpp 106](#_Toc105883994)

[AssembleInfo.cpp 106](#_Toc105883995)

# **ВСТУП**

Дана курсова робота є застосуванням на практиці знань, які були набуті під час проходження курсів “Основи програмування 1. Базові конструкції” та “Основи програмування 2. Модульне програмування”. Під час виконання курсової роботи буде ознайомлено з проектуванням та стандартами, що використовуються при розробці програмного забезпечення на прикладі задачі “8-puzzle”.

Під час написання буде використано:

1. Навички, які були набуті під час проходження вищезазначених курсів
2. Графічний інтерфейс для зображення візуальної частини програмного забезпечення
3. Мова C++ та її бібліотеки і стандартні методи
4. Поступова розробка програмного забезпечення та документація проміжних результатів
5. Дослідження, аналіз і розробка алгоритмів для вирішення поставленої задачі
6. Ведення документації, що відповідає стандартам

Отже, під час виконання даної роботи буде набуто нових навичок розробки графічного інтерфейсу, досліджено алгоритми для виконання задачі і використано вже набутих навичок.

# **1 ПОСТАНОВКА ЗАДАЧІ**

Розробити програмне забезпечення, що буде вирішувати задачу 8-puzzle наступними методами:

а) LDFS;

б) BFS;

Вхідними даними для даної роботи є поле 3х3 і 9 фішок на ньому. Програмне забезпечення повинно обробляти місцеположення фішок і розв’язувати задачу вибраним методом. Результати роботи програми записуються у файл і можуть потім бути виведені у графічний інтерфейс.

Вихідними даними для даної роботи є фішки розташовані у правильному порядку, а також послідовність кроків необхідних для розв’язання і їх кількість та час роботи вибраного алгоритму, що і виводиться на екран.

# **2 ТЕОРЕТИЧНІ ВІДОМОСТІ**

Задача 8-puzzle полягає у тому щоб знайти шлях до стандартного розміщення фішок на полі 3x3. Ця може бути вирішена різними методами серед яких є BFS і LDFS:

## BFS

Пошук в ширину(Breadth-First Search) – це метод обходу графу в якій в першу чергу обробляється(розглядається) кореневий вузол, а потім всі його нащадки, потім нащадки нащадків і так далі. Тобто перш ніж перейти на наступний рівень дерева, обробляються всі вузли на поточній глибині.

Пошук в ширину відбувається за допомогою черги (перший елемент зайшов – перший вийшов) у яку при розгляді записуються всі нащадки розглядаємої в даний момент вершини і потім достаються в тому ж порядку з неї, оброблюєься і їх нащадки так само записуються в чергу. Процес відбувається поки не буде знайдений шуканий елемент.

Через це алгоритм знаходить найбільш оптимальний елемент, тобто шуканий елемент на мінімальній глибині дерева.

На рисунку 2.1 показано пошук у ширину у дереві.

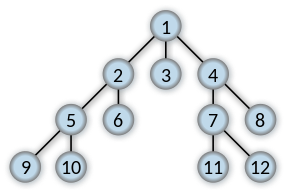


Рис. 2.1 – Пошук в ширину у дереві (цифри ілюструють порядок проходу по вершинах)

Оцінка складності. Корінь дерева породжує *b* вузлів на першому рівні, кожен з яких породжує ще b вузлів на другому рівні, що відповідає загальній кількості вузлів, що дорівнює *b2* . Кожен з них породжує ще b вузлів на третьому рівні, що відповідає загальній кількості вузлів, що дорівнює *b3* і т.д. Припустімо, що рішення знаходиться на глибині d. В найгіршому випадку на рівні d необхідно розгорнути всі вузли, крім останньому (адже сам цільовий вузол не розгортається), що призводить до обробки *bd+1-b* вузлів на рівні *d+1*. Це означає, що загальна кількість оброблених вузлів дорівнює:

*b + b2 + b3 + … bd + (bd+1 -b) = O(bd+1)*

## 2.2 LDFS

Пошук в глибину з обмеженням у глибині(Limited Depth-First Search) – це метод обходу графу при якому розгортається найбільш глибокий вузол в дереві( або найглибший до вказаної глибини пошуку). Коли вузол вже не має спадкоємців(їх не існую або вони вже оброблені), алгоритм повертається до більш поверхневого вузла у якому є недосліджені нащадки і пошук продовжується поки не буде знайдено шуканий елемент або оброблені всі вершини дерева.

Цей алгоритм реалізується по принципу стека (Перший зайшов – останній вийшов) або ж рекурсивною функцією яка викликає себе для кожного з нащадків.

На відміну від BFS алгоритму, LDFS не обов’язково знаходить найбільш оптимальний елемент( тобто той елемент-рішення, який знаходиться на мінімальній глибині) або не знаходить його взагалі на заданій глибині пошуку.

На рисунку 2.2 показано пошук у глибину з обмеженням у глибині в дереві.

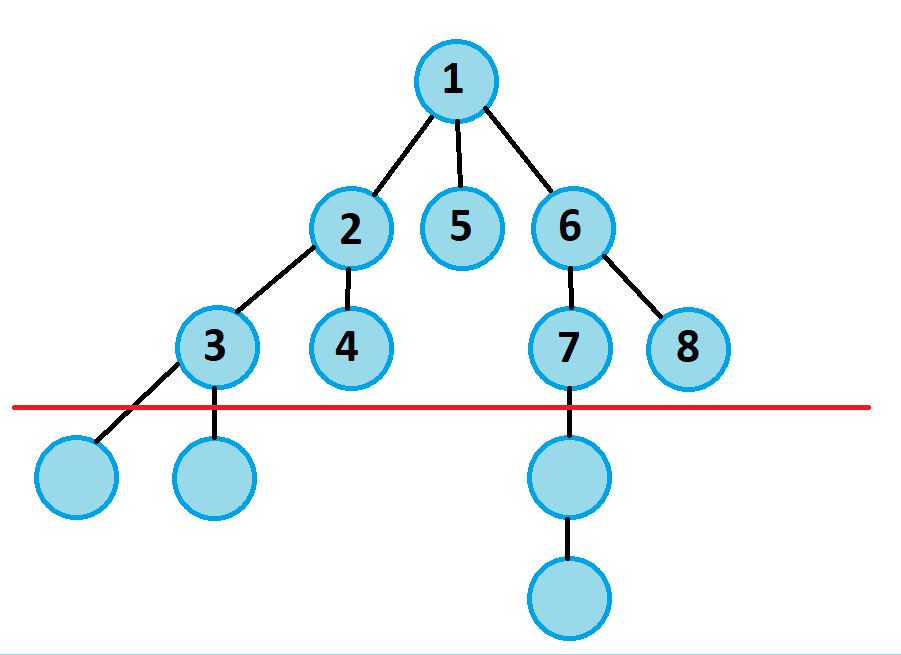


Рис. 2.2 – Пошук в глибину з обмеженням у глибині з максимальним заглибленням у 3 рівні (цифри ілюструють порядок проходу по вершинах)

Складність алгоритму відповідно – O(b^f), де – f – це визначений ліміт глибини, а b – кількість нащадків кожного вузла.

# **ОПИС АЛГОРИТМІВ**

Перелік всіх основних змінних та їхнє призначення наведено в таблиці 3.1.

Таблиця 3.1 – Основні змінні та їхні призначення

|  |  |
| --- | --- |
| Змінна | Призначення |
| pointTmp | Поточний оброблюваний вузол у BFS алгоритмі |
| vert | Поточний елемент який представляє вузол з пересунотим нулем |
| element | Поточний оброблюваний вузол у LDFS алгоритмі |
| tmpDepth | Поточна глибина LDFS алгоритму |
| maxDepth | Максимальна глибина LDFS алгоритму |
| bfs | Вектор який представляє собою чергу для BFS алгоритму |
| visited | Вектор що містить всі оброблені вузли BFS алгоритму |
| closed | Вектор що містить всі оброблені вузли LDFS алгоритму |
| solution | Вектор, що містить шлях до вирішення задачі |

## 3.1 Загальний алгоритм

1. ПОЧАТОК
2. Зчитати максимальну глибину для LDFS алгоритму
3. Зчитати значення фішок з поля задачі:

3.1 Цикл проходу по всіх фішках:

(

3.1.1 ЯКЩО поточний елемент матриці немає напису – записати його як 0.

ІНАКШЕ записати зчитане число.

1. ЯКЩО обраний метод BFS, ТО обробити дані згідно алгоритму методу BFS (пункт 1.2)
2. ЯКЩО обраний метод LDFS, ТО
   1. Створити першу вершину для LDFS алгоритму(root)
   2. Створити вектор оброблених вузлів(visited)
   3. Обробити дані згідно алгоритму методу LDFS (пункт 1.3)
   4. Визначити шлях до вирішення задачі(sol)

5.5 Вивести інформацію про виконання алгоритму та записати її у файл

6. Вивести результат програми на фішки:

6.1 Цикл проходу по масиву - рішенню: (sol[0][i] – поточний елемент):

6.2ЯКЩО поточний елемент є нулем, ТО зробити текст фішки порожнім, ІНАКШЕ записати в текст фішки поточний елемент

1. КІНЕЦЬ

## 3.2 Алгоритм BFS

1. ПОЧАТОК
2. Створити початковий вузол(vert)
3. Створити чергу для запису вузлів(bfs) і вектор для перевірених вузлів(closed)
4. Додати у чергу першу вершину(vert)
5. Цикл проходження по всіх можливих вершинах якщо вони не є шуканими(pointTmp – поточний елемент):

5.1 Додати поточний вузол(pointTmp) у вектор перевірених вузлів(closed)

5.2 Зробити поточним вузлом(pointTmp) перший елемент черги(bfs)

5.3 Видалити перший елемент з черги(bfs)

5.4 Створити вектор можливих рухів нуля(mooves)

5.5 Цикл проходження по всіх можливих рухах нуля (vert – поточний елемент):

5.5.1 ЯКЩО поточний елемент(vert) не міститься у черзі(bfs) та векторі перевіерних вузлів(closed), ТО додати поточний елемент(vert) до черги. ІНАКШЕ видалити поточний елемент(vert)

6. Зробити вектор, що містить положення фішок на кожному кроці до вирішення(solution)

7. Вивести дані про роботу алгоритму у графічний інтерфейс

8. Видалити створені вершини з вектора(closed) і черги(bfs)

9. Записати результат роботи програми у файл

10. Повернути вектор що містить всі кроки до вирішення(solution)

11. КІНЕЦЬ

# Алгоритм LDFS

1.ПОЧАТОК

2. ЯКЩО поточна глибина(tmpDepth) >=максимальній глибині(maxDepth), ТО повернути поточний вузол(element)

3.ЯКЩО поточний елемент є шуканим, ТО повернути поточний вузол(element)

4. Додати у вектор оброблених вершин(visited) поточний елемент(element)

5. Створити вектор зі всіма можливами рухами нуля(mooves)

6. Цикл проходження по всіх можливих рухах нуля(поточний елемент - vert):

6.1 ЯКЩО поточний вузол(vert) є шуканим, ТО повернути поточний вузол(vert)

6.2 ЯКЩО поточний вузол(vert) міститься в векторі оброблених вузлів(visited), ТО видалити поточну вершину(vert).

ІНАКШЕ

6.2.1 Виконати алгоритм з пункту 1.3 збільшивши поточну глибину(tmpDepth) на 1

6.2.2 ЯКЩО поточний елемент(element) є шуканим, вернути поточний елемент(element)

7. Повернути поточний елемент(елемент)

8.КІНЕЦЬ

# **ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

## Діаграма класів програмного забезпечення



Рисунок 4.1 – Діаграма класів

## Опис методів частин програмного забезпечення

### Стандартні методи

У таблиці 4.1 наведені стандартні методи, що використані у програмному забезпеченні.

Таблиця 4.1 – Стандартніметоди

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 1 |  | to\_string | Перетворює val значення у рядок | Число, яке потрібно перетвори-ти у рядок | Строка типу string | string |
| 2 | string | c\_str | Перетворює строку у const char\* | - | Const char\* | string |
| 3 | vector | push\_back | Вставити елемент у кінець вектора | Елемент типу<Puzzle> | - | vector |
| 4 | vector | size | Визначити розмір вектора | - | Розмір вектора | vector |
| 5 | vector | erase | Видаляє з вектора один або декілька елементів | Ітератор first, що є позицією першого елементу для видалення | - | vector |
| 6 | vector | begin | Повертає ітератор що вказує на перший елемент вектора | - | Ітератор на перший елемент | vector |
| 7 |  | stoi | Перетворю строку на ціле число | Строка яку необхідно перетворити | Ціле число | string |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 8 | string | ToString | Перетворює значення val у строку | Значення val яке необхідно перетворити на строку | Строка string | string |
| 9 |  | Clock | Повертає час процесора витрачений програмою | - | Число секунд що пройшли з моменту початку виконання програми | ctime |
| 10 |  | time | Повертає календарний час | NULL | Seconds - об’єкт time\_t | ctime |
| 11 |  | localtime | Заповнює структуру time\_t поточною датою та часом | Seconds - Покажчик на time\_t | Timeinfo - tm структура в якій записані поточна дата і час | ctime |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 12 |  | strcpy | Копіює сі строку у масив символів | data – масив символів в який копіюється, TimeDate – C строка яка копіюється | - | string |
| 13 | MessageBox | Show | Виводить повідомлення | Строка string^ що є повідомленням, строка string^ що є назвою повідомлення | - |  |
| 14 | ofstream | is\_open | Перевіряє чи зв’язаний потік з файлом | - | Чи зв’язаний потік з файлом | fstream |
| 15 | ofstream | write | Вставляє перші n символів масиву на який вказує s у потік | result-Покажчик на масив символів, sizeof(Info) – кількість символів які треба вставити | - | fstream |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 16 |  | sizeof | Визначає розмір об’єкту або типу | Info – назва класу | Розмір типу size\_t |  |
| 17 | istream | read | Читає перші n символів з потоку у масив символів на який вказує s | result-Покажчик на масив символів, sizeof(Info) – кількість символів які треба прочитати | - | fstream |
| 18 | oistream | close | Закриває файл пов’язаний з об’єктом відключаючи його від потоку | - | - | fstream |
| 19 | istream | seekg | Встановлює файловий покажчик на певну позицію у файлі | 0 – позиція на яку ставиться покажчик, ios::end – кінець потоку | - | fstream |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 20 | istream | tellg | Повертає поточну позицію файлового покажчика | - | file\_s -поточна позиціа файлового покажчика | fstream |
| 21 | string | substr | Створює нову строку копіюванням частини старої | Pos – позиція звідки починається обрізання, len - уількість символів для обрізання | Date – строка що є підстрокою вхідної строки | string |
| 22 | TextBox | select | Переміщює курсор | i – довжина строки | - |  |
| 23 |  | isdigit | Визначає чи є символ цифрою | Line[i] - символ | Чи є символ цифрою | ctype.h |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 24 |  | atoi | Розбиває рядок Сі інтерпретуючи його вміст як ціле число яке повертається як значення типу int | curText - символ | Ціле число int | stdio.h |
| 25 | TextBox | TextBox | Створює TextBox | - | - |  |
| 26 | RadioButton | RadioButton | Створює RadioButton | - | - |  |
| 27 | Font | Font | Задати новий шрифт | String^ - назва шрифта, float – розмір шрифта, Font Style – стиль щрифта, unsigned char – кодування шрифта | - |  |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 28 | Button | Button | Створює Button | Немає | - |  |
| 29 | Size | Size | Створює новий об’єкт класу Size | Int width, int height – висота і ширина | - |  |
| 30 | Point | Point | Створює новий об’єкт класу Point | Int x, int y – координати для розміщення | - |  |
| 31 |  | EventHundler | Відображає метод що обробляє подію | Objec^ sender – джерело події; EventArgs^ e – об'єкт, який не має даних події | - |  |
| 32 | ControlCollection | Add | Додає елемент у колекцію | Control^ value – елемент управління | - |  |
| 33 | Label | Label | Створює Label | Немає | - |  |

Продовження таблиці 4.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 34 | FlowLayoutPanel | FlowLayoutPanel | Сворює FlowLayoutPanel | Немає | - |  |
| 35 | TableLayourPanel | TableLayourPanel | Створює TableLayourPanel | Немає | - |  |
| 36 |  | marshal\_as | Перетворює System::String в std::String | val – об’єкт System::String | Val – об’єкт std::string | Msclr/marshal\_cppstd.h |

### Користувацькі методи

У таблиці 4.2 наведені користувацькі методи, використані у програмному забезпеченні.

Таблиця 4.2 – Користувацькі методи

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 1 | Puzzle | Puzzle | Конструктор класу | ar – масив що выдображаэ положення фышок у 8-puzzle, ancictor -–покажчик на предка. | - | Puzzle.h |
| 2 | Puzzle | ~Puzzle | Деструктор – видаляє масив | - | - | Puzzle.h |
| 3 | Puzzle | mooves | Визначити куди може рухатися кнопка з написом 0 | - | moov - вектор об’єктів класу Puzzle, що зберігає всі можливі рухи поточної кнопки 0. | Puzzle.h |
| 4 | Puzzle | mooveRight | Визначити чи може 0 рухатися направо | - | Vertex – об’єкт класу Puzzle, з пересунотим нулем вправо | Puzzle.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 5 | Puzzle | mooveLeft | Визначити чи може 0 рухатися наліво | - | Vertex – об’єкт класу Puzzle,з пересунотим нулем вліво | Puzzle.h |
| 6 | Puzzle | mooveUp | Визначити чи може 0 рухатися вверх | - | Vertex – об’єкт класу Puzzle,з пересунотим нулем вверх | Puzzle.h |
| 7 | Puzzle | mooveDown | Визначити чи може 0 рухатися вниз | - | Vertex – об’єкт кла-су Puzzle,з пересунотим нулем вниз | Puzzle.h |
| 8 | Puzzle | getArr | Гетер, що повертає масив з розташуванням фішок у 8-puzzle | - | arr - масив що відоб- -ражає пол- -оження фішок у 8-puzzle | Puzzle.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 9 | Puzzle | getZeropos | Гетер, що повертає позицію нуля у масиві arr | - | posZero – позиція 0 у масиві arr | Puzzle.h |
| 10 | Puzzle | getAnc | Повертає покажчик на предка | - | anc – покажчик на предка | Puzzle.h |
| 11 | Info |  | Конструктор класу | steps – кількість кроків до вирішення, WorkTime – час роботи алгоритму, algorithm – назва алгоритму, startArr – початкове полодення фішок 8-puzzle, finishArr – фінальні позиції фішок у 8-puzzle, Date – дата коли алгоритм закінчив роботу, Depth – максимальна глибина для LDFS алгоритму | - | Info.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 12 | Info | getWorkTime | Гетер, що повертає час виконання алгоритму | - | time – час роботи алгоритму | Info.h |
| 13 | Info | Info | Пустий конструктор | - | - | Info.h |
| 14 | Info | getName | Гетер, що повертає назву алгоритму | - | name – назва алгоритму | Info.h |
| 15 | Info | getSteps | Гетер, що повертає кільість кроків до розв’язання | - | steps\_num – кількість кроків до розв’язання | Info.h |
| 16 | Info | getArrStart | Гетер, що повертає початкові позиції фішок у 8-puzzle | - | arrStart – масив що зберігає початкові позиції фішок у 8-puzzle | Info.h |
| 17 | Info | getArrFinish | Гетер, що повертає кінцеві позиції фішок у 8-puzzle | - | arrFinish – масив що зберігає кінцеві позиції фішок у 8-puzzle | Info.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 18 | Info | getDate | Гетер, що повертає дату виконання алгоритму | - | date – дата виконання алгоритму | Info.h |
| 19 | Info | getDepth | Гетер, що повертає максимальну глибину пошуку | - | depth максимальна глибина пошуку - | Info.h |
| 20 |  | mooveButton | Пересунути поточну кнопку на сусідню кнопку з 0 якщо така є | curButt – поточна кнопка, на яку було натиснуто, arr – масив кнопок з поля 8-puzzle | - | Header.h |
| 21 |  | BFS | Вирішення задачі 8-puzzle за допомогою BFS алгоритму | ar – масив поточних місцеположень фішок 8-puzzle, text -–текстове поле для виводу результатів алгоритму | solution – вектор, що зберігає позиції фішок 8-puzzle під час кожного кроку до вирішення | Header.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 22 |  | getCurDate | Отримання поточної дати та часу | - | data – поточна дата та час | Header.h |
| 23 |  | addZero | Додає 0 до дати або часу за необхідності | Data – дата або час виражені у годинах/хвилинах/днях/місяцях | Date – зміненений за необхідності час або дата | Header.h |
| 24 |  | chack | Перевіряє чи відповідаю поточне положення фішок 8-puzzle стандартному | arr – масив поточних положень фішок | checked – чи відповідає поточний масив положень фішок стандартному | Header.h |
| 25 |  | Contains | Перевіряє чи був поточний об’єкт класу Puzzle колись вже оброблений | list – вектор що зберігає покажчики на об’єкти класу Puzzle, element – елемент який шукається у list | isEquel – чи є вказаний елемент у списку перевірених | Header.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 26 |  | removeData | Видаляє вже непотрібні об’єкти класу Puzzle | del – вектор покажчиків на об’єкти класу Puzzle | - | Header.h |
| 27 |  | SolutionPath | Формує вектор масивів, що зберігають положення фішок на кожному кроці до вирішення використовуючи предків елемента-вирішення | solution –покажчик на об’єкт класу Puzzle, що є вирішенням | Path – вектор масивів що зберігають положення фішок на кожному кроці до вирішення | Header.h |
| 28 |  | Display | Виводить інформацію про вирішену задачку у текстове поле | solutionArr – вектор масивів що зберігають положення фішок на кожному кроці до вирішення, text – текстове поле у яке виводиться інфомація, name – назва алгоритму, sec – час роботи алгоритму у секундах | - | Header.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 29 |  | DLS | Вирішує поточну задачу 8-puzzle за допомогою DLS алгоритму | element – перший об’єкт класу Puzzle який зберігає поточне положення вішок, visited – вектор об’єктів класу Puzzle, що вже перевірядися, tmpDepth – поточна глибина на якій відбувається перевірка, maxDepth – максимальна глибина для алгоритму | element – об’єкт класу Puzzle, що є вирішенням задачі | Header.h |
| 30 |  | WriteToFile | Пише інформацію про результати роботи програми у файл | result – об’єкт класу Info, що зберігає інформацію про роботу алгоритма | - | Header.h |
| 31 |  | ReadFile | Читає інформацію з файла і виводить у текстове поле | text – текстове поле, у яке виводиться інформація | - | Header.h |
| 32 | Form1 | createArr | Створює масив кнопок, що є фішками у 8-puzzle | - | arr – масив кнопок | Form1.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 33 | Form1 | createTextArr | Створює масив текстових полів для вводу значень фішок 8-puzzle | - | arr1 – масив текстових полів | Form1.h |
| 34 | Form1 | Apply\_Click | Записати значення з текстових полів у фішки 8-puzzle | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 35 | Form1 | Start\_Click | Вирішує задачу 8-puzzle алгоритмом який вибрав користувач | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 36 | Form1 | Clear\_Click | Повертає фішки у початкові позиції | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 37 | Form1 | textBox11\_TextChanged | Перевіряє на правильність введені символи | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 38 | Form1 | Form1\_Load | Встановлює текст текстового поля 11 “10” | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 39 | Form1 | Read\_Click | Читає дані з файла і записує у текстове поле | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 40 | Form1 | ClearFile\_Click | Очищує файл з результатами | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 41 | Form1 | button1\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 42 | Form1 | Button2\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 43 | Form1 | Button3\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 44 | Form1 | Button4\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 45 | Form1 | Button5\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 46 | Form1 | Button6\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 47 | Form1 | Button7\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |

Продовження таблиці 4.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | Назва классу | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 48 | Form1 | Button8\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |
| 49 | Form1 | Button9\_Click | Переміщує кнопку на позицію кнопки з нулем якщо воно є поруч | sender – посилання на об'єкт, що викли-кав подію; e – об'єкт, який від-носиться до оброб-люваної події | - | Form1.h |

# **ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

### 5.1 План тестування

План тестування програмного забезпечення для перевірки роботи програми при некоректно введених даних, для перевірки роботи методів BFS і LDFS, а також запису та читання результатів роботи програми з файлу.

1. Тестування правильності введених значень.
   1. Тестування при введенні некоректних або повторюваних символів.
2. Тестування коректності роботи методів BFS, LDFS.
   1. Перевірка коректності роботи методу BFS.
   2. Перевірка коректності роботи методу LDFS.
   3. Перевірка коректності роботи методу LDFS з недостатньою глибиною пошуку.

в) Тестування запису результатів у файл

1) Тестування запису результатів роботи BFS або LDFS методу у файл

г) Тестування читання результатів з файлу

1) Тестування читання результатів роботи алгоритмів BFS і LDFS з файлу

2) Тестування читання результатів роботи алгоритмів BFS і LDFS з файлу у випадку, коли файл порожній

д) Тестування очищення файлу результатів

1) Тестування правильності очищення файлу результатів

# 5.2 Приклади тестування

Тестування роботи програми за наведеним вище планом перевірки. Результати зображені у таблицях і рисунках.

Таблиця 5.1 – Робота програми при некоректному введені даних для місцеположення фішок

|  |  |
| --- | --- |
| Мета тесту | Перевірити роботу програми при некоректному введенні місцеположення фішок |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | У текстові поля введені некоректні символи (Наприклад: 1, 6, 9, h, 0, 3, y ,9 , -) |
| Схема проведення тесту | Вводимо місцеположення фішок, деякі з символів вводим некоректні |
| Очікуваний результат | Повідомлення про помилку при записування місцеположення фішок на поле |
| Стан програми після проведення випробувань | Видано помилку «Invalid Values!» (рис. 5.1) |

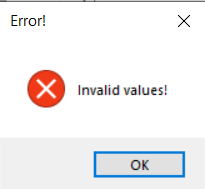


Рисунок 5.1 – Повідомлення про некоректному введені місцеположення вішок

Таблиця 5.2 – Приклад роботи програми при вибраному методі BFS

|  |  |
| --- | --- |
| Мета тесту | Перевірити роботу методу BFS для вирішення задачі 8-puzzle |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Місцеположення 9 фішок на полі(1, 6, 2, 4, 0 3, 7, 5, 8), обраний метод BFS |
| Схема проведення тесту | Обираємо метод для вирішення задачі і початкове місцеположення фішок на полі |
| Очікуваний результат | Вивід результату роботи алгоритму у текстове поле |
| Стан програми після проведення випробувань | Результат роботи алгоритму виведено у текстове поле (рис. 5.2) |

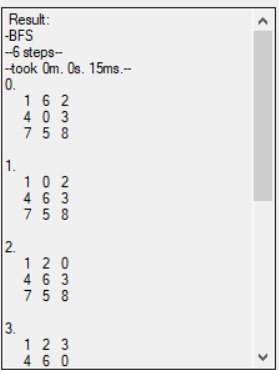


Рисунок 5.2 – Приклад виведення результату про роботу алгоритму BFS

Таблиця 5.3 – Приклад роботи програми при вибраному методі LDFS

|  |  |
| --- | --- |
| Мета тесту | Перевірити роботу методу LDFS для вирішення задачі 8-puzzle |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Місцеположення 9 фішок на полі(1, 6, 2, 4, 0 3, 7, 5, 8), обраний метод LDFS, максимальна глибина пошуку - 10 |
| Схема проведення тесту | Обираємо метод для вирішення задачі , початкове місцеположення фішок на полі і вводимо максимальну глибину пошуку |
| Очікуваний результат | Вивід результату роботи алгоритму у текстове поле |
| Стан програми після проведення випробувань | Результат роботи алгоритму виведено у текстове поле (рис. 5.3) |

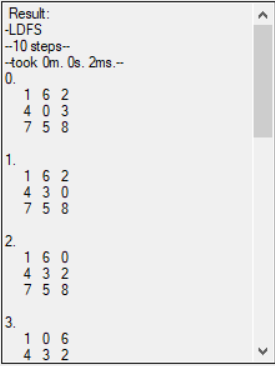


Рисунок 5.3 - Приклад виведення результату про роботу алгоритму LDFS

Таблиця 5.4 – Приклад роботи програми при вибраному методі LDFS і недостатній максимальній глибині пошуку

|  |  |
| --- | --- |
| Мета тесту | Перевірити роботу методу LDFS для вирішення задачі 8-puzzle з замаленькою максимальною шлибиною пошуку |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Місцеположення 9 фішок на полі(1, 6, 2, 4, 0 3, 7, 5, 8), обраний метод LDFS, максимальна глибина пошуку - 2 |
| Схема проведення тесту | Обираємо метод для вирішення задачі , початкове місцеположення фішок на полі і вводимо максимальну глибину пошуку |
| Очікуваний результат | Повідомлення про помилку при спробі знайти рішення на маленькій глибині і виведено інформацію про роботу алгоритму у текстове поле |
| Стан програми після проведення випробувань | Видано помилку “Too small depth!” (рис. 5.4) і записано результати роботи програми у текстове поле (рис. 5.5) |

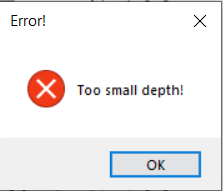


Рисунок 5.4 – Помилка при спробі знайти рішення на малій глибині

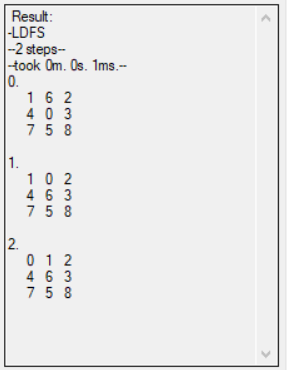


Рисунок 5.5 – Приклад роботи LDFS алгоритму при малій глибині пошуку

Таблиця 5.5 – Запис результатів роботи алгоритму у файл

|  |  |
| --- | --- |
| Мета тесту | Перевірити запис результатів у файл |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Вид алгоритму, яким була вирішена задачка, час роботи алгоритму, кількість кроків до вирішення, початкове та кінцеве положення фішок, поточна дата |
| Схема проведення тесту | Обираємо метод розв’язування, правильно вводимо початкове місцеположення фішок і глибину за необхідності |
| Очікуваний результат | Результат записується у бінарний файл |
| Стан програми після проведення випробувань | Результат записується у файл (рис. 5.6) |

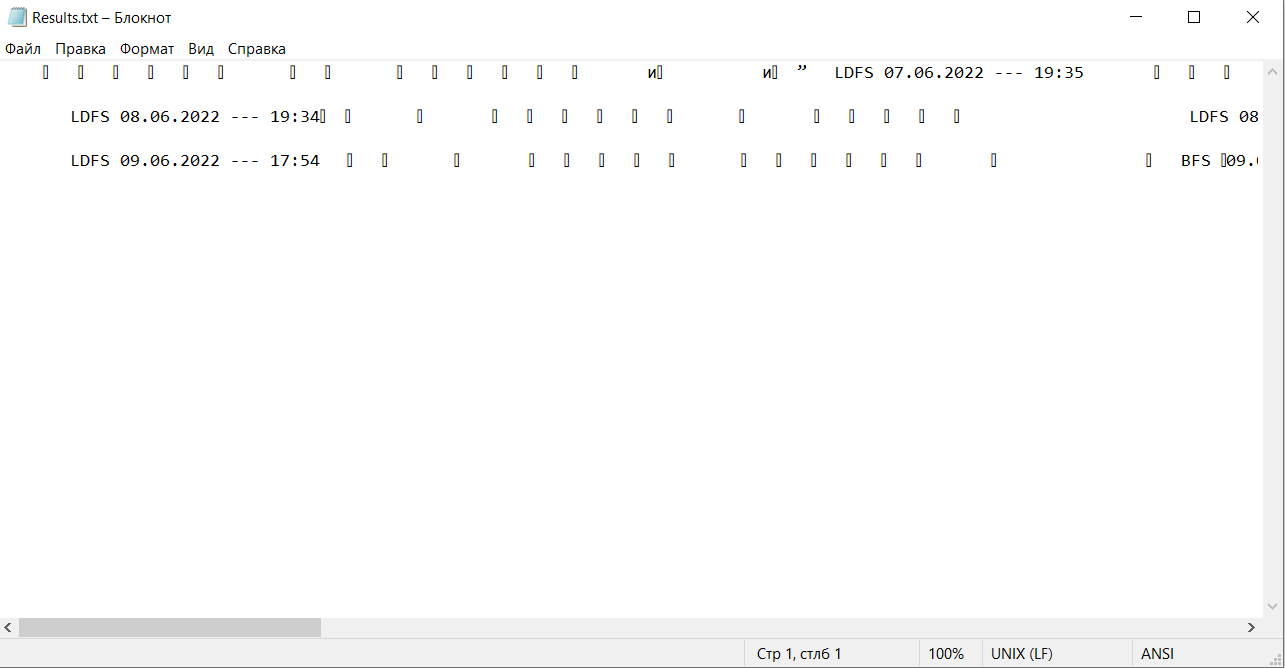


Рисунок 5.6 – Результат записано у файл Results.txt

Таблиця 5.6 – Читання результатів роботи алгоритмів з файлу

|  |  |
| --- | --- |
| Мета тесту | Перевірити запис результатів у файл |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Назва файлу – Results.txt |
| Схема проведення тесту | Натискаємо кнопку Read File |
| Очікуваний результат | Результат записується у текстове поле |
| Стан програми після проведення випробувань | Результат записується у текстове поле (рис. 5.7) |

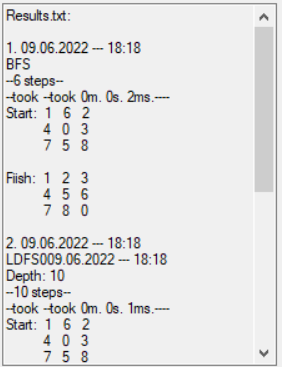


Рисунок 2.6 – Читання результатів з файлу

Таблиця 5.7 - Читання результатів роботи алгоритмів з файлу, якщо файл порожній

|  |  |
| --- | --- |
| Мета тесту | Перевірити читання результатів з файлу |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Назва файлу – Results.txt(Порожній) |
| Схема проведення тесту | Натискаємо кнопку Read File |
| Очікуваний результат | Повідомлення про помилку читання з файлу |
| Стан програми після проведення випробувань | Видано помилку “File is empty!” (рис. 5.8) |

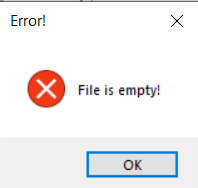


Рисунок 5.8 – Повідомлення про те, що файл порожній

Таблиця 5.7 – Очищення файлу результатів

|  |  |
| --- | --- |
| Мета тесту | Перевірити очищення файлу результатів |
| Початковий стан програми | Відкрите вікно програми |
| Вхідні дані | Назва файлу – Results.txt |
| Схема проведення тесту | Натискаємо кнопку Clear File |
| Очікуваний результат | Повідомлення про те, що файл очищено |
| Стан програми після проведення випробувань | Видано повідомлення “File is empty now!” (рис. 5.9) |

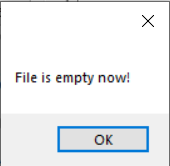


Рисунок 5.9 – Повідомлення про те, що файл очищено

# **Інструкція користувача**

## Робота з програмою

Після запуску виконавчого файлу з розширенням \*.exe, відкривається головне вікно програми (Рисунок 6.1).

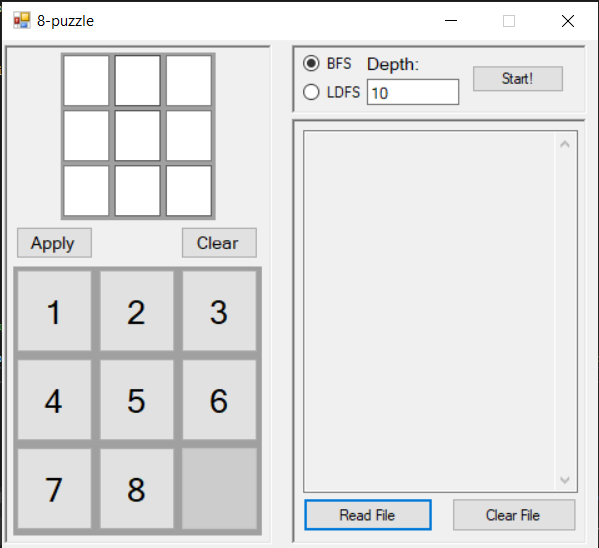


Рисунок 6.1 – Головне вікно програми

Далі за допомогою натискання на кнопки(1 - 9) або введення цифр з клавіатури у текстові поля на руснку 6.2 потрібно задати початкове положення фішок.

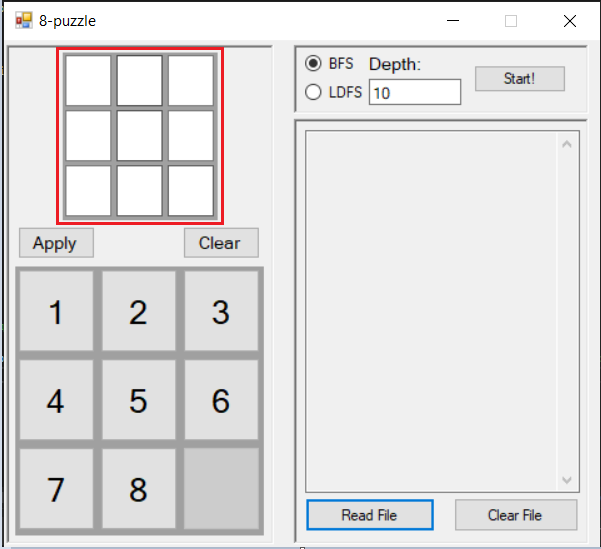


Рисунок 6.2

Далі вибрати мето розв’язання та встановити глибину якщо вибраний LDFS алгоритм і натиснути кнопку Start для початку роботи алгоритму(Рисунок 6.3)

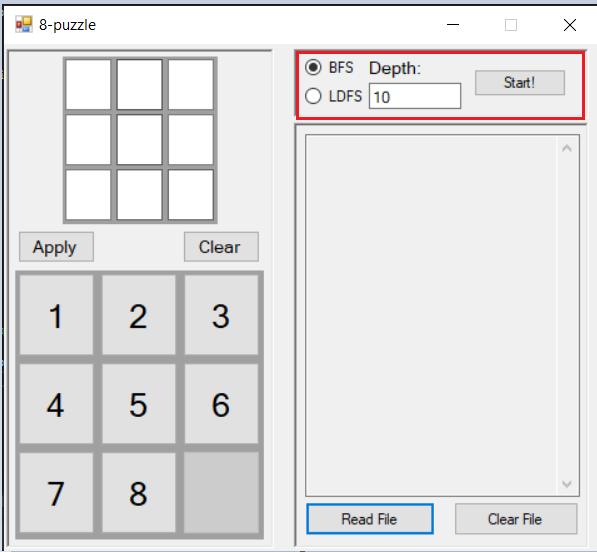


Рисунок 6.3

Результати роботи програми будуть виведені у текстове поле(Рисунок 6.4)

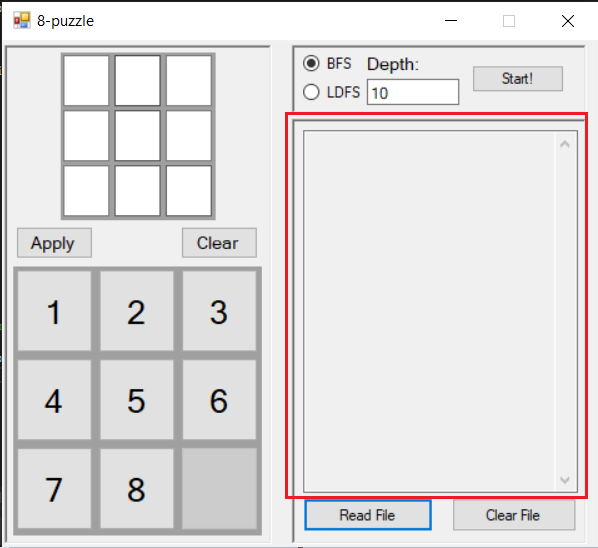


Рисунок 6.4

За допомогою кнопок Read File і Clear File(Рисунок 6.5) можна вивести вміст файла у текстове поле(Рисунок 6.5) і очистити файл відповідно.

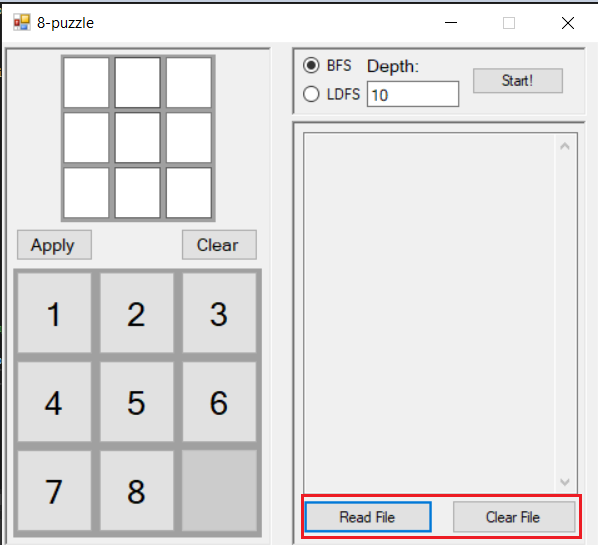


Рисунок 6.5

## Формат вхідних та вихідних даних

Користувачем на вхід програми подається місцеположення фішок на полі 3x3, а також, за необхідності, максимальна глибина пошуку для LDFS алгоритму.

Результатом виконання програми є розв’язок задачі 8-puzzle, а також всі кроки до вирішення і час роботи програми, які видаються у текстове.

## Системні вимоги

Системні вимоги до програмного забезпечення наведені в таблиці 6.1.

Таблиця 6.1 – Системні вимоги програмного забезпечення

|  |  |  |
| --- | --- | --- |
|  | Мінімальні | Рекомендовані |
| Операційна система | Windows® XP/Windows Vista/Windows 7/ Windows 8/Windows 10/ Windows 11 (з останніми обновленнями) | Windows 7/ Windows 8/Windows 10/Windows11  (з останніми обновленнями) |
| Процесор | Intel® Pentium® ІІІ  1.0 GHz або  AMD Athlon™ 1.0 GHz | Intel® Pentium® D або AMD Athlon™ 64 X2 |
| Оперативна пам'ять | 256 MB RAM (для Windows® XP) / 1 GB RAM (для Windows Vista/Windows 7/  Windows 8/Windows 10)/4 GB RAM (для Windows 11) | 4 GB RAM |
| Відеоадаптер | Intel GMA 950 з відеопам'яттю об'ємом не менше 64 МБ (або сумісний аналог) | |
| Дисплей | 800х600 | 1024х768 або краще |

Продовження таблиці 6.1

|  |  |  |
| --- | --- | --- |
|  | Мінімальні | Рекомендовані |
| Прилади введення | Клавіатура, комп’ютерна миша | |
| Додаткове програмне забезпечення | Microsoft .Net Framework 4.7.2 або вище | |

# **Аналіз і узагальнення результатів**

Головною задачею курсової роботи була реалізація програми для розв’язання 8-puzzle наступними методами: BFS, LDFS.

Критичні ситуації у роботі програми виявлені не були. Під час тестування було виявлено, що програма може працювати задовго у випадку, коли користувач задає початкову розстановку фішок таку, що в результаті формується завелике дерево. Всі дані, які вводить користувач з клавіатури, провіряються на правильність і лише потім подаються на обробку програмі.

Для перевірки та доведення достовірності результатів виконання програмного забезпечення достатньо підтвердити правильність розташування фішок на полі у графічному інтерфейсі:

а) Метод BFS.

Результат виконання методу BFS та початкове місцеположення фішок наведено на рисунку 7.1 та 7.2 відповідно:

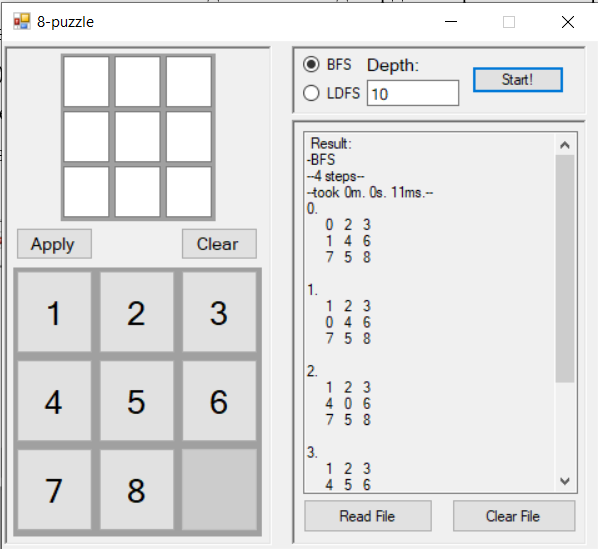


Рисунок 7.1 – Результат роботи методу BFS

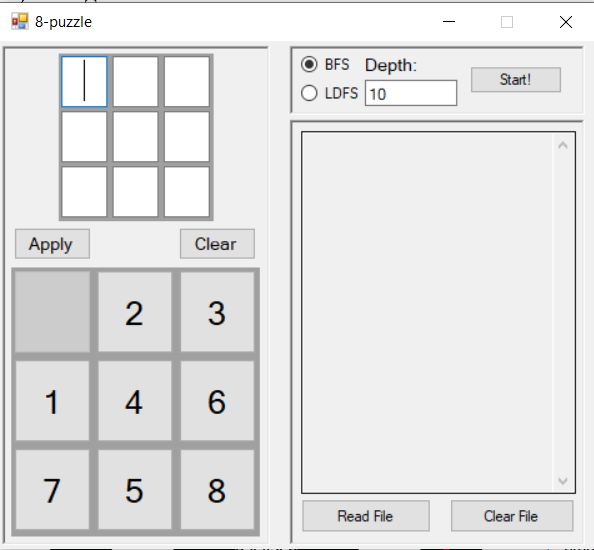


Рисунок 7.2 – Початкове місцеположення фішок

Рисунок 7.1 – Результат виконання методу Якобі

Оскільки кінцеве положення фішок на полі є правильним (Рисунок 7.1) метод працює коректно. Також результат можна перевірити за допомогою сайту <https://deniz.co/8-puzzle-solver/> (Рисунок 7.3).

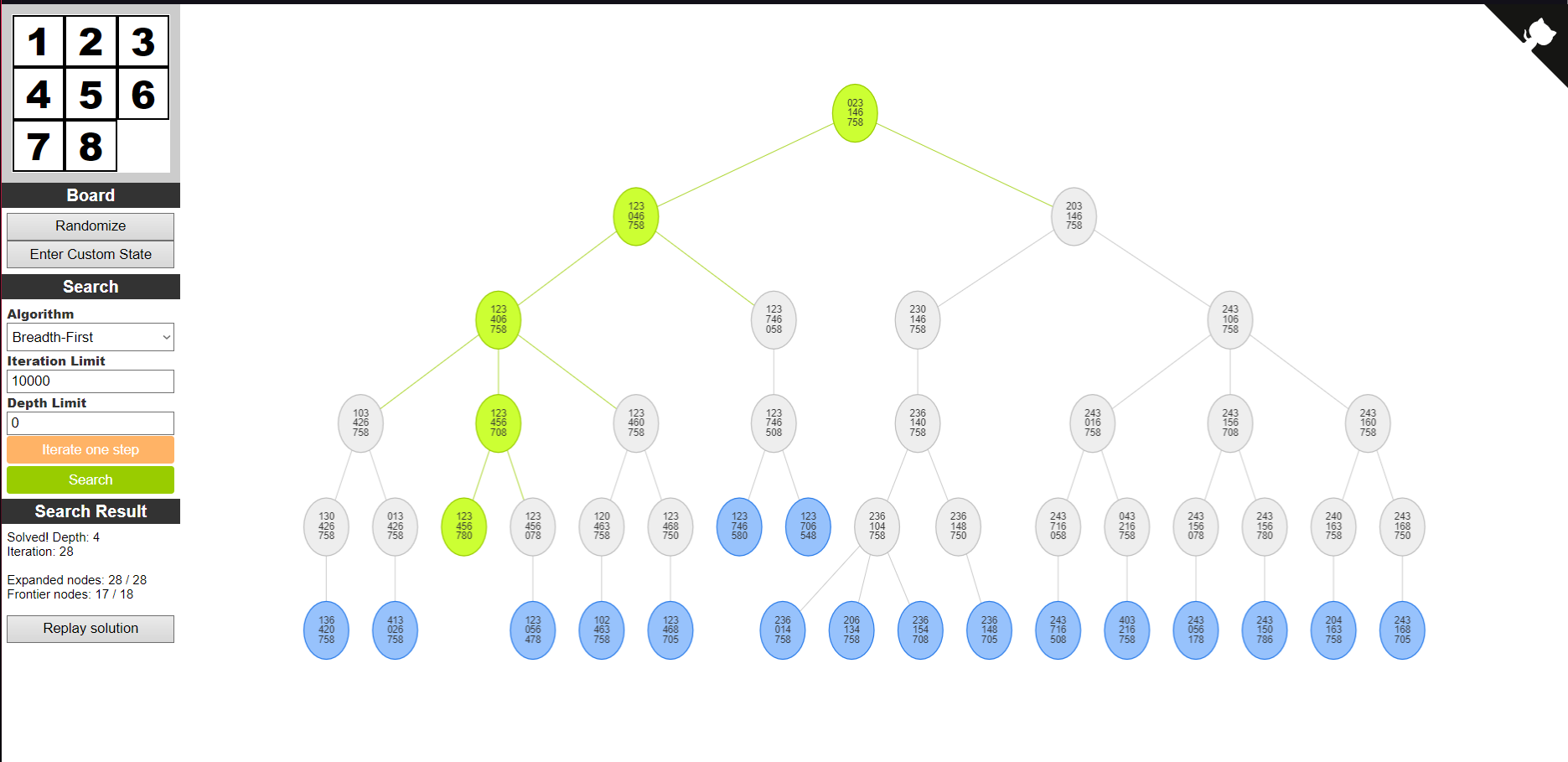


Рисунок 7.3- Результат роботи алгоритму BFS із вказаного сайту

б) Метод LDFS

Результат виконання методу LDFS та початкове місцеположення фішок наведено на рисунку 7.4 та 7.5 відповідно:

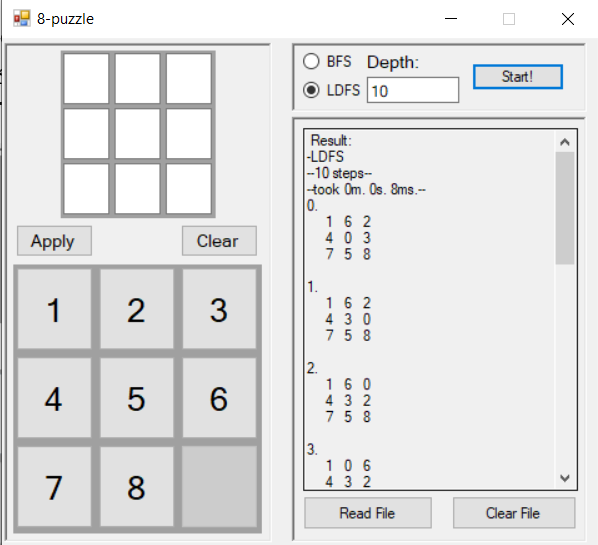


Рисунок 7.4 – Результат роботи методі LDFS

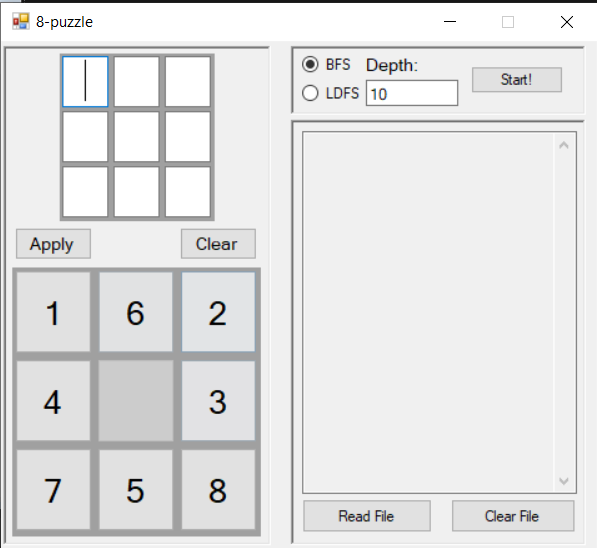


Рисунок 7.5 – Початкове місцеположення фішок

Оскільки кінцеве положення фішок на полі є правильним (Рисунок 7.4) метод працює коректно. Також результат можна перевірити за допомогою сайту <https://deniz.co/8-puzzle-solver/> (Рисунок 7.6).

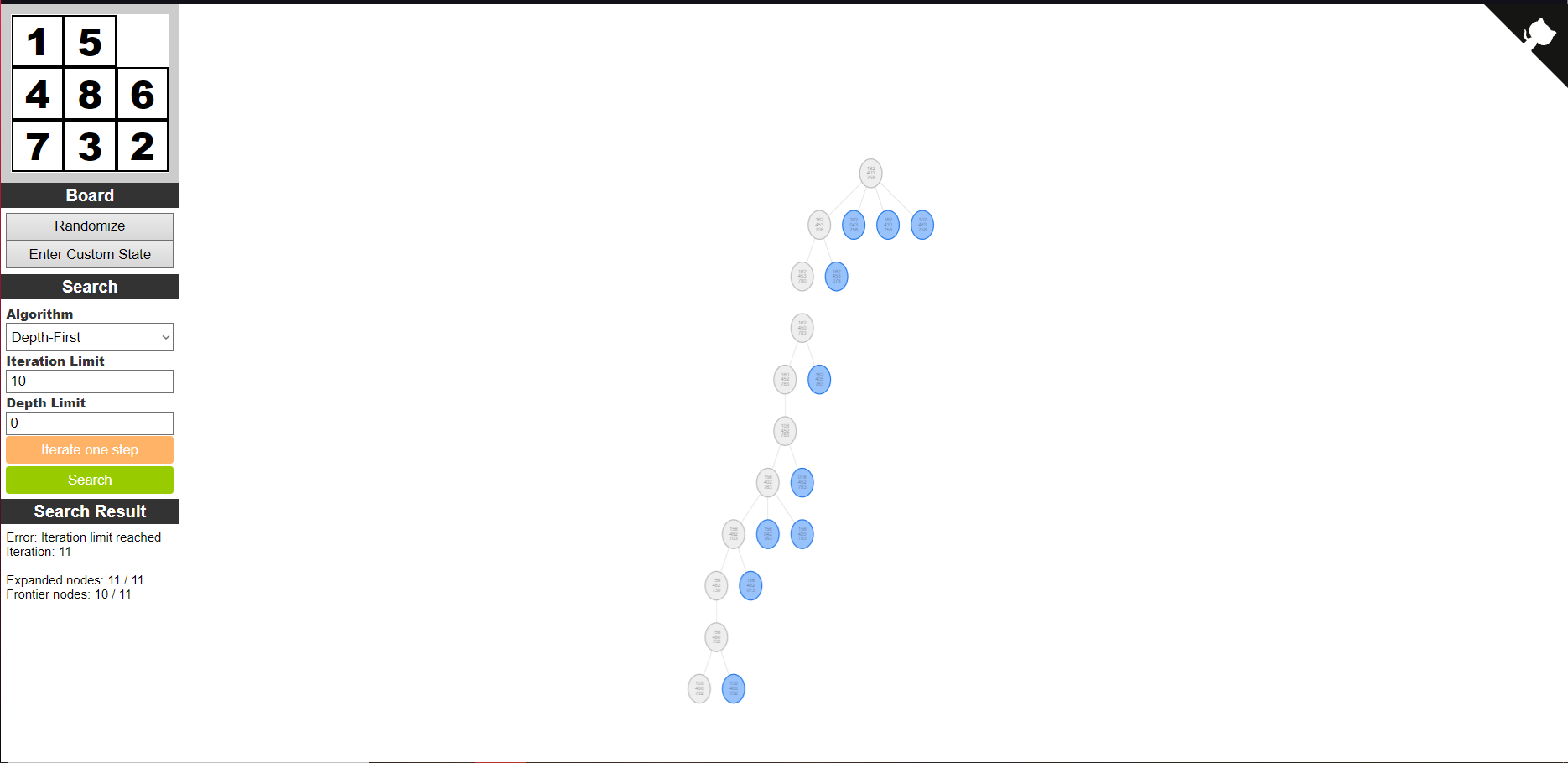


Рисунок 7.6 - Результат роботи алгоритму LDFS із вказаного сайту

Для проведення тестування ефективності програми перевіримо загальну кількість створених вузлів для обох алгоритмів при однаковому початковому положенні фішок, час роботи алгоритмів та кількість кроків до вирішення задачі при різному початковому положенні фішок на полі.

Таблиця 7.1 – Тестування ефективності методів

|  |  |  |  |
| --- | --- | --- | --- |
| Позиції фішок | Параметри тестування | Метод | |
| LDFS | BFS |
|  | Кількість створених вузлів | 31(11 глибина) | 534 |
| Час роботи алгоритма | 10ms | 26ms |
| Кількість кроків | 9 | 9 |
|  | Кількість створених вузлів | 28(11 глибина) | 1266 |
| Час роботи алгоритма | 64ms | 66ms |
| Кількість кроків | 10 | 10 |
|  | Кількість створених вузлів | 46(12 глибина) | 2351 |
| Час роботи алгоритма | 89ms | 207ms |
| Кількість кроків | 12 | 12 |
|  | Кількість створених вузлів | 66(20 глибина) | 4207 |
| Час роботи алгоритма | 1s 475ms | 641ms |
| Кількість кроків | 17 | 13 |
|  | Кількість створених вузлів | 47(20 глибина) | 13090 |
| Час роботи алгоритма | 336ms | 6s 102ms |
| Кількість кроків | 18 | 16 |

Візуалізацію результатів таблиці 7.1 наведено на рисунках 7.7 та 7.8:

Рисунок 7.7 – Діаграма залежності кількості кроків від обраного методу і початкової розстановки фішок

Рисунок 7.8 – Діаграма залежності кількості одночасно існуючих вузлів від обраного методі ті початкових позицій фішок

За результатами тестування можна зробити такі висновки:

а) Всі розглянуті методи дозволяють розв’язати задачу 8-puzzle.

б) З розглянутих методів найоптимальнішим для практичного використання є метод LDFS, оскільки він виконується найшвидше та використовує менше всього вершин дерева.

в) З розглянутих методів метод BFS знаходить найоптимальніший шлях до вирішення задачі, тобто такий, що займає меншу кількість кроків, але одночасно існує велика кількість вузлів.

# **ВИСНОВКИ**

Для написання даної курсової роботи було використано навички набуті під час проходження курсу з Основ Програмування, а також були набуті нові навички розробки графічного інтерфейсу і досліджені алгоритми (BDS та LDFS), за допомогою яких було вирішено поставлену задачу.

Було створено програмне забезпечення засобами мови C++ та Windows Forms для вирішення задачі.

Було зроблено відповідну документацію та записано проміжні результати під час розробки програмного забезпечення.

Було проведено тестування програмного забезпечення з використанням різних методів і порівняно результати виконання. На основі результатів було визначено, що метод LDFS працює швидше за BFS.

Програмне забезпечення виконує поставлені задачі, отже розробку можна вважати успішною.

# **ПЕРЕЛІК ПОСИЛАНЬ**

1. Алгоритму пошуку. URL: <https://er.nau.edu.ua/bitstream/NAU/40676/1/Методи%20та%20системи%20штучного%20інтелекту%20_Навч_посібн.pdf> (дата звернення 25.05.2022)
2. LDFS алгоритм. URL <https://www.javatpoint.com/depth-first-search-algorithm> (дата звернення 27.05.2022)
3. BFS алгоритм. URL <https://en.wikipedia.org/wiki/Breadth-first_search>

( дата звернення 27.05.2022)

Додаток А Технічне завдання

КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

Кафедра

інформатики та програмної інженерії

Затвердив

Керівник: Головченко Максим Миколайович

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_ р.

Виконавець:

Студент Тітов Роман Едуардович

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_ р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання курсової роботи

на тему: 8-puzzle

з дисципліни:

«Основи програмування»

Київ 2022

* 1. *Мета*: Метою курсової роботи є розробка програмного забезпечення
  2. *Дата початку роботи*: «13»травня 2022 р.
  3. *Дата закінчення роботи*: «12»червня 2022 р.
  4. *Вимоги до програмного забезпечення*.

1. Функціональні вимоги:

* Можливість ручного розміщення восьми фішок на полі розміром 3х3.
* Можливість обирати метод розв’язання.
* Можливість розв’язання 8-puzzle за допомогою обраного метода.
* Можливість збереження результатів у файл.
* Можливість читання результатів з файла
* Можливість виведення розв’язку задачі у графічний інтерфейс.

1. Нефункціональні вимоги:

* Підтримка ОС Windows 10 і вище.
* Все програмне забезпечення та супроводжуюча технічна документація повинні задовольняти наступним ДЕСТам:

ГОСТ 29.401 - 78 - Текст програми. Вимоги до змісту та оформлення.

ГОСТ 19.106 - 78 - Вимоги до програмної документації.

ГОСТ 7.1 - 84 та ДСТУ 3008 - 2015 - Розробка технічної документації.

* 1. *Стадії та етапи розробки*:

1. Об'єктно-орієнтований аналіз предметної області задачі (до21.04.2022 р.)
2. Об'єктно-орієнтоване проектування архітектури програмної системи (до 02.05.2022р.)
3. Розробка програмного забезпечення (до 05.05.2022р.)
4. Тестування розробленої програми (до 07.06.2022р.)
5. Розробка пояснювальної записки (до 11.06.2022 р.).
6. Захист курсової роботи (до 18.06.2022 р.).
   1. *Порядок контролю та приймання*. Поточні результати роботи над КР регулярно демонструються викладачу. Своєчасність виконання основних етапів графіку підготовки роботи впливає на оцінку за КР відповідно до критеріїв оцінювання.

Додаток Б Тексти програмного коду

*студента групи ІП-14 І курсу*

*Тітова Р.Е.*

(Вид носія даних)

*Електронний носій*

(Обсяг програми (документа), арк., Кб)

*42 арк, 363 Кб*

(Найменування програми (документа))

*Тексти програмного коду програмного забезпечення*

*для вирішення задачі 8-puzzle*

# **Info.h**

#pragma once

#include <iostream>

class Info { // Класс з інформацією про результат роботи алгоритму

int arrStart[9]; // Початкове положення

int arrFinish[9]; // Фінальне положення

int steps\_num; // Кількість кроків

int sec; // Кількість секунд на алгоритм

int min; // Кількість хвилин на алгоритм

int depth; // Максимальна глибина пошуку для LDFS

int milisec; // Кількість мілісекунд на алгоритм

char name[5]; // Назва алгоритму

char date[20]; // Дата запису

public:

Info(int, int, const char\*, int\*, int\*, char\*, int);

Info(){}

std::string getWorkTime();

char\* getName() { return name; }

int getSteps() { return steps\_num; };

int\* getArrStart() { return arrStart; }

int\* getArrFinish() { return arrFinish; }

char\* getDate() { return date; }

int getDepth() { return depth; }

};

# Info.cpp

#include "pch.h"

#include "Info.h"

#include <string>

using namespace std;

Info::Info(int steps, int WorkTime, const char\* algorithm, int\* startArr, int\* finishArr, char\* Data, int Depth) {

steps\_num = steps;

depth = Depth;

for (int i = 0; i < 9; i++) {

arrStart[i] = startArr[i];

arrFinish[i] = finishArr[i];

}

for (int i = 0; i < 4; i++)

name[i] = algorithm[i];

min = WorkTime / 60000; // Переведення мілісекунд в хвилини

sec = (WorkTime % 60000) / 1000; // Переведення мілісекунд в секунди

milisec = WorkTime % 1000; // Кількість мілісекунд

for (int i = 0; i < 20; i++)

date[i] = Data[i];

}

std::string Info::getWorkTime() {

std::string time = "--took " + to\_string(min) + "m. " + to\_string(sec) + "s. " + to\_string(milisec) + "ms.--";

return time;

}

# Form1.h

#pragma once

#include "Header.h"

#include <iostream>

using namespace std;

namespace CppCLRWinFormsProject {

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>

/// Summary for Form1

/// </summary>

public ref class Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

arr = createArr(); // Заповнення масиву кнопок

arrText = createTextArr(); // Заповнення масиву текстових полів

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::TableLayoutPanel^ tableLayoutPanel1;

protected:

private: System::Windows::Forms::Button^ button9;

private: System::Windows::Forms::Button^ button8;

private: System::Windows::Forms::Button^ button5;

private: System::Windows::Forms::Button^ button4;

private: System::Windows::Forms::Button^ button3;

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

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

private: System::Windows::Forms::Button^ button6;

private: System::Windows::Forms::Button^ button7;

private: System::Windows::Forms::TableLayoutPanel^ tableLayoutPanel2;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::TextBox^ textBox9;

private: System::Windows::Forms::TextBox^ textBox8;

private: System::Windows::Forms::TextBox^ textBox7;

private: System::Windows::Forms::TextBox^ textBox6;

private: System::Windows::Forms::TextBox^ textBox5;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::TextBox^ textBox2;

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

private: System::Windows::Forms::FlowLayoutPanel^ flowLayoutPanel1;

private: cli::array<Button^, 1>^ arr; // Масив кнопок

private: cli::array<TextBox^, 1>^ arrText; // Масив текстових полів

private: System::Windows::Forms::RadioButton^ radioButton1;

private: System::Windows::Forms::RadioButton^ radioButton2;

private: System::Windows::Forms::FlowLayoutPanel^ flowLayoutPanel2;

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

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

private: System::Windows::Forms::TextBox^ textBox10;

private: System::Windows::Forms::FlowLayoutPanel^ flowLayoutPanel3;

private: System::Windows::Forms::TextBox^ textBox11;

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

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

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

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->tableLayoutPanel1 = (gcnew System::Windows::Forms::TableLayoutPanel());

this->button8 = (gcnew System::Windows::Forms::Button());

this->button5 = (gcnew System::Windows::Forms::Button());

this->button4 = (gcnew System::Windows::Forms::Button());

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

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

this->button6 = (gcnew System::Windows::Forms::Button());

this->button7 = (gcnew System::Windows::Forms::Button());

this->button9 = (gcnew System::Windows::Forms::Button());

this->button3 = (gcnew System::Windows::Forms::Button());

this->tableLayoutPanel2 = (gcnew System::Windows::Forms::TableLayoutPanel());

this->textBox9 = (gcnew System::Windows::Forms::TextBox());

this->textBox8 = (gcnew System::Windows::Forms::TextBox());

this->textBox7 = (gcnew System::Windows::Forms::TextBox());

this->textBox6 = (gcnew System::Windows::Forms::TextBox());

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

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

this->flowLayoutPanel1 = (gcnew System::Windows::Forms::FlowLayoutPanel());

this->radioButton1 = (gcnew System::Windows::Forms::RadioButton());

this->radioButton2 = (gcnew System::Windows::Forms::RadioButton());

this->flowLayoutPanel2 = (gcnew System::Windows::Forms::FlowLayoutPanel());

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

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

this->textBox10 = (gcnew System::Windows::Forms::TextBox());

this->flowLayoutPanel3 = (gcnew System::Windows::Forms::FlowLayoutPanel());

this->textBox11 = (gcnew System::Windows::Forms::TextBox());

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

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

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

this->tableLayoutPanel1->SuspendLayout();

this->tableLayoutPanel2->SuspendLayout();

this->SuspendLayout();

//

// tableLayoutPanel1

//

this->tableLayoutPanel1->BackColor = System::Drawing::SystemColors::ControlDark;

this->tableLayoutPanel1->ColumnCount = 3;

this->tableLayoutPanel1->ColumnStyles->Add((gcnew System::Windows::Forms::ColumnStyle(System::Windows::Forms::SizeType::Percent,

33.33333F)));

this->tableLayoutPanel1->ColumnStyles->Add((gcnew System::Windows::Forms::ColumnStyle(System::Windows::Forms::SizeType::Percent,

33.33333F)));

this->tableLayoutPanel1->ColumnStyles->Add((gcnew System::Windows::Forms::ColumnStyle(System::Windows::Forms::SizeType::Percent,

33.33333F)));

this->tableLayoutPanel1->Controls->Add(this->button8, 1, 2);

this->tableLayoutPanel1->Controls->Add(this->button5, 1, 1);

this->tableLayoutPanel1->Controls->Add(this->button4, 0, 1);

this->tableLayoutPanel1->Controls->Add(this->button2, 1, 0);

this->tableLayoutPanel1->Controls->Add(this->button1, 0, 0);

this->tableLayoutPanel1->Controls->Add(this->button6, 2, 1);

this->tableLayoutPanel1->Controls->Add(this->button7, 0, 2);

this->tableLayoutPanel1->Controls->Add(this->button9, 2, 2);

this->tableLayoutPanel1->Controls->Add(this->button3, 2, 0);

this->tableLayoutPanel1->Location = System::Drawing::Point(12, 223);

this->tableLayoutPanel1->Margin = System::Windows::Forms::Padding(0);

this->tableLayoutPanel1->Name = L"tableLayoutPanel1";

this->tableLayoutPanel1->RowCount = 3;

this->tableLayoutPanel1->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Percent, 50)));

this->tableLayoutPanel1->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Percent, 50)));

this->tableLayoutPanel1->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Absolute, 88)));

this->tableLayoutPanel1->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Absolute, 20)));

this->tableLayoutPanel1->Size = System::Drawing::Size(265, 265);

this->tableLayoutPanel1->TabIndex = 0;

//

// button8

//

this->button8->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button8->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button8->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button8->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button8->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button8->Location = System::Drawing::Point(91, 179);

this->button8->Name = L"button8";

this->button8->Size = System::Drawing::Size(82, 82);

this->button8->TabIndex = 8;

this->button8->TabStop = false;

this->button8->Tag = L"7";

this->button8->Text = L"8";

this->button8->UseVisualStyleBackColor = true;

this->button8->Click += gcnew System::EventHandler(this, &Form1::button8\_Click);

//

// button5

//

this->button5->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button5->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button5->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button5->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button5->Location = System::Drawing::Point(91, 91);

this->button5->Name = L"button5";

this->button5->Size = System::Drawing::Size(82, 82);

this->button5->TabIndex = 5;

this->button5->TabStop = false;

this->button5->Tag = L"4";

this->button5->Text = L"5";

this->button5->UseVisualStyleBackColor = true;

this->button5->Click += gcnew System::EventHandler(this, &Form1::button5\_Click);

//

// button4

//

this->button4->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button4->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button4->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button4->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button4->Location = System::Drawing::Point(3, 91);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(82, 82);

this->button4->TabIndex = 4;

this->button4->TabStop = false;

this->button4->Tag = L"3";

this->button4->Text = L"4";

this->button4->UseVisualStyleBackColor = true;

this->button4->Click += gcnew System::EventHandler(this, &Form1::button4\_Click);

//

// button2

//

this->button2->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button2->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button2->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button2->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button2->Location = System::Drawing::Point(91, 3);

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

this->button2->Size = System::Drawing::Size(82, 82);

this->button2->TabIndex = 2;

this->button2->TabStop = false;

this->button2->Tag = L"1";

this->button2->Text = L"2";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &Form1::button2\_Click);

//

// button1

//

this->button1->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button1->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button1->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button1->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button1->Location = System::Drawing::Point(3, 3);

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

this->button1->Size = System::Drawing::Size(82, 82);

this->button1->TabIndex = 1;

this->button1->TabStop = false;

this->button1->Tag = L"0";

this->button1->Text = L"1";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &Form1::button1\_Click);

//

// button6

//

this->button6->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button6->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button6->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button6->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button6->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button6->Location = System::Drawing::Point(179, 91);

this->button6->Name = L"button6";

this->button6->Size = System::Drawing::Size(82, 82);

this->button6->TabIndex = 6;

this->button6->TabStop = false;

this->button6->Tag = L"5";

this->button6->Text = L"6";

this->button6->UseVisualStyleBackColor = true;

this->button6->Click += gcnew System::EventHandler(this, &Form1::button6\_Click);

//

// button7

//

this->button7->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button7->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button7->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button7->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button7->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button7->Location = System::Drawing::Point(3, 179);

this->button7->Name = L"button7";

this->button7->Size = System::Drawing::Size(82, 82);

this->button7->TabIndex = 7;

this->button7->TabStop = false;

this->button7->Tag = L"6";

this->button7->Text = L"7";

this->button7->UseVisualStyleBackColor = true;

this->button7->Click += gcnew System::EventHandler(this, &Form1::button7\_Click);

//

// button9

//

this->button9->Enabled = false;

this->button9->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button9->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button9->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button9->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button9->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button9->Location = System::Drawing::Point(179, 179);

this->button9->Name = L"button9";

this->button9->Size = System::Drawing::Size(83, 83);

this->button9->TabIndex = 9;

this->button9->TabStop = false;

this->button9->Tag = L"8";

this->button9->UseVisualStyleBackColor = true;

this->button9->Click += gcnew System::EventHandler(this, &Form1::button9\_Click);

//

// button3

//

this->button3->FlatAppearance->BorderColor = System::Drawing::Color::Silver;

this->button3->FlatAppearance->MouseDownBackColor = System::Drawing::Color::Silver;

this->button3->FlatAppearance->MouseOverBackColor = System::Drawing::Color::Silver;

this->button3->FlatStyle = System::Windows::Forms::FlatStyle::System;

this->button3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20));

this->button3->Location = System::Drawing::Point(179, 3);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(82, 82);

this->button3->TabIndex = 3;

this->button3->TabStop = false;

this->button3->Tag = L"2";

this->button3->Text = L"3";

this->button3->UseVisualStyleBackColor = true;

this->button3->Click += gcnew System::EventHandler(this, &Form1::button3\_Click);

//

// tableLayoutPanel2

//

this->tableLayoutPanel2->BackColor = System::Drawing::SystemColors::ButtonShadow;

this->tableLayoutPanel2->ColumnCount = 3;

this->tableLayoutPanel2->ColumnStyles->Add((gcnew System::Windows::Forms::ColumnStyle(System::Windows::Forms::SizeType::Percent,

33.33333F)));

this->tableLayoutPanel2->ColumnStyles->Add((gcnew System::Windows::Forms::ColumnStyle(System::Windows::Forms::SizeType::Percent,

33.33334F)));

this->tableLayoutPanel2->ColumnStyles->Add((gcnew System::Windows::Forms::ColumnStyle(System::Windows::Forms::SizeType::Percent,

33.33334F)));

this->tableLayoutPanel2->Controls->Add(this->textBox9, 2, 2);

this->tableLayoutPanel2->Controls->Add(this->textBox8, 1, 2);

this->tableLayoutPanel2->Controls->Add(this->textBox7, 0, 2);

this->tableLayoutPanel2->Controls->Add(this->textBox6, 2, 1);

this->tableLayoutPanel2->Controls->Add(this->textBox5, 1, 1);

this->tableLayoutPanel2->Controls->Add(this->textBox4, 0, 1);

this->tableLayoutPanel2->Controls->Add(this->textBox3, 2, 0);

this->tableLayoutPanel2->Controls->Add(this->textBox2, 1, 0);

this->tableLayoutPanel2->Controls->Add(this->textBox1, 0, 0);

this->tableLayoutPanel2->Location = System::Drawing::Point(63, 12);

this->tableLayoutPanel2->Name = L"tableLayoutPanel2";

this->tableLayoutPanel2->RowCount = 3;

this->tableLayoutPanel2->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Percent, 33.33333F)));

this->tableLayoutPanel2->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Percent, 33.33333F)));

this->tableLayoutPanel2->RowStyles->Add((gcnew System::Windows::Forms::RowStyle(System::Windows::Forms::SizeType::Percent, 33.33333F)));

this->tableLayoutPanel2->Size = System::Drawing::Size(165, 165);

this->tableLayoutPanel2->TabIndex = 1;

//

// textBox9

//

this->textBox9->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox9->Location = System::Drawing::Point(112, 113);

this->textBox9->MaxLength = 1;

this->textBox9->Name = L"textBox9";

this->textBox9->Size = System::Drawing::Size(48, 49);

this->textBox9->TabIndex = 10;

this->textBox9->Tag = L"8";

this->textBox9->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox8

//

this->textBox8->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox8->Location = System::Drawing::Point(57, 113);

this->textBox8->MaxLength = 1;

this->textBox8->Name = L"textBox8";

this->textBox8->Size = System::Drawing::Size(48, 49);

this->textBox8->TabIndex = 9;

this->textBox8->Tag = L"7";

this->textBox8->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox7

//

this->textBox7->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox7->Location = System::Drawing::Point(3, 113);

this->textBox7->MaxLength = 1;

this->textBox7->Name = L"textBox7";

this->textBox7->Size = System::Drawing::Size(48, 49);

this->textBox7->TabIndex = 8;

this->textBox7->Tag = L"6";

this->textBox7->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox6

//

this->textBox6->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox6->Location = System::Drawing::Point(112, 58);

this->textBox6->MaxLength = 1;

this->textBox6->Name = L"textBox6";

this->textBox6->Size = System::Drawing::Size(48, 49);

this->textBox6->TabIndex = 7;

this->textBox6->Tag = L"5";

this->textBox6->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox5

//

this->textBox5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox5->Location = System::Drawing::Point(57, 58);

this->textBox5->MaxLength = 1;

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(48, 49);

this->textBox5->TabIndex = 6;

this->textBox5->Tag = L"4";

this->textBox5->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox4

//

this->textBox4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox4->Location = System::Drawing::Point(3, 58);

this->textBox4->MaxLength = 1;

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(48, 49);

this->textBox4->TabIndex = 5;

this->textBox4->Tag = L"3";

this->textBox4->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox3

//

this->textBox3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox3->Location = System::Drawing::Point(112, 3);

this->textBox3->MaxLength = 1;

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(48, 49);

this->textBox3->TabIndex = 4;

this->textBox3->Tag = L"2";

this->textBox3->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox2

//

this->textBox2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox2->Location = System::Drawing::Point(57, 3);

this->textBox2->MaxLength = 1;

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(48, 49);

this->textBox2->TabIndex = 3;

this->textBox2->Tag = L"1";

this->textBox2->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox1

//

this->textBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 22));

this->textBox1->Location = System::Drawing::Point(3, 3);

this->textBox1->MaxLength = 1;

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(48, 49);

this->textBox1->TabIndex = 2;

this->textBox1->Tag = L"0";

this->textBox1->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// Apply

//

this->Apply->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10));

this->Apply->Location = System::Drawing::Point(15, 183);

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

this->Apply->Size = System::Drawing::Size(82, 32);

this->Apply->TabIndex = 2;

this->Apply->TabStop = false;

this->Apply->Text = L"Apply";

this->Apply->UseVisualStyleBackColor = true;

this->Apply->Click += gcnew System::EventHandler(this, &Form1::Apply\_Click);

//

// flowLayoutPanel1

//

this->flowLayoutPanel1->BorderStyle = System::Windows::Forms::BorderStyle::Fixed3D;

this->flowLayoutPanel1->Location = System::Drawing::Point(3, 5);

this->flowLayoutPanel1->Name = L"flowLayoutPanel1";

this->flowLayoutPanel1->Size = System::Drawing::Size(284, 490);

this->flowLayoutPanel1->TabIndex = 3;

//

// radioButton1

//

this->radioButton1->AutoSize = true;

this->radioButton1->Checked = true;

this->radioButton1->Location = System::Drawing::Point(321, 12);

this->radioButton1->Name = L"radioButton1";

this->radioButton1->Size = System::Drawing::Size(54, 20);

this->radioButton1->TabIndex = 4;

this->radioButton1->TabStop = true;

this->radioButton1->Text = L"BFS";

this->radioButton1->UseVisualStyleBackColor = true;

//

// radioButton2

//

this->radioButton2->AutoSize = true;

this->radioButton2->Location = System::Drawing::Point(321, 41);

this->radioButton2->Name = L"radioButton2";

this->radioButton2->Size = System::Drawing::Size(62, 20);

this->radioButton2->TabIndex = 5;

this->radioButton2->Text = L"LDFS";

this->radioButton2->UseVisualStyleBackColor = true;

//

// flowLayoutPanel2

//

this->flowLayoutPanel2->BorderStyle = System::Windows::Forms::BorderStyle::Fixed3D;

this->flowLayoutPanel2->Location = System::Drawing::Point(310, 5);

this->flowLayoutPanel2->Name = L"flowLayoutPanel2";

this->flowLayoutPanel2->Size = System::Drawing::Size(314, 67);

this->flowLayoutPanel2->TabIndex = 6;

//

// Start

//

this->Start->Location = System::Drawing::Point(502, 25);

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

this->Start->Size = System::Drawing::Size(98, 27);

this->Start->TabIndex = 7;

this->Start->Text = L"Start!";

this->Start->UseVisualStyleBackColor = true;

this->Start->Click += gcnew System::EventHandler(this, &Form1::Start\_Click);

//

// Clear

//

this->Clear->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10));

this->Clear->Location = System::Drawing::Point(191, 183);

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

this->Clear->Size = System::Drawing::Size(82, 32);

this->Clear->TabIndex = 8;

this->Clear->TabStop = false;

this->Clear->Text = L"Clear";

this->Clear->UseVisualStyleBackColor = true;

this->Clear->Click += gcnew System::EventHandler(this, &Form1::Clear\_Click);

//

// textBox10

//

this->textBox10->Location = System::Drawing::Point(321, 89);

this->textBox10->Multiline = true;

this->textBox10->Name = L"textBox10";

this->textBox10->ReadOnly = true;

this->textBox10->ScrollBars = System::Windows::Forms::ScrollBars::Vertical;

this->textBox10->Size = System::Drawing::Size(292, 356);

this->textBox10->TabIndex = 9;

this->textBox10->TabStop = false;

//

// flowLayoutPanel3

//

this->flowLayoutPanel3->BorderStyle = System::Windows::Forms::BorderStyle::Fixed3D;

this->flowLayoutPanel3->Location = System::Drawing::Point(310, 78);

this->flowLayoutPanel3->Name = L"flowLayoutPanel3";

this->flowLayoutPanel3->Size = System::Drawing::Size(314, 417);

this->flowLayoutPanel3->TabIndex = 7;

//

// textBox11

//

this->textBox11->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9));

this->textBox11->Location = System::Drawing::Point(389, 38);

this->textBox11->Name = L"textBox11";

this->textBox11->Size = System::Drawing::Size(98, 24);

this->textBox11->TabIndex = 10;

this->textBox11->TextChanged += gcnew System::EventHandler(this, &Form1::textBox11\_TextChanged);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10));

this->label1->Location = System::Drawing::Point(385, 12);

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

this->label1->Size = System::Drawing::Size(59, 20);

this->label1->TabIndex = 0;

this->label1->Text = L"Depth:";

//

// Read

//

this->Read->Location = System::Drawing::Point(321, 451);

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

this->Read->Size = System::Drawing::Size(138, 33);

this->Read->TabIndex = 0;

this->Read->Text = L"Read File";

this->Read->UseVisualStyleBackColor = true;

this->Read->Click += gcnew System::EventHandler(this, &Form1::Read\_Click);

//

// ClearFile

//

this->ClearFile->Location = System::Drawing::Point(480, 451);

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

this->ClearFile->Size = System::Drawing::Size(133, 33);

this->ClearFile->TabIndex = 11;

this->ClearFile->Text = L"Clear File";

this->ClearFile->UseVisualStyleBackColor = true;

this->ClearFile->Click += gcnew System::EventHandler(this, &Form1::ClearFile\_Click);

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

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

this->ClientSize = System::Drawing::Size(636, 500);

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

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

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

this->Controls->Add(this->textBox11);

this->Controls->Add(this->textBox10);

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

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

this->Controls->Add(this->radioButton2);

this->Controls->Add(this->radioButton1);

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

this->Controls->Add(this->tableLayoutPanel2);

this->Controls->Add(this->tableLayoutPanel1);

this->Controls->Add(this->flowLayoutPanel1);

this->Controls->Add(this->flowLayoutPanel2);

this->Controls->Add(this->flowLayoutPanel3);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedSingle;

this->MaximizeBox = false;

this->Name = L"Form1";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"8-puzzle";

this->Load += gcnew System::EventHandler(this, &Form1::Form1\_Load);

this->tableLayoutPanel1->ResumeLayout(false);

this->tableLayoutPanel2->ResumeLayout(false);

this->tableLayoutPanel2->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: cli::array<Button^, 1>^ createArr(); // Заповнення масиву кнопок

private: cli::array<TextBox^, 1>^ createTextArr(); // Заповнення масиву текстових полів

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button1, arr); }

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

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button3, arr); }

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button4, arr); }

private: System::Void button5\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button5, arr); }

private: System::Void button6\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button6, arr); }

private: System::Void button7\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button7, arr); }

private: System::Void button8\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button8, arr); }

private: System::Void button9\_Click(System::Object^ sender, System::EventArgs^ e) { mooveButton(button9, arr); }

private: System::Void Apply\_Click(System::Object^ sender, System::EventArgs^ e);

private: System::Void Start\_Click(System::Object^ sender, System::EventArgs^ e);

private: System::Void Clear\_Click(System::Object^ sender, System::EventArgs^ e);

private: System::Void textBox11\_TextChanged(System::Object^ sender, System::EventArgs^ e);

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e);

private: System::Void Read\_Click(System::Object^ sender, System::EventArgs^ e);

private: System::Void ClearFile\_Click(System::Object^ sender, System::EventArgs^ e);

};

}

# Form1.cpp

#include "pch.h"

#include "Form1.h"

#include <random>

#include <fstream>

#include <msclr\marshal\_cppstd.h>

namespace CppCLRWinFormsProject {

using namespace std;

cli::array<Button^, 1>^ Form1::createArr() { // Створення масиву кнопок

cli::array<Button^, 1>^ arr = gcnew cli::array<Button^, 1>(9);

arr[0] = button1;

arr[1] = button2;

arr[2] = button3;

arr[3] = button4;

arr[4] = button5;

arr[5] = button6;

arr[6] = button7;

arr[7] = button8;

arr[8] = button9;

return arr;

}

cli::array<TextBox^, 1>^ Form1::createTextArr() { // Створення масиву текстових полів

cli::array<TextBox^, 1>^ arr1 = gcnew cli::array<TextBox^, 1>(9);

arr1[0] = textBox1;

arr1[1] = textBox2;

arr1[2] = textBox3;

arr1[3] = textBox4;

arr1[4] = textBox5;

arr1[5] = textBox6;

arr1[6] = textBox7;

arr1[7] = textBox8;

arr1[8] = textBox9;

return arr1;

}

System::Void Form1::Apply\_Click(System::Object^ sender, System::EventArgs^ e) { // Кнопка для перенесення написів з текстових полів на кнопки

for (int i = 0; i < 9; i++) { // Перевірка всіх чисел текстових полів

char curText = (msclr::interop::marshal\_as<string>(arrText[i]->Text->ToString()))[0];

if (isdigit(curText)) { // Чи є символ цифрою

if (atoi(&curText) != 9) { // Чи не є символ девяткою

for (int j = 0; j < 9; j++)

if (curText == (msclr::interop::marshal\_as<string>(arrText[j]->Text->ToString()))[0] && arrText[i]->Tag != arrText[j]->Tag) // Перевірка на однакові цифри

arrText[i]->Text = "";

}

else

arrText[i]->Text = "";

}

else

arrText[i]->Text = "";

}

int arrNum[9];

bool fill = true;

for (int i = 0; i < 9; i++) {

if (arrText[i]->Text == "") { // Перевірка чи всі цифри введені коректно

MessageBox::Show("Invalid values!", "Error!", MessageBoxButtons::OK, MessageBoxIcon::Error);

fill = false;

break;

}

arrNum[i] = stoi(msclr::interop::marshal\_as<string>(arrText[i]->Text->ToString()));

}

if (fill) // Перенесення значень на кнопки

for (int i = 0; i < 9; i++) {

if (arrNum[i] == 0) {

arr[i]->Text = "";

arr[i]->Enabled = false;

}

else {

arr[i]->Text = arrNum[i].ToString();

arr[i]->Enabled = true;

}

}

}

System::Void Form1::Start\_Click(System::Object^ sender, System::EventArgs^ e) { // Кнопка для вирішення 8-puzzle

int depth = 0; // Максимальна глибина для LDFS алгоритму

if (textBox11->Text != "")

depth = stoi(msclr::interop::marshal\_as<string>(textBox11->Text->ToString()));

int \*arrNumer = new int[9]; // Масив для цифр з поля 8-puzzle

for (int i = 0; i < 9; i++) {

string an = msclr::interop::marshal\_as<string>(arr[i]->Text->ToString());

if (an != "")

arrNumer[i] = stoi(an);

else

arrNumer[i] = 0;

}

vector<int\*> sol; // Шлях до вирішення задачі

if (radioButton1->Checked)

sol = BFS(arrNumer, textBox10); // BFS алгоритм

else {

Puzzle\* root = new Puzzle(arrNumer); // Перший елемент

vector<Puzzle\*> visited; // Елементи які були оброблені

int time\_start = clock(); // Час початку роботи алгоритму

root = DLS(root, visited, 0, depth); // LDFS(DLS) алгоритм

int time\_finish = clock(); // Час закінчення роботи алгоритму

sol = SolutionPath(root); // Шлях до вирішення

Display(sol, textBox10, "LDFS", time\_finish - time\_start); // Вивід результатів

Info write(sol.size() - 1, time\_finish - time\_start, "LDFS", sol[sol.size() - 1], sol[0], getCurDate(), depth); // Об'єкт де зберігаються результати роботи алгоритму

WriteToFile(write); // Запис результатів у файл

}

if (!chack(sol[0]))

MessageBox::Show("Too small depth!", "Error!", MessageBoxButtons::OK, MessageBoxIcon::Error);

for (int i = 0; i < 9; i++) { // Відображення результату на кнопки

if (sol[0][i] != 0) {

arr[i]->Text = sol[0][i].ToString();

arr[i]->Enabled = true;

}

else {

arr[i]->Text = "";

arr[i]->Enabled = false;

}

}

}

System::Void Form1::Clear\_Click(System::Object^ sender, System::EventArgs^ e) { // Вернути 8-puzzle в початкове положення

for (int i = 1; i < 9; i++) {

arr[i - 1]->Text = i.ToString();

arr[i - 1]->Enabled = true;

}

arr[8]->Enabled = false;

arr[8]->Text = "";

}

System::Void Form1::textBox11\_TextChanged(System::Object^ sender, System::EventArgs^ e) { // Поле для введення глубини

if (textBox11->Text != "") { //Первірка чи не пусте поле

string line = msclr::interop::marshal\_as<string>(textBox11->Text->ToString());

for(int i = 0; i < line.size(); i++)

if (!isdigit(line[i])) { // Перевірка чи є символ цифрою

line = line.substr(0, i) + line.substr(i + 1);

textBox11->Text = gcnew System::String(line.c\_str());

textBox11->Select(i, 0);

}

}

}

System::Void Form1::Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

textBox11->Text = "10"; // Значення за замовчуванням

}

System::Void Form1::Read\_Click(System::Object^ sender, System::EventArgs^ e) {

ReadFile(textBox10);

}

System::Void Form1::ClearFile\_Click(System::Object^ sender, System::EventArgs^ e) {

ofstream File("Results.txt", ios::binary);

File.close();

MessageBox::Show("File is empty now!");

}

}

# Header.h

#pragma once

#include "pch.h"

#include <vector>

#include "Puzzle.h"

#include "Info.h"

using namespace std;

void mooveButton(System::Windows::Forms::Button^, cli::array<System::Windows::Forms::Button^, 1>^); // Рух кнопки

vector<int\*> BFS(int\*, System::Windows::Forms::TextBox^);// BFS алгоритм

Puzzle\* DLS(Puzzle\*, vector<Puzzle\*>, int, int);// DLS алгоритм

bool chack(int\*);// Перевірка на шуканий результат

void removeData(vector<Puzzle\*>); // Видалення вузлів

bool Contains(vector<Puzzle\*>, int\*); // Перевірка чи був обролений вузол

vector<int\*> SolutionPath(Puzzle\*); // Формування вектора кроків до вирішення

void Display(vector<int\*>, System::Windows::Forms::TextBox^, string, int); // Вивід результатів у TextBox

void WriteToFile(Info); // Запис у файл

void ReadFile(System::Windows::Forms::TextBox^); // Читання з файлу

string addZero(string); // Додавання нуля до дати або часу

char\* getCurDate(); // Поточна дата і час

# Func.cpp

#include "pch.h"

#include "Header.h"

#include <string>

#include <ctime>

#include <fstream>

#include <msclr\marshal\_cppstd.h>

using namespace std;

void mooveButton(System::Windows::Forms::Button^ curButt, cli::array<System::Windows::Forms::Button^, 1>^ arr) { // Переміщення кнопок вручну

int currTag = stoi(msclr::interop::marshal\_as<string>(curButt->Tag->ToString())); // Поточна кнопка

if (currTag + 1 < 9) // Чи не виходить за поле

if (arr[currTag + 1]->Text == "" && currTag / 3 == (currTag + 1) / 3) { // Чи можна рухати цю кнопку

string text = msclr::interop::marshal\_as<string>(curButt->Text->ToString()); // Напис на кнопці

curButt->Text = ""; // Пересунути 0 на місце поточної кнопки

arr[currTag + 1]->Text = stoi(text) + ""; // Записати на місце 0 текст поточної кнопки

curButt->Enabled = false; // Зробити 0 недоступним для натискання

arr[currTag + 1]->Enabled = true; // Відновити можливість натимнення на кнопку де немає 0

}

if (currTag - 1 >= 0) // Чи не виходить за поле

if (arr[currTag - 1]->Text == "" && currTag / 3 == (currTag - 1) / 3) {// Чи можна рухати цю кнопку

string text = msclr::interop::marshal\_as<string>(curButt->Text->ToString());

curButt->Text = "";

arr[currTag - 1]->Text = stoi(text) + "";

curButt->Enabled = false;

arr[currTag - 1]->Enabled = true;

}

if (currTag + 3 < 9) // Чи не виходить за поле

if (arr[currTag + 3]->Text == "" && currTag / 3 + 1 == (currTag + 3) / 3) {// Чи можна рухати цю кнопку

string text = msclr::interop::marshal\_as<string>(curButt->Text->ToString());

curButt->Text = "";

arr[currTag + 3]->Text = stoi(text) + "";

curButt->Enabled = false;

arr[currTag + 3]->Enabled = true;

}

if (currTag - 3 >= 0) // Чи не виходить за поле

if (arr[currTag - 3]->Text == "" && currTag / 3 - 1 == (currTag - 3) / 3) {// Чи можна рухати цю кнопку

string text = msclr::interop::marshal\_as<string>(curButt->Text->ToString());

curButt->Text = "";

arr[currTag - 3]->Text = stoi(text) + "";

curButt->Enabled = false;

arr[currTag - 3]->Enabled = true;

}

}

vector<int\*> BFS(int\* ar, System::Windows::Forms::TextBox^ text) { // БФС алгоритм

unsigned int start\_time = clock(); // Час початку роботи алгоритму

Puzzle\* vert = new Puzzle(ar); // Вузол дерева

vector<Puzzle\*> bfs; // Черга

vector<Puzzle\*> closed; // Елементи які були проаналізовані

bfs.push\_back(vert); // Додавання вузла до черги

Puzzle\* pointTmp = vert; // Покажчик на поточний вузол

while (!chack(pointTmp->getArr())) { // Поки не знайдено рішення

closed.push\_back(pointTmp); // Вузол оброблено

pointTmp = bfs[0]; // Перший елемент черги

bfs.erase(bfs.begin()); // Видалення першого елементу черги

vector<Puzzle\*> mooves = pointTmp->mooves(); // Всі можливі переміщення

for (Puzzle\* vert : mooves) { // Обробка всіх переміщень

if (!Contains(closed, vert->getArr()) && !Contains(bfs, vert->getArr())) // Перевірка чи був колись оброблений цей вузол

bfs.push\_back(vert); // Додати вузол до черги

else

delete vert; // Видалення не потрібної вершини

}

}

vector<int\*> solution = SolutionPath(pointTmp); // Шлях до вирішенняa

unsigned int finish\_time = clock(); // Час закінчення роботи алгоритму

Display(solution, text, "BFS", finish\_time - start\_time); // Вивід результату роботи програми у TextBox

removeData(closed);

removeData(bfs);

Info write(solution.size() - 1, finish\_time - start\_time, "BFS", solution[solution.size() - 1], solution[0], getCurDate(), 0); // Створення об'єкту для запису у файл

WriteToFile(write); // Запис об'єкта у файл

return solution;

}

char\* getCurDate() { // Поточний час і дата

time\_t seconds = time(NULL);

tm\* timeinfo = localtime(&seconds);

string date = addZero(to\_string(timeinfo->tm\_mday)) + "." + addZero(to\_string(timeinfo->tm\_mon + 1)) + "." + to\_string(timeinfo->tm\_year + 1900); // Поточна дата

string time = addZero(to\_string(timeinfo->tm\_hour)) + ":" + addZero(to\_string(timeinfo->tm\_min)); // Поточний час

string TimeDate = date + " --- " + time;

char\* data = new char[20];

strcpy(data, TimeDate.c\_str());

return data;

}

string addZero(string Date) { // Додавання нуля за відсутності

if (Date.size() == 1)

return "0" + Date;

return Date;

}

bool chack(int \*arr) { // Перевірка на правильну відповідь

bool checked = true;

for (int i = 0; i < 8; i++)

if (arr[i] != i + 1)

checked = false;

return checked;

}

bool Contains(vector<Puzzle\*> list, int\* element) { // Чи є елемент у векторі

bool isEquel = true;

for (int i = 0; i < list.size(); i++) { // Обхід вектора

isEquel = true;

int\* arrVec = list[i]->getArr();

for (int j = 0; j < 9; j++)

if (element[j] != arrVec[j]) // Порівнення елементів

isEquel = false;

if (isEquel)

break;

}

if (list.size() == 0)

isEquel = false;

return isEquel;

}

void removeData(vector<Puzzle\*> del) { // Видалення непотрібних вузлів

for (int i = 1; i < del.size(); i++)

delete del[i];

}

vector<int\*> SolutionPath(Puzzle\* solution) { // Формування вектору з масиву на кожен крок до вирішення

vector<int\*> Path; // Шлях до вирішення

Puzzle\* point = solution;

while (point->getAnc()) { // Поки існує балько

int\* newArr = new int[9];

for (int i = 0; i < 9; i++)

newArr[i] = point->getArr()[i];

Path.push\_back(newArr); // Записуємо масив до шляху

point = point->getAnc();

}

int\* newArr = new int[9];

for (int i = 0; i < 9; i++)

newArr[i] = point->getArr()[i];

Path.push\_back(newArr); // Записуємо кореневий елемент до шляху

return Path;

}

void Display(vector<int\*> solutionArr, System::Windows::Forms::TextBox^ text, string name, int sec) { // Вивід інформації у TextBox

int minuts = sec / 60000; // Хвилини

int seconds = (sec % 60000)/1000; // Секунди

int milisec = sec % 1000; // Мілісекунди

string time = to\_string(minuts) + "m. " + to\_string(seconds) + "s. " + to\_string(milisec) + "ms.--";

text->Text = " Result: ";

text->Text += "\r\n" + "-" + gcnew System::String(name.c\_str()) + "\r\n--" + gcnew System::String(to\_string(solutionArr.size() - 1).c\_str()) + " steps--\r\n--took " + gcnew System::String(time.c\_str());

string oneT, secT, serdT; // Три рядки матриці 8-puzzle

for (int j = 0; j < solutionArr.size(); j++) { // Вивід шляху до кінцевого результату

oneT = secT = serdT = "";

for (int i = 0; i < 9; i++) {

if (i / 3 == 0)

oneT += to\_string(solutionArr[solutionArr.size() - 1 - j][i]) + " "; // Запис у перший рядок

if (i / 3 == 1)

secT += to\_string(solutionArr[solutionArr.size() - 1 - j][i]) + " "; // Запис у другий рядок

if (i / 3 == 2)

serdT += to\_string(solutionArr[solutionArr.size() - 1 - j][i]) + " "; // Запис у третій рядок

}

text->Text += "\r\n" + gcnew System::String(to\_string(j).c\_str()) + ".";

text->Text += "\r\n " + gcnew System::String(oneT.c\_str());

text->Text += "\r\n " + gcnew System::String(secT.c\_str());

text->Text += "\r\n " + gcnew System::String(serdT.c\_str()) + "\r\n";

}

}

Puzzle\* DLS(Puzzle\* element, vector<Puzzle\*> visited, int tmpDepth, int maxDepth) { // DLS алгоритм

if (tmpDepth >= maxDepth) // Перевірка на максимальну глубину

return element;

if (chack(element->getArr())) // Чи є елемент шуканим

return element;

visited.push\_back(element); // Додати у оброблені вершини

vector<Puzzle\*> mooves = element->mooves(); // Всі можливі рухи

for (Puzzle\* vert : mooves) {

if (chack(vert->getArr())) // Чи є шуканим

return vert;

if (Contains(visited, vert->getArr())) // Чи був цей елемент колись вже перевірений

delete vert;

else {

element = DLS(vert, visited, tmpDepth + 1, maxDepth); // Обробка нового елементу

if (chack(element->getArr())) // Чи є елемент шуканим

return element;

}

}

return element;

}

void WriteToFile(Info result) { // Запис у файл

ofstream File("Results.txt", ios::binary | ios::app); // Відкриття бінарного файла для запису

if (!File.is\_open())

System::Windows::Forms::MessageBox::Show("File is not open!", "Error!");

else

File.write((char\*)&result, sizeof(Info)); // Запис у файл

File.close();

}

void ReadFile(System::Windows::Forms::TextBox^ text) { // Читання із файла і виведення результатів у TextBox

ifstream File1("Results.txt", ios::binary); // Відриття бінарного файла для читання

long file\_size; // Довжина файла

File1.seekg(0, ios::end); // Переміщення файлового покажчика на цінець файлу

file\_size = File1.tellg(); // Позиція покажчика у файлі

string one, two, three;

File1.seekg(0, ios::beg); // Переміщення файлового покажчика на початок файлу

if (file\_size != 0) {

Info output;

text->Text = "Results.txt:\r\n";

int i = 1;

while (File1.read((char\*)&output, sizeof(Info))) { // Читання із файлу

string Date = output.getDate();

Date = Date.substr(0, 20);

text->Text += "\r\n" + gcnew System::String(to\_string(i).c\_str()) + ". " + gcnew System::String(Date.c\_str()); // Дата запису

text->Text += "\r\n" + gcnew System::String(output.getName()); // Назва алгоритму

if (output.getName()[0] == 'L')

text->Text += "\r\nDepth: " + output.getDepth(); // Глибина для LDFS

text->Text += "\r\n" + "--" + gcnew System::String(to\_string(output.getSteps()).c\_str()) + " steps--"; // Кількість кроків для вирішення

text->Text += "\r\n" + "--took " + gcnew System::String(output.getWorkTime().c\_str()) + "--\r\nStart:"; // Час роботи алгоритму

int\* arr = output.getArrStart();

one = two = three = "";

for (int i = 0; i < 9; i++) {

if (i / 3 == 0) one += to\_string(arr[i]) + " ";

if (i / 3 == 1) two += to\_string(arr[i]) + " ";

if (i / 3 == 2) three += to\_string(arr[i]) + " ";

}

text->Text += " " + gcnew System::String(one.c\_str()); // Виведення позицій кнопок

text->Text += "\r\n " + gcnew System::String(two.c\_str()); // при запуску алгоритму

text->Text += "\r\n " + gcnew System::String(three.c\_str()); //

text->Text += "\r\n";

arr = output.getArrFinish();

one = two = three = "";

for (int i = 0; i < 9; i++) {

if (i / 3 == 0) one += to\_string(arr[i]) + " ";

if (i / 3 == 1) two += to\_string(arr[i]) + " ";

if (i / 3 == 2) three += to\_string(arr[i]) + " ";

}

text->Text += "\r\nFiish:";

text->Text += " " + gcnew System::String(one.c\_str()); // Виведення позицій кнопок

text->Text += "\r\n " + gcnew System::String(two.c\_str()); // при закінченні обробки

text->Text += "\r\n " + gcnew System::String(three.c\_str()); //

text->Text += "\r\n";

i++;

}

}

else

System::Windows::Forms::MessageBox::Show("File is empty!", "Error!", System::Windows::Forms::MessageBoxButtons::OK, System::Windows::Forms::MessageBoxIcon::Error);

File1.close();

}

# Puzzle.h

#pragma once

#include <iostream>

#include <vector>

class Puzzle

{

int \*arr; // Масив цифр

int posZero; // Позиція порожньої кнопки

Puzzle\* anc; // Батько поточного елементу

public:

Puzzle(int\*, Puzzle\* = NULL);

~Puzzle() { delete arr; }

int\* getArr() { return arr; }

int getZeropos() { return posZero; }

Puzzle\* getAnc() { return anc; }

std::vector<Puzzle\*> mooves(); // Всі можливі переміщення

Puzzle\* mooveRight(); // Переміщення вправо

Puzzle\* mooveLeft(); // Переміщення вліво

Puzzle\* mooveUp(); // Переміщення вверх

Puzzle\* mooveDown(); // Переміщення вниз

};

# Puzzle.cpp

#include "pch.h"

#include "Puzzle.h"

using namespace std;

Puzzle::Puzzle(int\* ar, Puzzle\* ancictor) {

arr = new int[9];

for (int i = 0; i < 9; i++)

arr[i] = ar[i];

for (int i = 0; i < 9; i++)

if (arr[i] == 0)

posZero = i;

anc = ancictor;

}

vector<Puzzle\*> Puzzle::mooves() {

vector<Puzzle\*> moov;

Puzzle\* vertix = this->mooveRight();

if(vertix != NULL)

moov.push\_back(vertix);

vertix = this->mooveLeft();

if (vertix != NULL)

moov.push\_back(vertix);

vertix = this->mooveUp();

if (vertix != NULL)

moov.push\_back(vertix);

vertix = this->mooveDown();

if (vertix != NULL)

moov.push\_back(vertix);

vertix = NULL;

delete vertix;

return moov;

}

Puzzle\* Puzzle::mooveRight() {

int zeroPos = this->getZeropos();

if (zeroPos + 1 < 9 && zeroPos / 3 == (zeroPos + 1) / 3) {

int\* arr = new int[9];

int\* oldArr = this->getArr();

for (int i = 0; i < 9; i++)

arr[i] = oldArr[i];

arr[zeroPos] = arr[zeroPos + 1];

arr[zeroPos + 1] = 0;

Puzzle\* vertix = new Puzzle(arr, this);

delete[] arr;

return vertix;

}

return NULL;

}

Puzzle\* Puzzle::mooveLeft() {

int zeroPos = this->getZeropos();

if (zeroPos - 1 >= 0 && zeroPos / 3 == (zeroPos - 1) / 3) {

int\* arr = new int[9];

int\* oldArr = this->getArr();

for (int i = 0; i < 9; i++)

arr[i] = oldArr[i];

arr[zeroPos] = arr[zeroPos - 1];

arr[zeroPos - 1] = 0;

Puzzle\* vertix = new Puzzle(arr, this);

delete[] arr;

return vertix;

}

return NULL;

}

Puzzle\* Puzzle::mooveUp() {

int zeroPos = this->getZeropos();

if (zeroPos + 3 < 9) {

int\* arr = new int[9];

int\* oldArr = this->getArr();

for (int i = 0; i < 9; i++)

arr[i] = oldArr[i];

arr[zeroPos] = arr[zeroPos + 3];

arr[zeroPos + 3] = 0;

Puzzle\* vertix = new Puzzle(arr, this);

delete[] arr;

return vertix;

}

return NULL;

}

Puzzle\* Puzzle::mooveDown() {

int zeroPos = this->getZeropos();

if (zeroPos - 3 >= 0) {

int\* arr = new int[9];

int\* oldArr = this->getArr();

for (int i = 0; i < 9; i++)

arr[i] = oldArr[i];

arr[zeroPos] = arr[zeroPos - 3];

arr[zeroPos - 3] = 0;

Puzzle\* vertix = new Puzzle(arr, this);

delete[] arr;

return vertix;

}

return NULL;

}

# pch.h

// файл за замовчуванням

#ifndef PCH\_H

#define PCH\_H

// add headers that you want to pre-compile here

#endif //PCH\_H

# pch.cpp

// файл за замовчуванням

// pch.cpp: source file corresponding to the pre-compiled header

#include "pch.h"

// When you are using pre-compiled headers, this source file is necessary for compilation to succeed.

# CppCLR\_WinFormsProject.cpp

// файл за замовчуванням

#include "pch.h"

using namespace System;

// int main(array<System::String ^> ^args)

// {

// return 0;

// }

#include "Form1.h"

using namespace System::Windows::Forms;

[STAThread]

int main()

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew CppCLRWinFormsProject::Form1());

}

# AssembleInfo.cpp

#include "pch.h"

using namespace System;

using namespace System::Reflection;

using namespace System::Runtime::CompilerServices;

using namespace System::Runtime::InteropServices;

using namespace System::Security::Permissions;

[assembly:AssemblyTitleAttribute(L"CppCLRWinFormsProject")];

[assembly:AssemblyDescriptionAttribute(L"")];

[assembly:AssemblyConfigurationAttribute(L"")];

[assembly:AssemblyCompanyAttribute(L"")];

[assembly:AssemblyProductAttribute(L"CppCLRWinFormsProject")];

[assembly:AssemblyCopyrightAttribute(L"Copyright (c) 2022")];

[assembly:AssemblyTrademarkAttribute(L"")];

[assembly:AssemblyCultureAttribute(L"")];

[assembly:AssemblyVersionAttribute(L"1.0.\*")];

[assembly:ComVisible(false)];