НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра

автоматизованих систем обробки інформації та управління

КУРСОВА РОБОТА

|  |
| --- |
| з                                                           «Основи програмування»  (назва дисципліни)  на тему:            «Пошук заданих елементів у масиві» |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  | Студента   І   курсу  ІП-63 групи Шелудько Д.М.  напряму підготовки 6.050103 «Програмна інженерія»  спеціальності «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»  Керівник                          Головченко М.М.                                    (прізвище та ініціали)  асистент    (посада, вчене звання, науковий ступінь)  Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Кількість балів: \_\_\_\_\_\_\_\_\_\_ Оцінка: ECTS \_\_\_\_\_\_\_ | | | Члени комісії | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали) | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали) | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали) | |  | | |     Київ 2016 |

Національний технічний університет України “КПІ”

(назва вищого навчального закладу)

Кафедра Автоматизованих систем обробки інформації та управління

Дисципліна Основи програмування

Напрям "Програмна інженерія"

Курс 1 Група      ІП-63       Семестр 2

**ЗАВДАННЯ**

**на курсову роботу студента**

Шелудько Дмитра Максимовича

(прізвище, ім’я, по батькові)

1. Тема роботи Пошук заданих елементів у масиві

2. Строк здачі студентом закінченої роботи 23.05.2017

3. Вихідні дані до роботи Технічне завдання

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

ПОСТАНОВКА ЗАДАЧІ, ТЕОРЕТИЧНА ЧАСТИНА, АЛГОРИТМИ РОЗВ’ЯЗАННЯ ЗАДАЧІ

ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ТЕСТУВАННЯ ПРОГРАМИ, ІНСТРУКЦІЯ КОРИСТУВАЧА, АНАЛІЗ І УЗАГАЛЬНЕННЯ РЕЗУЛЬТАТІВ, ВИСНОВКИ, СПИСОК ЛІТЕРАТУРИ, ТЕХНІЧНЕ ЗАВДАННЯ, ТЕКСТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

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

Знімки меню розробленого програмного забезпечення

6. Дата видачі завдання 6.03.2017

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Підписи керівника, студента |
| 1. | Отримання теми курсової роботи | 6.03.16-20.03.16 |  |
| 2. | Підготовка ТЗ | 21.03.16-03.04.16 |  |
| 3. | Пошук та вивчення літератури з питань курсової роботи | 04.04.16-17.04.16 |  |
| 4. | Розробка алгоритму вирішення задачі | 18.04.16-24.04.16 |  |
| 6. | Узгодження алгоритму з керівником | 18.04.16-24.04.16 |  |
| 5. | Розробка сценарію роботи програми | 25.04.16-01.05.16 |  |
| 6. | Узгодження сценарію роботи програми з керівником | 25.04.16-01.05.16 |  |
| 7. | Узгодження з керівником інтерфейсу користувача | 25.04.16-01.05.16 |  |
| 8. | Розробка програмного забезпечення | 02.05.16-09.05.16 |  |
| 9. | Налагодження розрахункової частини програми | 02.05.16-11.05.16 |  |
| 10. | Розробка та налагодження інтерфейсної частини програми | 12.05.16-16.05.16 |  |
| 11. | Узгодження з керівником набору тестів для контрольного прикладу | 12.05.16-16.05.16 |  |
| 12. | Тестування програми | 12.05.16-16.05.16 |  |
| 13. | Підготовка пояснювальної записки | 17.05.16-25.05.16 |  |
| 14. | Здача курсової роботи на перевірку | 26.05.16-08.06.16 |  |
| 15. | Захист курсової роботи | 08.06.16-16.03.16 |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  |  |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

|  |  |  |  |
| --- | --- | --- | --- |
| Керівник |  |  | Головченко Максим Миколайович |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

"\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 41 сторінка, 7 рисунків, 7 таблиць.

**Об’єкт дослідження:** масиви

**Мета роботи:** дослідження методів пошуку заданого елементу в масиві лінійним , бінарним , однорідним бінарним , Шара методами та розробка програмного забезпечення для розв’язання типової задачі.

Дана курсова робота включає в себе: опис вищезазначених методів, словесний алгоритм їх реалізації, та опис програмного забезпечення, що реалізує ці методи.

За допомогою розробленого програмного забезпечення перівнено швидкість алгоритмів. У висновках узагальнено підсумок проведеної роботи.

ЛІНІЙНИЙ ПОШУК, БІНАРНИЙ ПОШУК, ОДНОРІДНИЙ БІНАРНИЙ ПОШУК, МЕТОД ШАРА.

ЗМІСТ

[КУРСОВА РОБОТА 1](#_Toc483395194)

[Національний технічний університет України “КПІ” 2](#_Toc483395195)

[Кафедра Автоматизованих систем обробки інформації та управління 2](#_Toc483395196)

[АНОТАЦІЯ 4](#_Toc483395197)

[ЗМІСТ 5](#_Toc483395198)

[ВСТУП 7](#_Toc483395199)

[1 ПОСТАНОВКА ЗАДАЧІ 8](#_Toc483395200)

[2 ТЕОРЕТИЧНА ЧАСТИНА 9](#_Toc483395201)

[2.1.1 Послідовний(лінійний) метод 9](#_Toc483395202)

[Опис лінійного пошуку 9](#_Toc483395203)

[Трудомісткість алгоритму 9](#_Toc483395204)

[2.1.2 Бінарний метод 10](#_Toc483395205)

[Опис бінарного методу 10](#_Toc483395206)

[Трудомісткість алгоритму 10](#_Toc483395207)

[2.1.3 Однорідний бінарний метод 11](#_Toc483395208)

[Опис однорідного бінарного методу 11](#_Toc483395209)

[Трудомісткість алгоритму 11](#_Toc483395210)

[2.1.4 Метод Шара 12](#_Toc483395211)

[Трудоємність алгоритму 12](#_Toc483395212)

[3 АЛГОРИТМИ РОЗВ’ЯЗАННЯ ЗАДАЧІ 13](#_Toc483395213)

[Загальний алгоритм роботи програми 13](#_Toc483395214)

[Реалізація алгоритму лінійного пошуку 14](#_Toc483395215)

[Реалізація алгоритму бінарного пошуку 15](#_Toc483395216)

[Реалізація алгоритму однорідного бінарного пошуку 16](#_Toc483395217)

[Реалізація методу Шара 18](#_Toc483395218)

[4 ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 19](#_Toc483395219)

[Функціональна схема програмного забезпечення 19](#_Toc483395220)

[Опис функцій частин програмного забезпечення 19](#_Toc483395221)

[4.1.2 Функції користувача 20](#_Toc483395222)

[4.1.3 Стандартні функції, використані в програмі 22](#_Toc483395223)

[5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 23](#_Toc483395224)

[План тестування 23](#_Toc483395225)

[Приклади тестування 23](#_Toc483395226)

[6 ІНСТРУКЦІЯ КОРИСТУВАЧА 25](#_Toc483395227)

[Робота з програмою 25](#_Toc483395228)

[7 АНАЛІЗ І УЗАГАЛЬНЕННЯ РЕЗУЛЬТАТІВ 30](#_Toc483395229)

[ВИСНОВКИ 31](#_Toc483395230)

[СПИСОК ЛІТЕРАТУРИ 33](#_Toc483395231)

[ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ 34](#_Toc483395232)

[ДОДАТОК Б ТЕКСТИ ПРОГРАМНОГО КОДУ 37](#_Toc483395233)

[Лістинг програми (функціональної частини): 38](#_Toc483395234)

[files.h 38](#_Toc483395235)

[files.cpp 38](#_Toc483395236)

[generator.h 39](#_Toc483395237)

[generator.cpp 39](#_Toc483395238)

[libraries.h 40](#_Toc483395239)

[searching.h 40](#_Toc483395240)

[searching.cpp 41](#_Toc483395241)

[sorting.h 44](#_Toc483395242)

[sorting.cpp 44](#_Toc483395243)

[MyForm.h 45](#_Toc483395244)

ВСТУП

Комп'ютерні технології розвиваються дуже стрімко . Кожного дня з'являються нові задачі і кожного дня програмісти по всюму світу вирішують ці задачі . У теперішній час , люди оперують великим об'ємом даних і для швидкої роботи розробляються нові і нові алгоритми .

Пошук єлементів у будь – яких послідовностях об'єктів використовується майже в усіх програмах , які тільки можна собі уявити і в зв'язку із цим , розробляються різні методи виконання пошуку.

# ПОСТАНОВКА ЗАДАЧІ

**Розробити програмне забезпечення, що буде виконувати пошук заданого єлемента в масиві :**

* **послідовний метод**
* **бінарний метод;**
* **однорідний бінарний метод;**
* **метод Шара.**

**Вхідні дані:**

Діапазон чисел для генерування масиву унікальних єлементів , єлемент пошуку.

**Вихідні дані:**

Позиція єлемента пошуку і масиві , час роботи алгоритму , кількість ітерацій циклу.

# ТЕОРЕТИЧНА ЧАСТИНА

### Послідовний(лінійний) метод

## Опис лінійного пошуку

Алгоритм послідовного пошуку знаходження заданого значення довільної функції на деякому її відрізку. Цей алгоритм є найпростішим алгоритмом пошуку і на відміну, наприклад, від двійкового пошуку, не накладає жодних обмежень на функцію і має просту реалізацію.

Пошук значення функції здійснюється простим порівнянням чергового розглянутого значення (як правило пошук відбувається зліва направо, тобто від менших значень аргументу до більших , ітератор циклу при цьому змінюється за формулою **(2.1.1.0))**і, якщо значення збігаються (з тією або іншою точністю), то пошук вважається завершеним.

**Формули:**

**(2.1.1.0)**

## Трудомісткість алгоритму:

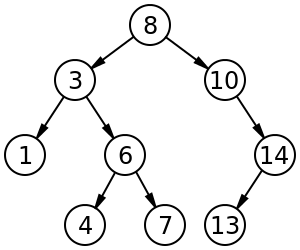
***-*** складність

***N* –** довжина відрізку

***𝜺 –*** точність

**-** час знаходження **(2.1.1.1)**

### Бінарний метод



## Опис бінарного методу

Суть даного методу полягає у порівнянні серединного елемента масиву з шуканим значенням, і повторенням алгоритму для тієї або іншої половини(використовується двійкове дерево пошуку )(Рисунок 2.1) , залежно від результату порівняння. В циклі ми розбиваємо наш відрізок на 2 частини і дивимось в якій із них знаходиться наш елемент , і долі вже працюємо з цією областю , так повторюємо до тих пір , поки не знайдемо елемент чи цикл не закінчить роботу.

Рисунок 2.1

На відміну від лінійного пошуку , бінарний метод змінює ітератор за формулою **(2.2.2.1)**

**Формули :**

**(2.2.2.1)**

## Трудомісткість алгоритму

**1+ -** складність

***n*** — кількість елементів у масиві

**(2.2.3.2)**

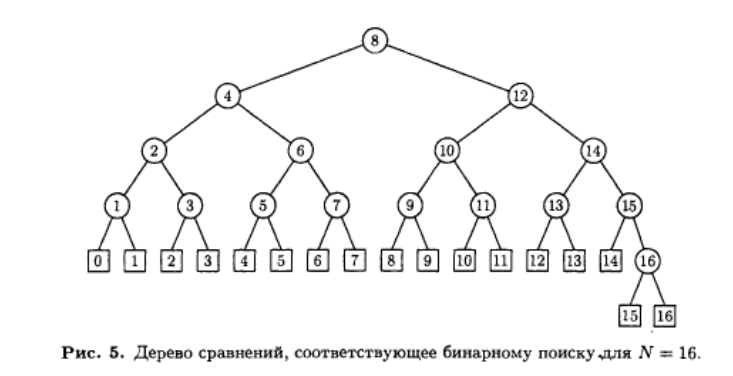


Рисунок 2.2 Дерево порівнянь пошуку бінарним методом

### Однорідний бінарний метод

## Опис однорідного бінарного методу

Ця модифікація бінарного пошуку потребує створення таблиці для збереження індексів. Ця таблиця демонструє можливі зміни індексу на кожному кроці пошуку.

Ми будуємо цю таблицю використовуючи формули:

**(2.1.3.1)**

Далі на кожному кроці ітерації , ми дістаємо індекс із таблиці і так отримуємо поточний індекс , який порівнюємо із елементом пошуку.

## Трудомісткість алгоритму

***-*** складність

***N*** – кількість елементів у масиві  **(2.1.3.2)**

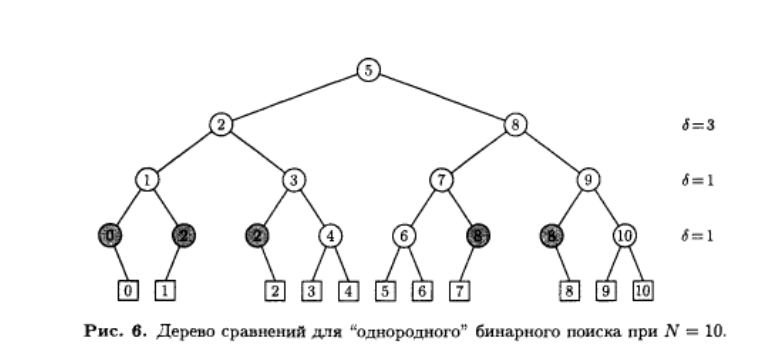


Рисунок 2.3 Дерево порівнянь пошуку однорідним бінарним методом

### Метод Шара

**Опис методу Шара**

Інша модифікація бінарного пошуку , запропонована Л.Е.Шаром , на деяких комп’ютерах вона буде працювати швидше , оскільки вона стає однорідною після першого кроку. Перший крок заключається в порівнянні елемента пошуку із i-им елементом масиву , де :

**(2.2.5)**

Якщо , ми записуємо в і використовуємо однорідний пошук. Якщо , ми записуємо в

і використовуємо бінарний пошук.

## Трудоємність алгоритму

***N* –** довжина відрізку **(2.2.6)**

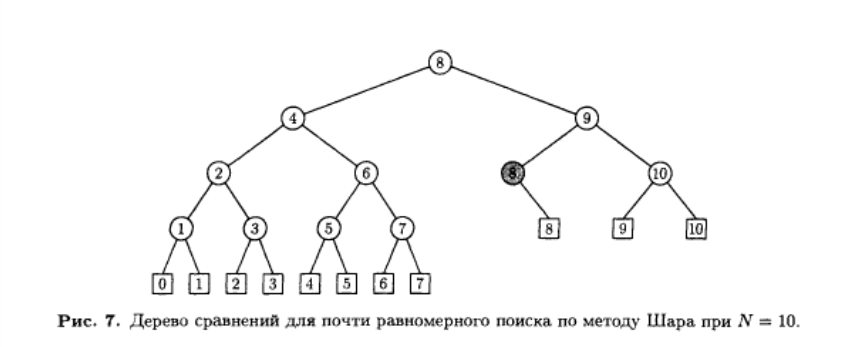


Рисунок 2.4 Дерево порівнянь для пошуку методом Шара

# АЛГОРИТМИ РОЗВ’ЯЗАННЯ ЗАДАЧІ

## Загальний алгоритм роботи програми

Початок

Задати діапазон чисел масиву

Перевірити довжину діапазону

Якщо діапазон менше 1000 елементів , то видати повідомлення про помилку

Якщо діапазон більше 1000 елементів продовжити роботу

Сгенерувати масив унікальних елементів із заданого діапазону

Задати метод пошуку елемента в масиві

Якщо вибрали лінійний пошук продовжити роботу

Якщо вибрали інші методи , відсортувати масив

Вибрати елемент пошуку

Викликати відповідну вибраному алгоритму функцію пошуку

Якщо функція повернула -1 , то повідомити про відсутність елементу пошуку в масиві

Вивести позицію елемента пошуку в масиві

Вивести інформацію про кількість ітерацій та час роботи певного алгоритму

Кінець

## Реалізація алгоритму лінійного пошуку

**Дані :**

array – показчик на масив

element – елемент пошуку

size – розмірність масиву

і – ітератор

**Алгоритм :**

1. **Початок**
2. Ініціалізуємо
3. Запускаємо цикл поки 
   1. **Якщо**
      1. Програма не знайшла елемент
      2. Повернути -1
      3. Завершити роботу
4. Програма знайшла елемент
5. Повернути його позицію
6. **Кінець**

## Реалізація алгоритму бінарного пошуку

**Дані :**

**array** – показчик на масив

**pointer** – позиція елемента в масиві

**element** – елемент пошуку

**size** – розмірнуість масиву

**start** – початок масиву

**finish** – кінець масиву

**Алгоритм :**

1. **Початок**
2. Ініціалізувати позицію елемента в масиві
3. Запускаємо цикл поки виконуються умови і і
   1. **Якщо**
      1. Ініціалізувати , ,
   2. **В іншому випадку** 
      1. Ініціалізувати , ,
4. **Якщо**
   1. Програма не знайшла елемент
   2. Повернути -1
   3. Завершити роботу
5. **В іншому випадку** 
   1. Програма знайшла елемент
   2. Повернути його позицію
6. **Кінець**

## Реалізація алгоритму однорідного бінарного пошуку

**Дані** **:**

**arr** – показчик на масив елементів

**c** – обчислюється за формулою **(2.2.3)**

**D** – показчик на масив , обчислюється за формулою **(2.2.3)**

**size** – розмірність масиву

**K** – елемент пошуку

**j** – позиція в масиві D

**i** – позиція в масиві **arr**

**Алгоритм** **:**

1. **Початок**
2. Ініціалізуємо змінну с за формулою **(2.1.3.1)**
3. Динамічно створити масив D із розмірністю с
4. Цикл з ітератором
   1. Ініціалізувати масив **D** за формулою **(2.1.3.1)**
5. Ініціалізувати
6. Цикл з ітератором
   1. **Якщо**
      1. **Якщо** і і
         1. Елемент знайдено
         2. Повертаємо позицію :
      2. **В іншому випадку** 
         1. Елемент не знайдено
         2. Повернути -1
         3. Завершити роботу
   2. **В іншому випадку**
      1. **Якщо**
         1. Елемент знайдено
         2. Повернути його позицію
      2. **В іншому випадку**
         1. **Якщо**
            1. Ініціалізувати
         2. **В іншому випадку**
            1. Ініціалізувати
   3. **Якщо** і і
      1. Програма не знайшла елемент
      2. Повернути -1
      3. Завершити роботу
   4. **В іншому випадку** 
      1. Програма знайшла елемент
      2. Повернути його позицію
7. **Кінець**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **j** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | 500 | 250 | 125 | 63 | 31 | 16 | 8 | 4 | 2 | 1 |

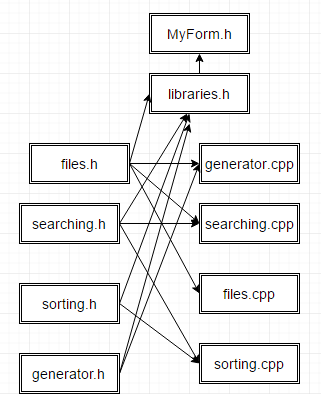
Елементи масиву D

## Реалізація методу Шара

# ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Функціональна схема програмного забезпечення

Програмний продукт містить головний файл, 4 файли реалізації, 5 заголовних файлів, заголовний файл та файл реалізації інтерфейсу та імпортований модуль відображення графіків функцій системи рівнянь.



#### : Структура програмного забезпечення

## Опис функцій частин програмного забезпечення

Програма поділена на наступні модулі:

**files.h** – реалізує обмін даними в програмі через файли

**generator.h** – відповідає за генерацію 1000 випадкових унікальних чисел

**searching.h** – відповідає за методи пошуку елемента в масиві

**sorting.h** – відповідає за сорутвання елементів у масиві

**libraries.h** – включає в себе всі бібліотеки використані в программі

**MyForm.h** – реалізує взаємодію з користувачем

### Функції користувача

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Назва функції** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** | **Заголовний файл** |
| 1 | InFile | Запис масиву у файл | Покажчик на масив і розмірність масиву | - | files.h |
| 2 | *FromFile* | Зчитування масиву з файлу | Покажчик на масив і розмірність масиву | - | files.h |
| 3 | *selectedIn* | Запис номера комірки у файл | Номер виділеної комірки | - | files.h |
| 4 | *selectedOut* | Зчитування номеру комірки з файлу | - | Номер виділеної комірки | files.h |
| 5 | *IterationIn* | Запис кількості ітерацій у файл | Кількість ітерацій | - | files.h |
| 6 | *IterationOut* | Зчитування кількості ітерацій з файлу | - | Кількість ітерацій | files.h |
| 7 | *Linear* | Лінійний пошук для генератора | Покажчик на масив , елемент пошуку , розмірність | Позицію знайденого елемента , чи -1 у випадку відсутності такого елемента | generator.h |
| 8 | *generate* | Генератор випадкових унікальних чисел | Покажчик на масив , межі генерування(з, по) | - | generator.h |
| 9 | *Binary* | Бінарний пошук | Покажчик на масив , елемент пошуку , розмірність , межі масиву(як правило 0 і розмірність -1) | Позицію знайденого елемента , чи -1 у випадку відсутності такого елемента | *searching.h* |
| 10 | *ABinary* | Однорідний бінарний пошук | Покажчик на масив , елемент пошуку , розмірність | Позицію знайденого елемента , чи -1 у випадку відсутності такого елемента | *searching.h* |
| 11 | *linear* | Лінійний пошук | Покажчик на масив , елемент пошуку , розмірність | Позицію знайденого елемента , чи -1 у випадку відсутності такого елемента | *searching.h* |
| 12 | *Shar* | Метод Шара | Покажчик на масив , елемент пошуку , розмірність | Позицію знайденого елемента , чи -1 у випадку відсутності такого елемента | *searching.h* |
| 13 | *buble* | Сортування масиву | Покажчик на масив , розмірність | - | *sorting.h* |

### Стандартні функції, використані в програмі

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № п/п | Назва функції | Призначення функції | Опис вхідних параметрів | Опис вихідних параметрів | Заголовний файл |
| 1 | *clock()* | Отримати час роботи функції | - | Повертає час пройдений із моменту запуску программи | time.h |
| 2. | *log2* | Обчислення двійкового логарифму числа | Дійсне число | Значення двійкового логарифм учисла | math.h |
| 3. | *abs* | Обчислення абсолютного значення числа | Дійсне число | Абсолютне значення числа | math.h |
|  | *toString* | Переводить дані в рядок | Число/рядок/символ | Рядок | String |
| 5. | *toInt32* | Переводить рядок в ціле число, якщо це можливо | Рядок | «Так» і ціле число, якщо дія можлива, інакше «ні» | Int |

# ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## План тестування

*Тестування реакції на неправильні введені значення*

* 1. Тестування при введенні діапазону із менше ніж 1000 елементів значень (Рисунок 6.4 1)
  2. Тестування при введенні елемента, якого не має в масиві (Рисунок 6.4 1)
  3. Тестування при не виборі методу пошуку(Рисунок 6.4 1)

## Приклади тестування

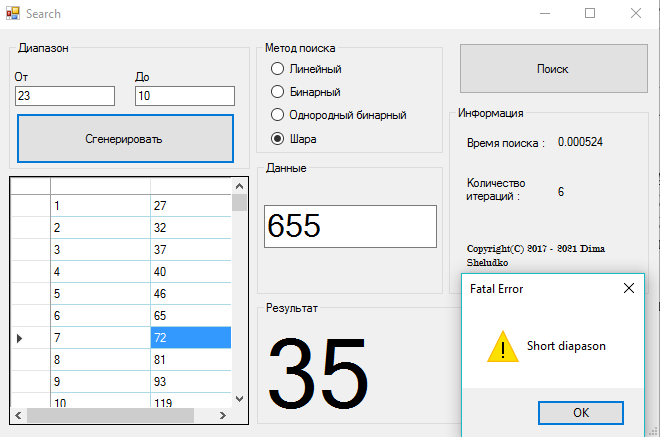
****

Рисунок 6.4 1

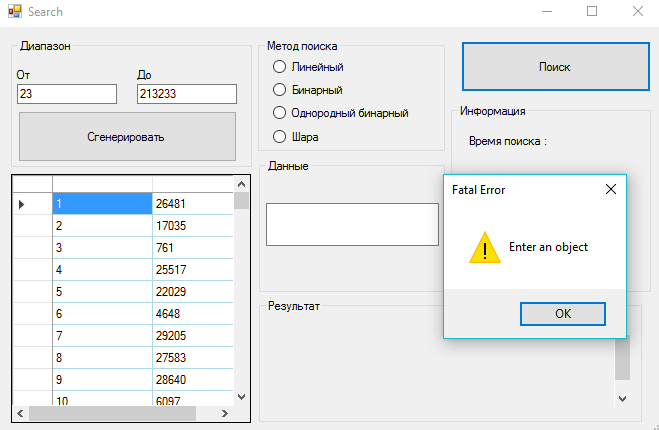
****

Рисунок 6.4 2

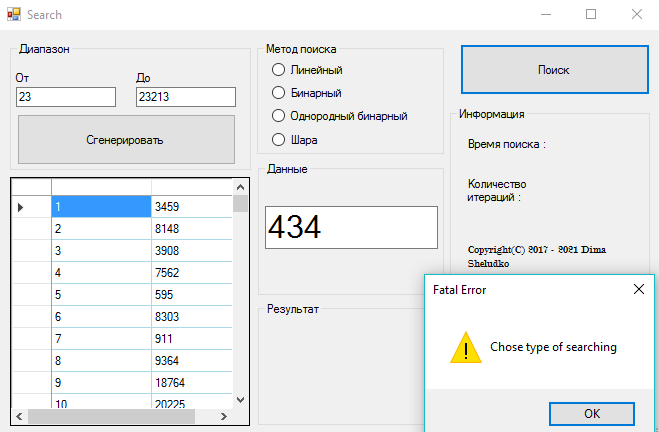
****

Рисунок 6.4 3

# ІНСТРУКЦІЯ КОРИСТУВАЧА

## Робота з програмою

Після запуску користувачем виконавчого файлу програми, на екрані комп’ютера з’явиться головне вікно програми (рисунок 6.1).

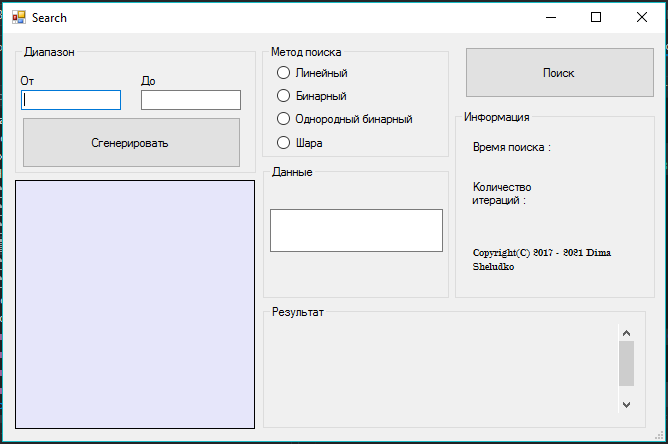


Рисунок 6.4 4

Далі потрібно ввести діапазон для генерування масиву.Після введення діапазону, користувач має натиснути на кнопку «Сгенерировать». Після чого буде виведено масив у вигляді (рисунок 6.2). Тепер потрібно вибрати бажаний метод пошуку. У полі дані потрібно ввести елемент пошуку , після чого натиснути кнопку «Поиск» і ми отримаємо результат пошуку : або номер елементу в таблиці(рисунок 6.3), де його буде також виділено , або сповіщення про відсутність діного елементу в таблиці(рисунок 6.4)

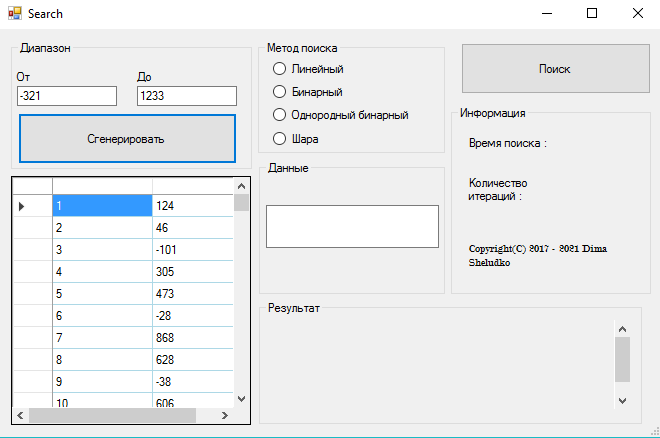


Рисунок 6.4 5

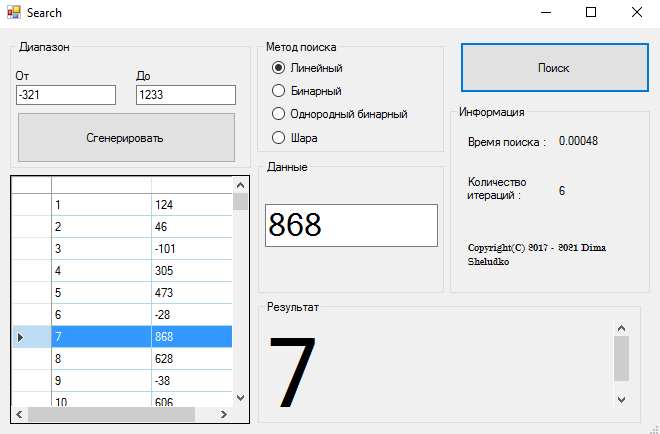


Рисунок 6.4 6

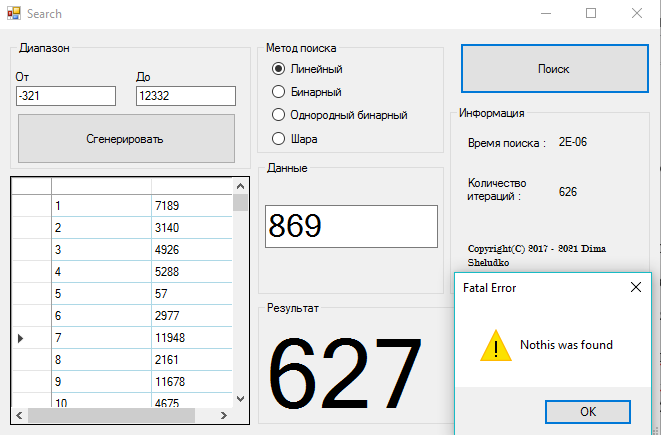


Рисунок 6.4 7

Користувач має змогу скористатися одним із 4 запропонованих методів : лінійний пошук (Рисунок 6.3) , бінарний пошук (Рисунок 6.5) , однорідний бінарний (Рисунок 6.6) , метод Шара (Рисунок 6.7)

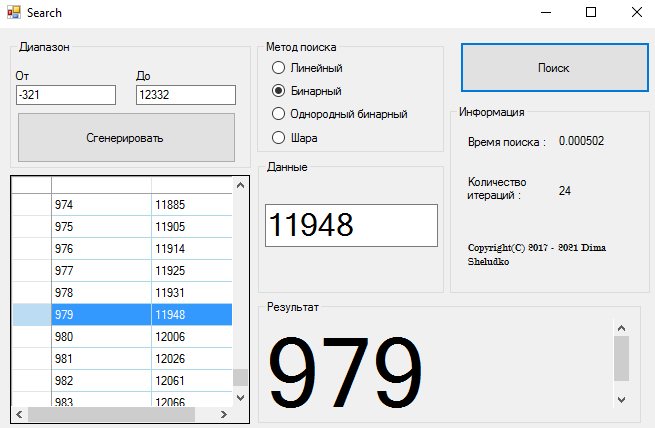


Рисунок 6.4 8

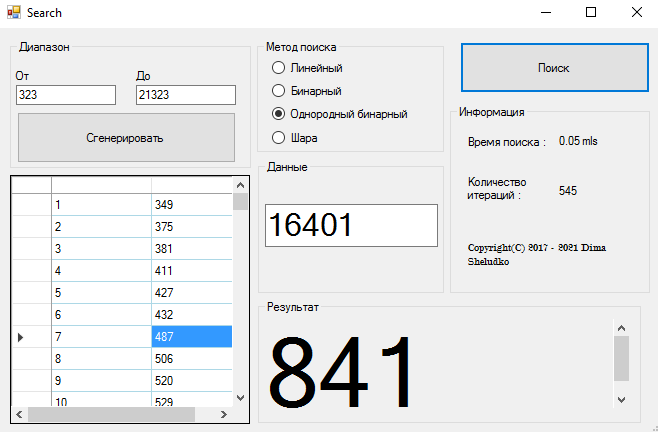


Рисунок 6.4 9

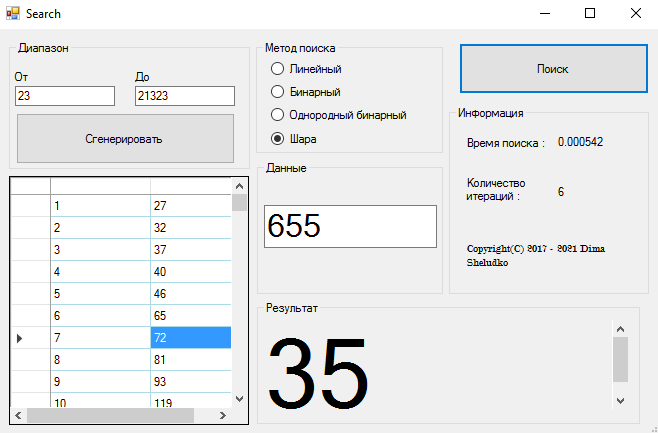


Рисунок 6.4 10

У меню справа «Інформація» программа показує інформацію про кожен із алгоритмів.

Для закінчення роботи програми користувач має натиснути червону кнопку в правому верхньому куті, після чого головне вікно закриється.

# АНАЛІЗ І УЗАГАЛЬНЕННЯ РЕЗУЛЬТАТІВ

Головною задачею при виконанні курсової роботи було створення програми для пошуку заданого елемента в масиві.

Програма може перейти у помилковий стан (будуть виведені повідомлення про помилковий стан програми) лише у разі некоректного вводу вхідних даних. Було розглянуто випадки введення користувачем некоректного діапазону, незадання елементу пошуку , невибір методу пошуку. Інші ж критичні стани програми були усунені під час розробки програми.

**Порівняння алгоритмів**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Метод | Лінійний пошук | Бінарний пошук | Однорідний бінарний пошук | Метод Шара |
| Дані | 3414 | 3414 | 3414 | 3414 |
| Час | 0.488 mls | 0.524 mls | 0.036 mls | 0.542 mls |
| Кількість ітерацій | 187 | 3 | 19 | 19 |

# ВИСНОВКИ

При ознайомленні з теоретичною основою цієї курсової роботи досліджено алгоритми ітеративного розв’язання систем нелінійних рівнянь, що використовуються для моделювання фізичних та інших процесів, проаналізовано їх як модулі для використання в межах більших проектів з реальним застосуванням. Таке використання було доведено реалізацією програмного забезпечення, яке розглянуто у даній курсовій роботі.

В створеному програмному забезпеченні реалізовано методи Якобі (простої ітерації) та Гауса-Зейделя. Результатом такої роботи стало набуття досвіду роботи з чисельними методами в програмуванні, використання та покращення знання мови програмування С++, та набуття практичних навичок її використання з точки зору процедурного та модульного філософій програмування.

Було проведено тестування, під час якого було визначені критичні стани, та написаний код обробки дій користувача, які призводили до переходу до них. Також за допомогою обчислень вручну та за допомогою стороннього програмного забезпечення було виконано аналіз та звірення отриманих результатів з відомо правильними, чим підтверджується достовірність і очікуваність результатів роботи програми.

Окрім реалізації власне алгоритмів, включених до задачі, та проведення тестування програмного забезпечення, було розроблено простий користувацький інтерфейс. Це збільшує коло людей, які можуть її використовувати, полегшуючи роботу з ними. Таке саме призначення має розроблена інструкція користувача, чітко пояснююча принципи взаємодії з програмним забезпеченням. Графічний інтерфейс розроблено за допомогою кросплатформеного тулкіту Qt, який дозволяє збирати та запускати проект майже під будь якою програмною або апаратною платформою, наприклад, під операційними системами FreeBSD та Linux, на комп’ютерах з процесорами архітектури ARM, і так далі. Цей факт також помітно розширює область застосування розробленого програмного забезпечення.

Результат цієї курсової роботи – готове для використання в навчальних цілях програмне забезпечення, яке реалізує розв’язок систем нелінійних рівнянь певного виду ітераційними методами Якобі та Гауса-Зейделя.

# СПИСОК ЛІТЕРАТУРИ

1. Дональд Е. Кнут “*Искусство программирования. Том 3*”.
2. Вірт Н. (1985), *Алгоритми та структури даних*, розділ 1.9.2.
3. [*Левитин А. В.*](https://www.wikidata.org/wiki/Q21694518) Глава 4. Метод декомпозиции: Бинарный поиск // [Алгоритмы. Введение в разработку и анализ](https://www.wikidata.org/wiki/Q21694522) — М.: [Вильямс](https://www.wikidata.org/wiki/Q21694521), 2006. — С. 180–183. — 576 с. — [ISBN 978-5-8459-0987-9](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785845909879)
4. *Амосов А. А., Дубинский Ю. А., Копченова Н. П.* Вычислительные методы для инженеров. — М.: Мир, 1998.
5. *Бахвалов Н. С.,*[*Жидков.*](https://ru.wikipedia.org/wiki/%D0%96%D0%B8%D0%B4%D0%BA%D0%BE%D0%B2,_%D0%9D%D0%B8%D0%BA%D0%BE%D0%BB%D0%B0%D0%B9_%D0%9F%D0%B5%D1%82%D1%80%D0%BE%D0%B2%D0%B8%D1%87)*, Кобельков Г. Г.* Численные методы. — 8-е изд. — М.: Лаборатория Базовых Знаний, 2000.
6. *Вирт Н.* Алгоритмы + структуры данных = программы. — М.: «[Мир](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D1%80_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE))», 1985. — С. 28.
7. *Волков Е. А.* Численные методы. — М.: Физматлит, 2003.
8. *Гилл Ф., Мюррей У., Райт М.* Практическая оптимизация. Пер. с англ. — М.: Мир, 1985.
9. [*Кормен, Т.*](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D0%BC%D0%B5%D0%BD,_%D0%A2%D0%BE%D0%BC%D0%B0%D1%81)*,*[*Лейзерсон, Ч.*](https://ru.wikipedia.org/wiki/%D0%9B%D0%B5%D0%B9%D0%B7%D0%B5%D1%80%D1%81%D0%BE%D0%BD,_%D0%A7%D0%B0%D1%80%D0%BB%D1%8C%D0%B7_%D0%AD%D1%80%D0%B8%D0%BA)*,*[*Ривест, Р.*](https://ru.wikipedia.org/wiki/%D0%A0%D0%B8%D0%B2%D0%B5%D1%81%D1%82,_%D0%A0%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%B4_%D0%9B%D0%B8%D0%BD%D0%BD)*,*[*Штайн, К.*](https://ru.wikipedia.org/wiki/%D0%A8%D1%82%D0%B0%D0%B9%D0%BD,_%D0%9A%D0%BB%D0%B8%D1%84%D1%84%D0%BE%D1%80%D0%B4) Алгоритмы: построение и анализ = Introduction to Algorithms / Под ред. И. В. Красикова. — 2-е изд. — М.: Вильямс, 2005. — 1296 с. — [ISBN 5-8459-0857-4](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5845908574).
10. *Корн Г., Корн Т.* Справочник по математике для научных работников и инженеров. — М.: Наука, 1970. — С. 575-576.
11. *Коршунов Ю. М., Коршунов Ю. М.* Математические основы кибернетики. — Энергоатомиздат, 1972.
12. *Максимов Ю. А., Филлиповская Е. А.* Алгоритмы решения задач нелинейного программирования. — М.: МИФИ, 1982.
13. *Роберт Седжвик.* Фундаментальные алгоритмы на C. Анализ/Структуры данных/Сортировка/Поиск = Algorithms in C. Fundamentals/Data Structures/Sorting/Searching. — СПб.: ДиаСофтЮП, 2003. — С. 672. — [ISBN 5-93772-081-4](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5937720814).

# ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ

МІНІСТЕРСТВО НАУКИ І ОСВІТИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КПІ»

Кафедра

автоматизованих систем обробки інформації та управління

Затвердив

Керівник Головченко М.М.

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2016 р.

Виконавець:

Студент Шелудько Д.М. *\_\_\_\_*

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2016 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання курсової роботи

на тему: Розв'язання систем нелінійних рівнянь

з дисципліни:

«Основи програмування»

Київ 2016

1.1 Мета: Метою курсової роботи є розробка комплексу програм для *знаходження елемента масиву*

1.2 Найменування та галузь застосування об'єкта розробки:Дана робота присвячена розробці програмного забезпечення для використання в банківській справі , математиці , фізиці , програмуванні.

1.3 Підстава для проведення робіт:Підставою для розробки програмного забезпечення є навчальний план спеціальності 6.050301 «Програмна інженерія», робоча програма дисципліни „Основи програмування”, індивідуальне завдання.

1.4 Дата початку роботи: «6 » березня 2016 р.

1.5 Дата закінчення роботи: «23»» травня 2016 р.

1.6 Призначення розробки: Функціональним призначенням програми є пошук заданого елемента масиву лінійним методом , бінарним методом , однорідним бінарним методом , методом Шара . Експлуатаційним призначенням програми є швидкий пошук елементів у масивах даних.

1.7 Вимоги до програми та програмної документації: Все програмне забезпечення та супроводжуюча технічна документація повинні задовольняти наступним ГОСТ:

ИСО 5807 - 85 ГОСТ на розробку програмних документів, схем алгоритмів програм, даних та систем.

ГОСТ 19.781 - 74 - Вимоги до розробки програмного забезпечення.

ГОСТ 19.101-77 (СТ СЭВ 1626 - 79) - Держстандарт на розробку програмної документації, видів програм та програмних документів.

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 - 95 - Розробка технічної документації.

1.8 Стадії та етапи розробки:

1. Аналіз методів вирішення поставленої задачі (до 17.04.2016 р.)

2. Розробка сценарію роботи програми (до 01.05.2016 р.)

3. Розробка алгоритмічного забезпечення (до 24.04.2016 р.)

4. Розробка програмного забезпечення (до 09.05.2016 р.)

5. Розробка інтерфейсу, планування, тестування розробленої програми (до 16.05.2016 р.)

6. Розробка пояснювальної записки (до 25.05.2016 р.).

7. Захист курсової роботи (до 12.06.2016 р.).

1.9 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 її оцінювання

# ДОДАТОК Б ТЕКСТИ ПРОГРАМНОГО КОДУ

*Тексти програмного коду програмного забезпечення*

«Пошук заданого елемента в масиві»

(Вид носія даних)

*DVD-RW*

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

*27 арк,60 Кb*

*студента групи ІП-63 І курсу*

Шелудька Дмитра Максимовича

## Лістинг програми (функціональної частини):

## files.h

#pragma once

void InFile(int \*&array, int size);

void FromFile(int \*&array, int size);

void selectedIn(int number);

int selectedOut();

void IterationIn(int sum);

int IterationOut();

## files.cpp

#pragma once

#include <fstream>

#include <iostream>

using namespace std;

void InFile(int \*&array, int size)

{

ofstream out;

out.open("data.info");

for (int i = 0; i<size - 1; i++)

{

out << array[i] << " ";

}

out << array[size - 1];

}

void FromFile(int \*&array, int size)

{

ifstream in;

in.open("data.info");

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

{

in >> array[i];

}

}

void selectedIn(int number)

{

ofstream out;

out.open("selected.data");

out << number;

}

int selectedOut()

{

ifstream out("selected.data");

int result;

out >> result;

return result;

}

void IterationIn(int sum)

{

ofstream out;

out.open("iteration.data");

out << sum;

}

int IterationOut()

{

ifstream in("iteration.data");

int result = 0;

in >> result;

return result;

}

## generator.h

#pragma once

#define SIZE 1000

namespace Generator

{

int Linear(int \*array, int element, int size);

void generate(int \*&array, int from, int to);

}

## generator.cpp

#include <stdlib.h>

#include <ctime>

#include <time.h>

#include "files.h"

#include "generator.h"

//#define SIZE 1000

using namespace std;

namespace Generator

{

int Linear(int \*array, int element, int size)

{

int result = 0;

int iteration = 0;

while (array[result] != element)

{

iteration++;

result++;

if (result > size)

{

return -1;

}

}

IterationIn(iteration);

return result;

}

void generate(int \*&array, int from, int to)

{

//int SIZE = abs(to) - abs(from);

srand(time(0));

array = new int[SIZE];

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

{

array[i] = 0;

}

int curr = 0;

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

{

curr = rand() % (from - to + 1) + from;

while (Linear(array, curr, SIZE) != -1)

{

curr = rand() % (from - to + 1) + from;

}

array[i] = curr;

}

}

}

## libraries.h

#pragma once

#include <iostream>

#include <stdlib.h>

#include <cstdlib>

#include <stdio.h>

#include "files.h"

#include "sorting.h"

#include <ctime>

#include <time.h>

#include "searching.h"

#include "generator.h"

## searching.h

#pragma once

namespace Searching

{

int binary(int \*array, int element, int size, int start, int finish);

int Binary(int \*&array, int element, int sizeSS, int start, int finish);

int UBinary(int \*&array, int K, int i, int m);

int linear(int \*array, int element, int size);

int ubinary(int array[], int K, int i, int m);

int Alone(int \*&array, int K, int N);

int ABinary(int \*&array, int K, int size);

int Shar(int \*&array, int K, int N);

int Shara(int \*&array, int K, int n);

}

## searching.cpp

#include <math.h>

#include "files.h"

#include "searching.h"

namespace Searching

{

int Binary(int \*&array, int element, int size, int start, int finish)

{

int pointer = size / 2;//right

int iteration = 0;

while (pointer <= finish && pointer > 0 && array[pointer] != element)

{

iteration++;

if (array[pointer] > element)

{

size = size / 2;

finish = (pointer - 1);

pointer = (finish - start + 1) / 2;

}

else

{

start = pointer;

size = finish - pointer + 1;

pointer = size / 2 + start;

}

}

IterationIn(iteration);

if (array[pointer] != element)

{

return -1;

}

else

{

return pointer;

}

}

int linear(int \*array, int element, int size)

{

int result = 0;

int iteration = 0;

while (array[result] != element)

{

iteration++;

result++;

if (result > size)

{

return -1;

}

}

IterationIn(iteration);

return result;

}

int ABinary(int \*&array, int K, int size)

{

int \*arr = new int[size + 1];

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

{

arr[i] = array[i - 1];

}

int c = log(size) + 4;

int \*D = new int[c];

D[0] = 0;

for (int j = 1; j < log(size) + 4; j++)

{

D[j] = (size + pow(2, j - 1)) / pow(2, j);

//cout << " j :" << j << ";" << D[j] << endl;

}

//D[9] = 2;

//D[10] = 1;

int i = D[1], j = 2;

while (i > 0 && i < size)

{

if (D[j] == 0)

{

if (i <= 10 && i >0 && D[i] == K)

return i - 1;

else

return -1;

}

else

{

if (arr[i] == K)

{

return i - 1;

}

else

{

if (arr[i] > K)

{

i = i - D[j];

j = j + 1;

}

else

{

i = i + D[j];

j = j + 1;

}

}

}

}

if (i > 0 && i < size && arr[i] != K)

{

return -1;

}

else

{

return i - 1;

}

}

int Shar(int \*&array, int K, int N)

{

int k = log(N);

int I = pow(2, k);

int sigma = I;

if (K < array[I])

{

int size = I;

int \*arr = new int[size + 1];

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

{

arr[i] = array[i - 1];

}

int c = log(size) + 4;

int \*D = new int[c];

D[0] = 0;

for (int j = 1; j < log(size) + 4; j++)

{

D[j] = (size + pow(2, j - 1)) / pow(2, j);

//cout << " j :" << j << ";" << D[j] << endl;

}

//D[9] = 2;

//D[10] = 1;

int i = D[1], j = 2;

while (i > 0 && i < size)

{

if (D[j] == 0)

{

if (i <= 10 && i >0 && D[i] == K)

return i - 1;

else

return -1;

}

else

{

if (arr[i] == K)

{

return i - 1;

}

else

{

if (arr[i] > K)

{

i = i - D[j];

j = j + 1;

}

else

{

i = i + D[j];

j = j + 1;

}

}

}

}

if (i > 0 && i < size && arr[i] != K)

{

return -1;

}

else

{

return i - 1;

}

}

else

{

int element = K;

int pointer = N / 2;//right

int iteration = 0;

int size = N / 2;

int start = I;

int finish = N - 1;

while (pointer <= finish && pointer > 0 && array[pointer] != element)

{

if (array[pointer] > element)

{

int size = size / 2;

finish = (pointer - 1);

pointer = (finish - start + 1) / 2;

}

else

{

start = pointer;

size = finish - pointer + 1;

pointer = size / 2 + start;

}

}

if (array[pointer] != element)

{

return -1;

}

else

{

return pointer;

}

}

}

## sorting.h

#pragma once

namespace Sorting

{

void buble(int \*&array, int size);

}

## sorting.cpp

#include "sorting.h"

using namespace std;

namespace Sorting

{

void buble(int \*&array, int size)

{

int curr = 0;

int c = 0;

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

{

curr = array[i];

for (int j = i; j < size; j++)

{

if (curr > array[j])

{

curr = array[j];

c = array[i];

array[i] = curr;

array[j] = c;

}

}

}

}

}

## MyForm.h

#pragma once

#include "libraries.h"

/\*#include "files.h"

#include <iostream>

#include <ctime>

#include <time.h>

#include "searching.h"\*/

#define N 500 //for timing

//using namespace Searching;

namespace Cursach {

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 MyForm

/// </summary>

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

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::GroupBox^ gbGenerate;

protected:

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

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

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

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

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

private: System::Windows::Forms::GroupBox^ groupBox1;

private: System::Windows::Forms::RadioButton^ radioButton4;

private: System::Windows::Forms::RadioButton^ radioButton3;

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

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

private: System::Windows::Forms::GroupBox^ groupBox2;

private: System::Windows::Forms::Label^ label3;

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

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

private: System::Windows::Forms::GroupBox^ groupBox3;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::GroupBox^ groupBox4;

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

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

private: System::Windows::Forms::DataGridView^ Table;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Label^ label10;

protected:

protected:

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->gbGenerate = (gcnew System::Windows::Forms::GroupBox());

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

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

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

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

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

this->groupBox1 = (gcnew System::Windows::Forms::GroupBox());

this->radioButton4 = (gcnew System::Windows::Forms::RadioButton());

this->radioButton3 = (gcnew System::Windows::Forms::RadioButton());

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

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

this->groupBox2 = (gcnew System::Windows::Forms::GroupBox());

this->label10 = (gcnew System::Windows::Forms::Label());

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

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

this->label5 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

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

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

this->groupBox3 = (gcnew System::Windows::Forms::GroupBox());

this->groupBox4 = (gcnew System::Windows::Forms::GroupBox());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->Table = (gcnew System::Windows::Forms::DataGridView());

this->gbGenerate->SuspendLayout();

this->groupBox1->SuspendLayout();

this->groupBox2->SuspendLayout();

this->groupBox3->SuspendLayout();

this->groupBox4->SuspendLayout();

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

this->SuspendLayout();

//

// gbGenerate

//

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

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

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

this->gbGenerate->Controls->Add(this->txtTo);

this->gbGenerate->Controls->Add(this->txtFrom);

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

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

this->gbGenerate->Size = System::Drawing::Size(241, 129);

this->gbGenerate->TabIndex = 0;

this->gbGenerate->TabStop = false;

this->gbGenerate->Text = L"Диапазон";

//

// button1

//

this->button1->Location = System::Drawing::Point(7, 72);

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

this->button1->Size = System::Drawing::Size(219, 51);

this->button1->TabIndex = 5;

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

this->button1->UseVisualStyleBackColor = true;

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

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(123, 29);

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

this->label2->Size = System::Drawing::Size(22, 13);

this->label2->TabIndex = 3;

this->label2->Text = L"До";

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(3, 29);

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

this->label1->Size = System::Drawing::Size(20, 13);

this->label1->TabIndex = 2;

this->label1->Text = L"От";

//

// txtTo

//

this->txtTo->Location = System::Drawing::Point(126, 45);

this->txtTo->Multiline = true;

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

this->txtTo->Size = System::Drawing::Size(100, 20);

this->txtTo->TabIndex = 1;

//

// txtFrom

//

this->txtFrom->Location = System::Drawing::Point(6, 45);

this->txtFrom->Multiline = true;

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

this->txtFrom->Size = System::Drawing::Size(100, 20);

this->txtFrom->TabIndex = 0;

//

// groupBox1

//

this->groupBox1->Controls->Add(this->radioButton4);

this->groupBox1->Controls->Add(this->radioButton3);

this->groupBox1->Controls->Add(this->radioButton2);

this->groupBox1->Controls->Add(this->radioButton1);

this->groupBox1->Location = System::Drawing::Point(259, 12);

this->groupBox1->Name = L"groupBox1";

this->groupBox1->Size = System::Drawing::Size(187, 113);

this->groupBox1->TabIndex = 2;

this->groupBox1->TabStop = false;

this->groupBox1->Text = L"Метод поиска";

//

// radioButton4

//

this->radioButton4->AutoSize = true;

this->radioButton4->Location = System::Drawing::Point(15, 89);

this->radioButton4->Name = L"radioButton4";

this->radioButton4->Size = System::Drawing::Size(52, 17);

this->radioButton4->TabIndex = 3;

this->radioButton4->TabStop = true;

this->radioButton4->Text = L"Шара";

this->radioButton4->UseVisualStyleBackColor = true;

//

// radioButton3

//

this->radioButton3->AutoSize = true;

this->radioButton3->Location = System::Drawing::Point(15, 65);

this->radioButton3->Name = L"radioButton3";

this->radioButton3->Size = System::Drawing::Size(142, 17);

this->radioButton3->TabIndex = 2;

this->radioButton3->TabStop = true;

this->radioButton3->Text = L"Однородный бинарный";

this->radioButton3->UseVisualStyleBackColor = true;

//

// radioButton2

//

this->radioButton2->AutoSize = true;

this->radioButton2->Location = System::Drawing::Point(15, 42);

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

this->radioButton2->Size = System::Drawing::Size(76, 17);

this->radioButton2->TabIndex = 1;

this->radioButton2->TabStop = true;

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

this->radioButton2->UseVisualStyleBackColor = true;

//

// radioButton1

//

this->radioButton1->AutoSize = true;

this->radioButton1->Location = System::Drawing::Point(15, 19);

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

this->radioButton1->Size = System::Drawing::Size(77, 17);

this->radioButton1->TabIndex = 0;

this->radioButton1->TabStop = true;

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

this->radioButton1->UseVisualStyleBackColor = true;

//

// groupBox2

//

this->groupBox2->Controls->Add(this->label10);

this->groupBox2->Controls->Add(this->Iteration);

this->groupBox2->Controls->Add(this->TIME);

this->groupBox2->Controls->Add(this->label5);

this->groupBox2->Controls->Add(this->label3);

this->groupBox2->Location = System::Drawing::Point(452, 77);

this->groupBox2->Name = L"groupBox2";

this->groupBox2->Size = System::Drawing::Size(200, 189);

this->groupBox2->TabIndex = 3;

this->groupBox2->TabStop = false;

this->groupBox2->Text = L"Информация";

//

// label10

//

this->label10->Font = (gcnew System::Drawing::Font(L"Modern No. 20", 8.249999F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->label10->Location = System::Drawing::Point(15, 135);

this->label10->Name = L"label10";

this->label10->Size = System::Drawing::Size(176, 30);

this->label10->TabIndex = 7;

this->label10->Text = L"Copyright(C) 2017 - 2021 Dima Sheludko";

//

// Iteration

//

this->Iteration->Location = System::Drawing::Point(106, 79);

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

this->Iteration->Size = System::Drawing::Size(74, 14);

this->Iteration->TabIndex = 5;

//

// TIME

//

this->TIME->Location = System::Drawing::Point(106, 29);

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

this->TIME->Size = System::Drawing::Size(74, 14);

this->TIME->TabIndex = 3;

//

// label5

//

this->label5->Location = System::Drawing::Point(15, 70);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(85, 48);

this->label5->TabIndex = 2;

this->label5->Text = L"Количество итераций :";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(15, 30);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(85, 13);

this->label3->TabIndex = 0;

this->label3->Text = L"Время поиска :";

//

// button2

//

this->button2->Location = System::Drawing::Point(462, 14);

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

this->button2->Size = System::Drawing::Size(190, 51);

this->button2->TabIndex = 4;

this->button2->Text = L"Поиск";

this->button2->UseVisualStyleBackColor = true;

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

//

// textBox3

//

this->textBox3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 24, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->textBox3->Location = System::Drawing::Point(7, 44);

this->textBox3->Multiline = true;

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

this->textBox3->Size = System::Drawing::Size(173, 43);

this->textBox3->TabIndex = 6;

//

// groupBox3

//

this->groupBox3->Controls->Add(this->textBox3);

this->groupBox3->Location = System::Drawing::Point(260, 132);

this->groupBox3->Name = L"groupBox3";

this->groupBox3->Size = System::Drawing::Size(186, 134);

this->groupBox3->TabIndex = 7;

this->groupBox3->TabStop = false;

this->groupBox3->Text = L"Данные";

//

// groupBox4

//

this->groupBox4->Controls->Add(this->richTextBox1);

this->groupBox4->Location = System::Drawing::Point(260, 272);

this->groupBox4->Name = L"groupBox4";

this->groupBox4->Size = System::Drawing::Size(383, 124);

this->groupBox4->TabIndex = 8;

this->groupBox4->TabStop = false;

this->groupBox4->Text = L"Результат";

//

// richTextBox1

//

this->richTextBox1->BackColor = System::Drawing::SystemColors::Control;

this->richTextBox1->BorderStyle = System::Windows::Forms::BorderStyle::None;

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 72, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->richTextBox1->Location = System::Drawing::Point(7, 19);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->Size = System::Drawing::Size(365, 89);

this->richTextBox1->TabIndex = 0;

this->richTextBox1->Text = L"";

//

// Table

//

this->Table->AllowUserToAddRows = false;

this->Table->AllowUserToDeleteRows = false;

this->Table->AllowUserToResizeColumns = false;

this->Table->AllowUserToResizeRows = false;

this->Table->BackgroundColor = System::Drawing::Color::Lavender;

this->Table->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->Table->GridColor = System::Drawing::Color::LightBlue;

this->Table->Location = System::Drawing::Point(12, 147);

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

this->Table->Size = System::Drawing::Size(240, 249);

this->Table->TabIndex = 9;

//

// MyForm

//

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

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

this->ClientSize = System::Drawing::Size(662, 408);

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

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

this->Controls->Add(this->groupBox4);

this->Controls->Add(this->groupBox3);

this->Controls->Add(this->groupBox2);

this->Controls->Add(this->groupBox1);

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

this->MaximumSize = System::Drawing::Size(678, 447);

this->MinimumSize = System::Drawing::Size(678, 447);

this->Name = L"MyForm";

this->SizeGripStyle = System::Windows::Forms::SizeGripStyle::Show;

this->Text = L"Search";

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

this->gbGenerate->ResumeLayout(false);

this->gbGenerate->PerformLayout();

this->groupBox1->ResumeLayout(false);

this->groupBox1->PerformLayout();

this->groupBox2->ResumeLayout(false);

this->groupBox2->PerformLayout();

this->groupBox3->ResumeLayout(false);

this->groupBox3->PerformLayout();

this->groupBox4->ResumeLayout(false);

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

this->ResumeLayout(false);

}

#pragma endregion

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

}

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

}

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

{

//DataGridViewRow row;

int from = 0, to = 0;

if (txtFrom->Text == "" || txtTo->Text == "")

{

MessageBox::Show("Enter a diapason", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else

{

from = Convert::ToInt32(txtFrom->Text);

to = Convert::ToInt32(txtTo->Text);

if (to < from)

{

MessageBox::Show("Short diapason", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else

{

if (abs(to - from) > 1000)

{

int \*array = 0;

Generator::generate(array, from, to);

Table->RowCount = SIZE;

Table->ColumnCount = 2;

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

{

Table->Rows[i]->Cells[0]->Value = i + 1;

Table->Rows[i]->Cells[1]->Value = array[i];

}

InFile(array, SIZE);

}

else

{

MessageBox::Show("Short diapason", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

}

}

}

private: void Finding()

{

if (textBox3->Text == "")

{

MessageBox::Show("Enter an object", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else

{

int element = Convert::ToInt32(textBox3->Text);

if (!radioButton1->Checked && !radioButton2->Checked && !radioButton3->Checked && !radioButton4->Checked)

{

MessageBox::Show("Chose type of searching", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else

{

//time\_t seconds = time(NULL);

//tm \*time1 , \*time2;

clock\_t start, end;

int size = Table->RowCount;

int \*array = new int[size];

int result = 0;

FromFile(array, size);

if (radioButton1->Checked)

{

start = clock();

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

{

result = Searching::linear(array, element, size);

}

end = clock();

TIME->Text = Convert::ToString(((double)end - (double)start) / ((double)CLOCKS\_PER\_SEC)/N);

if (result == -1)

{

MessageBox::Show("Nothis was found", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else

{

int test = 0;

//int iter = 0;

test = selectedOut();

if (test > SIZE)

{

test = 0;

}

Table->Rows[test]->Selected = false;

Table->Rows[result]->Selected = true;

selectedIn(result);

richTextBox1->Text = Convert::ToString(result + 1);

int iter = IterationOut();

Iteration->Text = Convert::ToString(iter);

}

}

else

{

if (radioButton2->Checked)

{

start = clock();

Sorting::buble(array, SIZE);

InFile(array, SIZE);

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

{

Table->Rows[i]->Cells[1]->Value = array[i];

}

int test = 0;

test = selectedOut();

if (test > SIZE)

{

test = 0;

}

Table->Rows[test]->Selected = false;

//const int n = 500;

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

{

result = Searching::Binary(array, element, SIZE, 0, SIZE - 1);

}

end = clock();

if (result != -1)

{

TIME->Text = Convert::ToString(((double)end - (double)start) / ((double)CLOCKS\_PER\_SEC)/N);

richTextBox1->Text = Convert::ToString(result+1);

Table->Rows[result]->Selected = true;

selectedIn(result);

int iter = IterationOut();

Iteration->Text = Convert::ToString(iter);

}

else

{

MessageBox::Show("Nothis was found", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

}

else

{

if (radioButton3->Checked)

{

start = clock();

Sorting::buble(array, SIZE);

InFile(array, SIZE);

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

{

Table->Rows[i]->Cells[1]->Value = array[i];

}

int test = 0;

test = selectedOut();

if (test > SIZE)

{

test = 0;

}

Table->Rows[test]->Selected = false;

result = 0;

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

{

result = Searching::ABinary(array, element, SIZE);

}

end = clock();

if (result != -1)

{

TIME->Text = Convert::ToString(((double)end - (double)start) / ((double)CLOCKS\_PER\_SEC)/N);

richTextBox1->Text = Convert::ToString(result + 1);

Table->Rows[result]->Selected = true;

selectedIn(result);

int iter = IterationOut();

Iteration->Text = Convert::ToString(iter);

}

else

{

MessageBox::Show("Nothis was found", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

}

else

{

start = clock();

Sorting::buble(array, SIZE);

InFile(array, SIZE);

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

{

Table->Rows[i]->Cells[1]->Value = array[i];

}

int test = 0;

test = selectedOut();

if (test > SIZE)

{

test = 0;

}

Table->Rows[test]->Selected = false;

result = 0;

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

{

result = Searching::Shar(array,element,SIZE);

}

end = clock();

if (result != -1)

{

TIME->Text = Convert::ToString(((double)end - (double)start) / ((double)CLOCKS\_PER\_SEC) / N);

richTextBox1->Text = Convert::ToString(result + 1);

Table->Rows[result]->Selected = true;

selectedIn(result);

int iter = IterationOut();

Iteration->Text = Convert::ToString(iter);

}

else

{

MessageBox::Show("Nothis was found", "Fatal Error",

MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

}

}

}

}

}

}

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

{

Finding();

}

};

}