**3 ОПИС ПРОГРАМНОГО ПРОДУКТУ**

## **3. 1 Опис функціональної моделі**

Програмний продукт (ПП) містить наступні складові:

1. **робота з індивідуальною задачею** (ІЗ):
   1. введення вручну даних ІЗ;
   2. генерація (випадковим чином) даних ІЗ;
   3. збереження в файл/читання з файлу/редагування даних ІЗ;
   4. розв’язання ІЗ усіма розробленими методами (з виведенням на екран відповідних розв’язків та значень ЦФ);
2. **експериментальне дослідження розроблених алгоритмів:** 
   1. задання діапазонів зміни параметрів експериментів:
      * розмірність задач (від; до; крок);
      * кількість ІЗ, яку необхідно згенерувати для кожної розмірності;
      * діапазоні зміні коефіцієнтів (ЦФ та обмежень);
   2. генерація множини ІЗ;
   3. розв’язання множини згенерованих ІЗ усіма розробленими методами;
   4. виведення результатів експериментів:
      * порівняння за точністю;
      * порівняння за часом.

## **3. 2 Рішення з інформаційного забезпечення**

### 3.2.1 Вхідні дані

На вхід алгоритм отримує матрицю суміжності. Матриця суміжності зчитується з файлу, генерується випадковим чином або вводиться через користувацький інтерфейс (рисунок 3.1).



Рисунок 3.1 - Приклад користувацького інтерфейсу для введення даних.

***3.2.2 Вихідні дані***

На вихід алгоритм видає час виконання у секундах, значення цільової функції z (оптимальну кількість тварин) та шлях – тварин, яких необхідно взяти, щоб їх кількість була найбільшою. Також програма малює граф по вхідним даним та позначає вершини, що входять в шлях (рисунок 3.2).

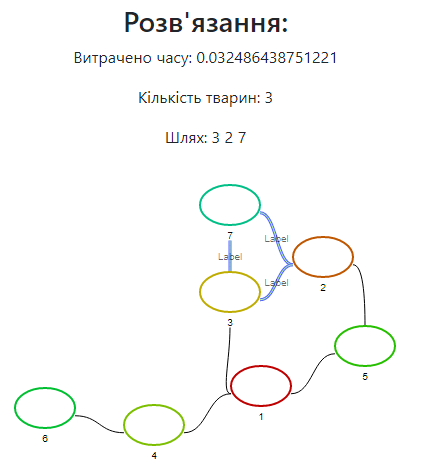


Рисунок 3.2 - Приклад вихідних даних.

### 3.2.3 Опис інформаційного забезпечення

В цьому пункті наводиться опис способів зберігання інформації.

Якщо при реалізації ПП розроблена база даних, то необхідно представити її у вигляді ER-діаграми з описом (або фізичної моделі бази даних). Також опишіть таблиці БД за шаблоном, представленим в таблиці 3.1.

***Таблиця 3.1 – Шаблон опису таблиці БД***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Назва таблиці*** | ***Назва поля*** | ***Тип*** | ***Опис*** |
| algorithmInfo | Id | Int(11) | Використовується для як унікальний ідентифікатор |
| Time | Float | Час виконання алгоритму |
| Animals | Int(11) | Кількість сумісних тварин |
| MatrixSize | Int(11) | Розмір матрці |
| Type | Tinyint(4) | Алгоритм мурашиний чи генетичний |

## **3.3 Рішення з програмного забезпечення**

### 3.3.1 Опис обраних засобів розробки

Усі обчислення виконувались засобами python. Серверна частина написана на php що викликав скрипти python для розрахунків. Фронтенд частина написана з використанням html, css, js, а саме такі фреймворки: bootstrap, dracula graph library, charjs.

### 3.3.2 Архітектура програмного забезпечення

#### 3.3.2.1 Схема архітектури ПП

#### ├── algorithm.php (вивід відомостей про алгоритм)

#### ├── antDescription.php (вивід опису про мурашиний алгоритм)

#### ├── authors.php (вивід відомостей про авторів)

#### ├── css (папка зі стилями)

#### │   └── bootstrap.min.css (стиля фреймворку бутстрап)

#### ├── example.php (вивід результату обчислення)

#### ├── favicon.ico (іконка)

#### ├── functions.php (файл з функціями пхп)

#### ├── geneticDescription.php (вивід відомостей про генетичний алгоритм)

#### ├── header.php (шапка сайту)

#### ├── img (папа з рисунками)

#### ├── index.php (сторінка з основною статистикою)

#### ├── js (папка з джс скриптами)

#### │   ├── bootstrap.min.js (скрипт бутсрапу)

#### │   ├── Chart.min.js (скрипт графіків)

#### │   ├── dracula.min.js (скрипт графів)

#### │   ├── jquery-3.3.1.slim.min.js (джейкуері скрипт)

#### │   ├── popper.min.js (скрипт бутсрапу)

#### │   └── raphael.js (скрипт графів)

#### ├── phpmyadmin -> /usr/share/phpmyadmin (інструмент керування БД)

#### └── system (папка з обчислювальними скриптами)

#### ├── countAlgorithm.php (скрипт що визиває потрібний python файл

#### ├── input.txt (вхідні дані користувача)

#### ├── output.txt (вихідні дані)

#### ├── python (папка з python скриптами)

#### │   ├── antAlgorithm.py (скрипт що обчислює матрицю за домопогою мурашиного алгоритму)

#### │   └── geneticAlgorithm.py (скрипт що обчислює матрицю за домопогою генетичного алгоритму)

#### └── uploadAlgorithm.php (скрипт що опрацьовує завантажені файли)

#### 3.3.2.3 Специфікація функцій

antAlgorithm.php

|  |  |  |
| --- | --- | --- |
| Назва функції | Аргументи | Опис |
| findAnimals | head | Знаходить сумісних тварин з даною вершиною |
| countNumberOfEdges |  | Підраховує кількість ребер в графі |
| countEdgesFromHead | head | Рахує ребра з вершини |
| countStartHeadProbability | numberOfEdges | Вираховує початкові ймовірності переходу для мурашиного алгоритму |
| countHeadProbability | head, commonHeads, headSum | Вираховує ймовірності переходу для мурашиного алгоритму |
| choosHeadToGo | startHeadProbability, randomWay | Обирає куди підти далі |
| countCommonHeads | markedList | Підраховує ребра між вершинами в поміченому списку |

geneticAlgorithm.py

|  |  |  |
| --- | --- | --- |
| Назва функції | Аргументи | Опис |
| returnAnimalsList | animals | Повертає список тварин з якими сумісна дана тварина |
| changeChar | s, p, r | Заміняє символ у строці |
| getRandomPopulation | stringLength | Генерує випадкову популяцію |
| checkAnimals | matrix, animalOneNumber, animalTwoNumber | Перевіряє тварин на сумісність |
| checkIfSatisfy | matrix, item | Перевіряє чи усі члени популяції сумісні |
| countAnimals | item | Підраховує тварин у популяції |
| printPopulations | count, startMatrix, populationItem | Виводить інформацію про популяцію |
| findMaxFromPopulations | firstGroup, secondGroup | Обирає найркащу популяцію з двох груп |
| crossingOver | globalCout, parentsList | Функція кросинговеру |

## **3.4 Керівництво користувача**

*3.4.1 Інструкція користувача*

Система складається з 4-х частин: головна сторінка зі статистикою, мурашиний алгоритм, генетичний алгоритм, відомості про авторів.

На головній сторінці можна побачити графіки залежності часу виконання програми від розмірності матриці (Рисунок 3.3, 3.4).

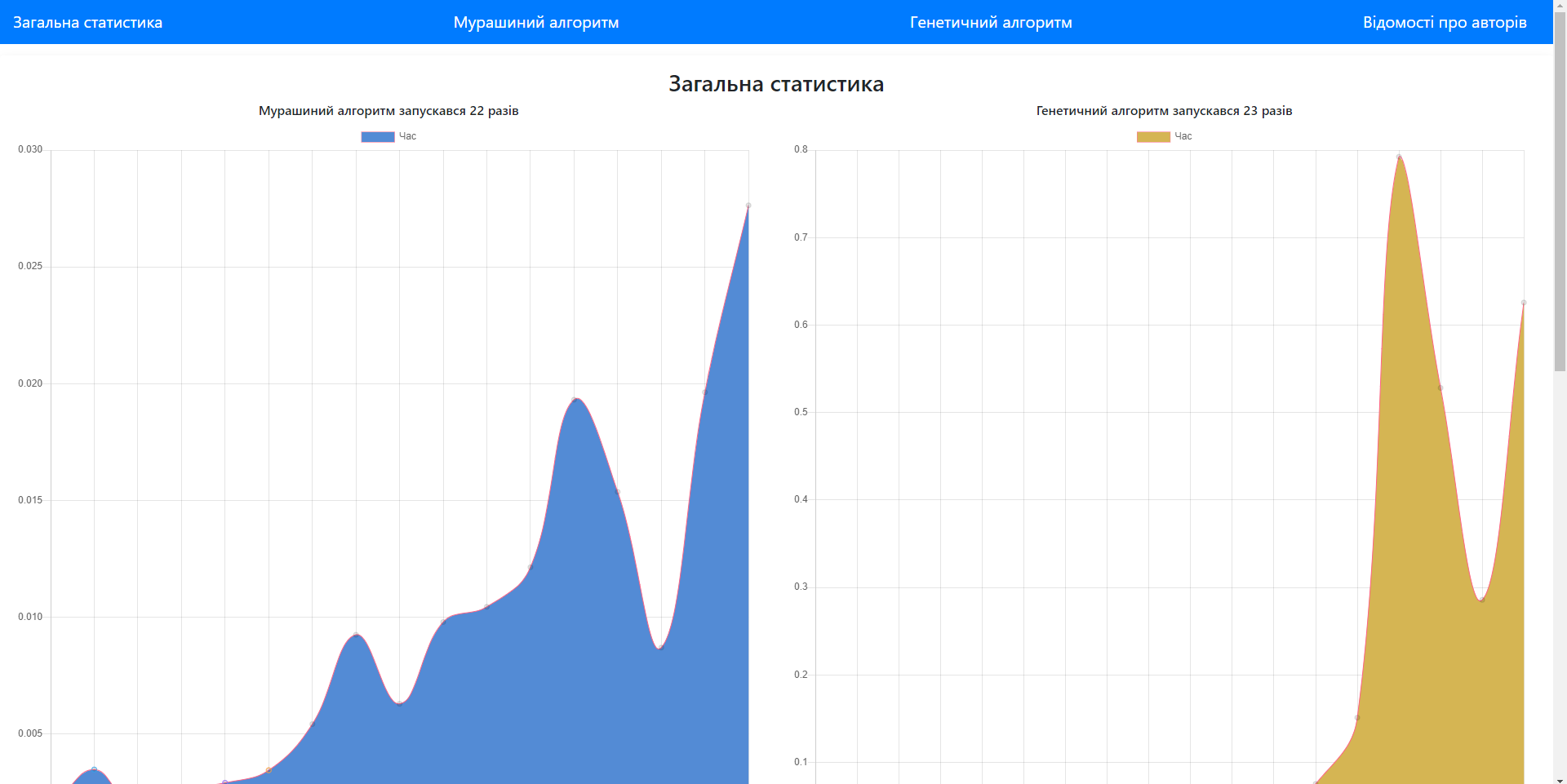


Рисунок 3.3 – Графіки залежності часу виконання від розміру вхідних даних

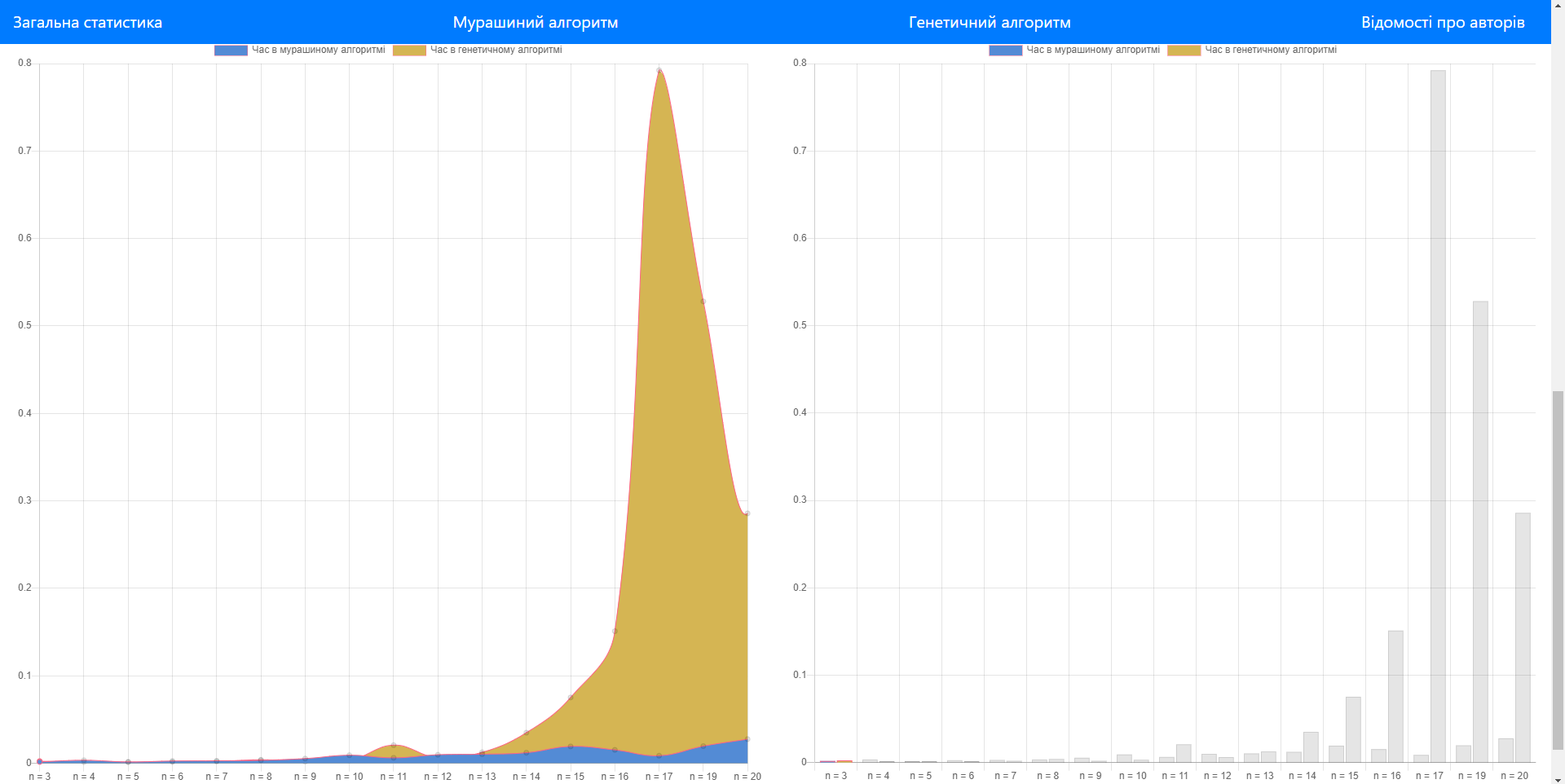


Рис. 3.4 - Графіки залежності часу виконання від розміру вхідних даних

На сторінці мурашиного алгоритму можна прочитати опис методу та обрати розмірність матриці (Рисунок 3.5).

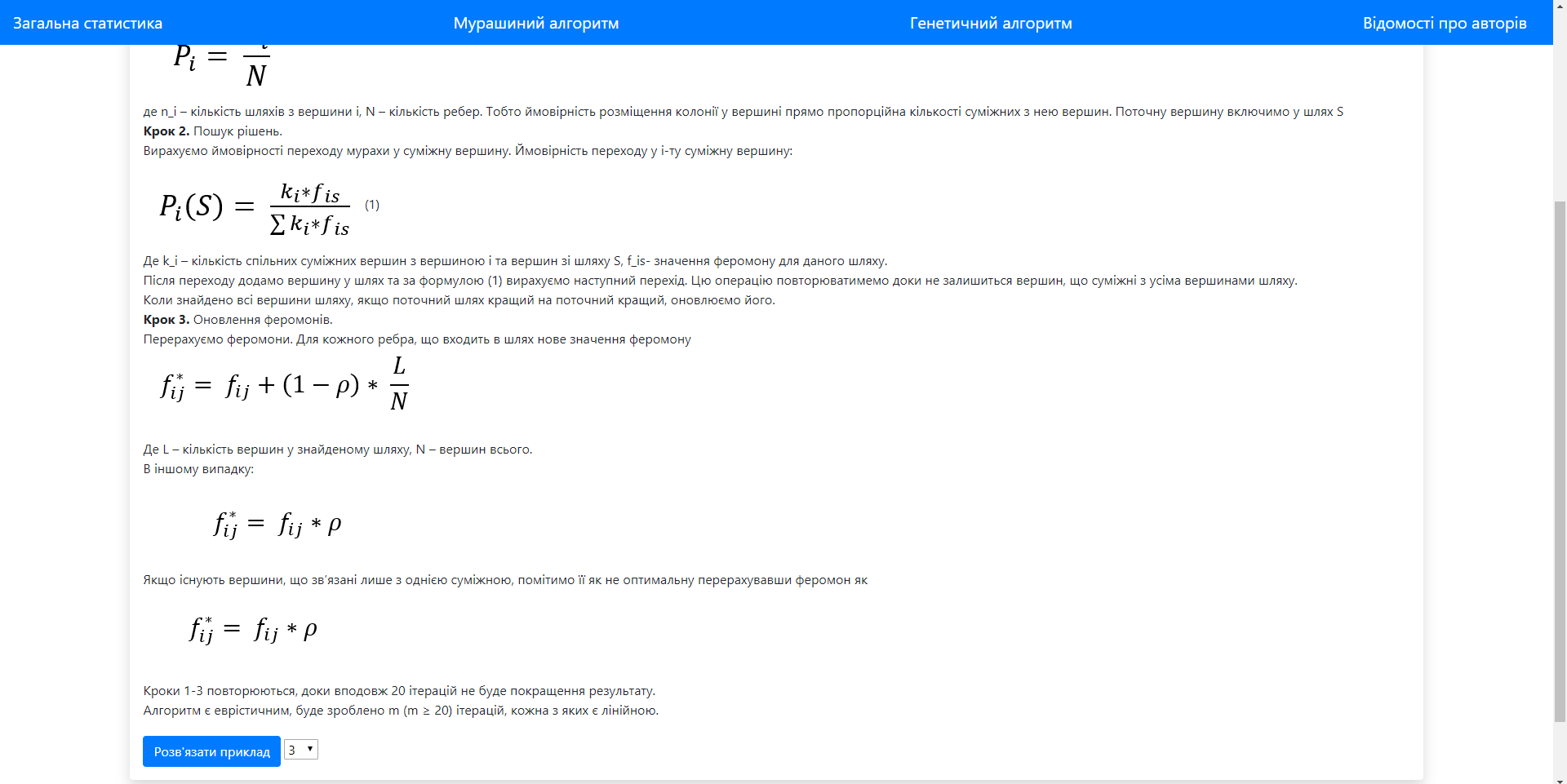


Рисунок 3.5 – стартова сторінка мурашиного алгоритму

Обравши розмірність матриці, користувач переходить на сторінку розв’язку матриці. Можна ввести дані вручну, заповнити випадковим чином або завантажити з файла. Після введення даних, треба натиснути кнопку «Роозв’зати» після чого з’явиться відповідь та граф з потрібними маршрутами (Рисунок 3.6).

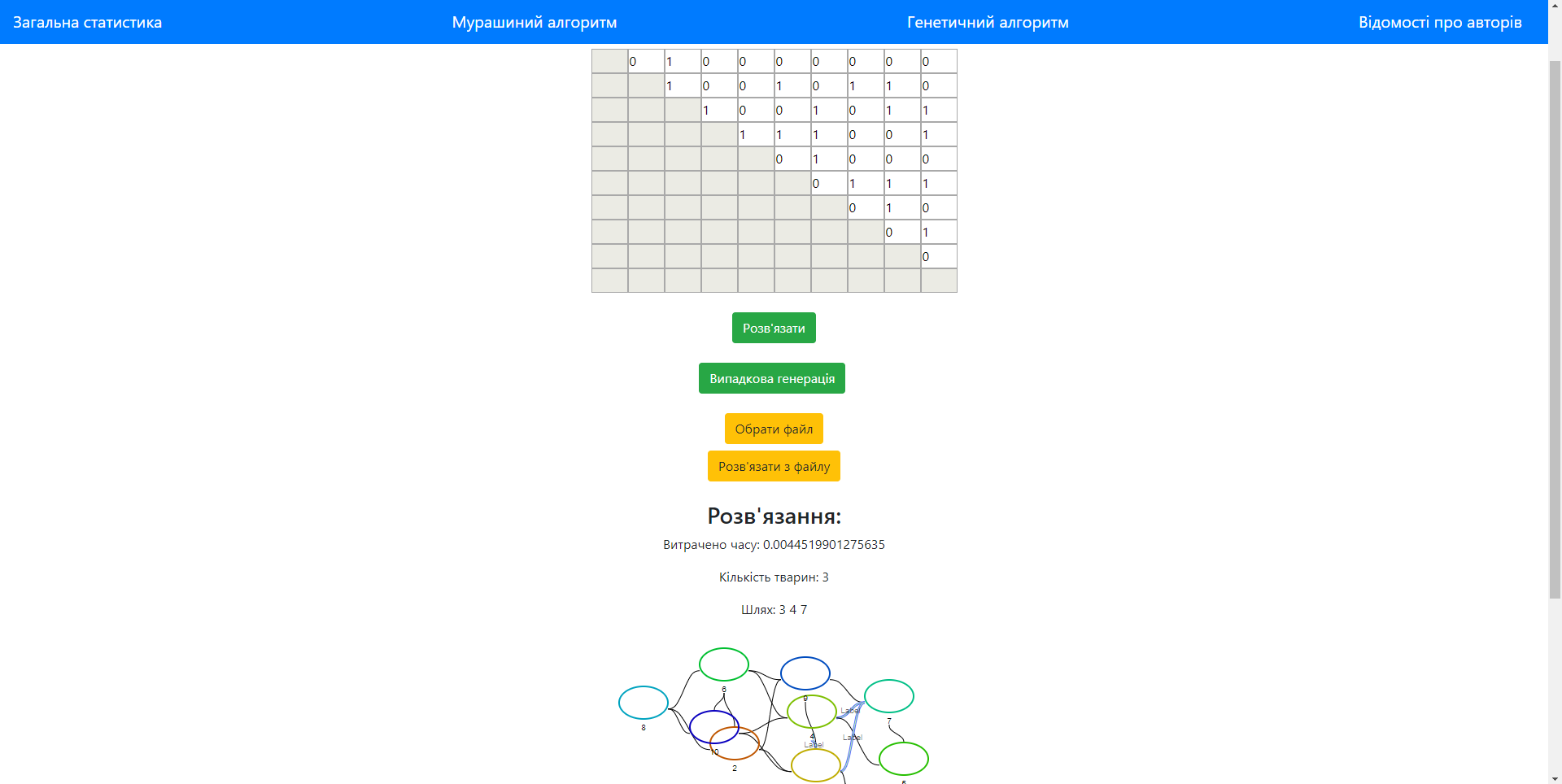


Рисунок 3.6 – Розв’язок задачі

Генетичний алгоритм має ті самі функції, відрізняється лише опис методу (Рисунок 3.7).

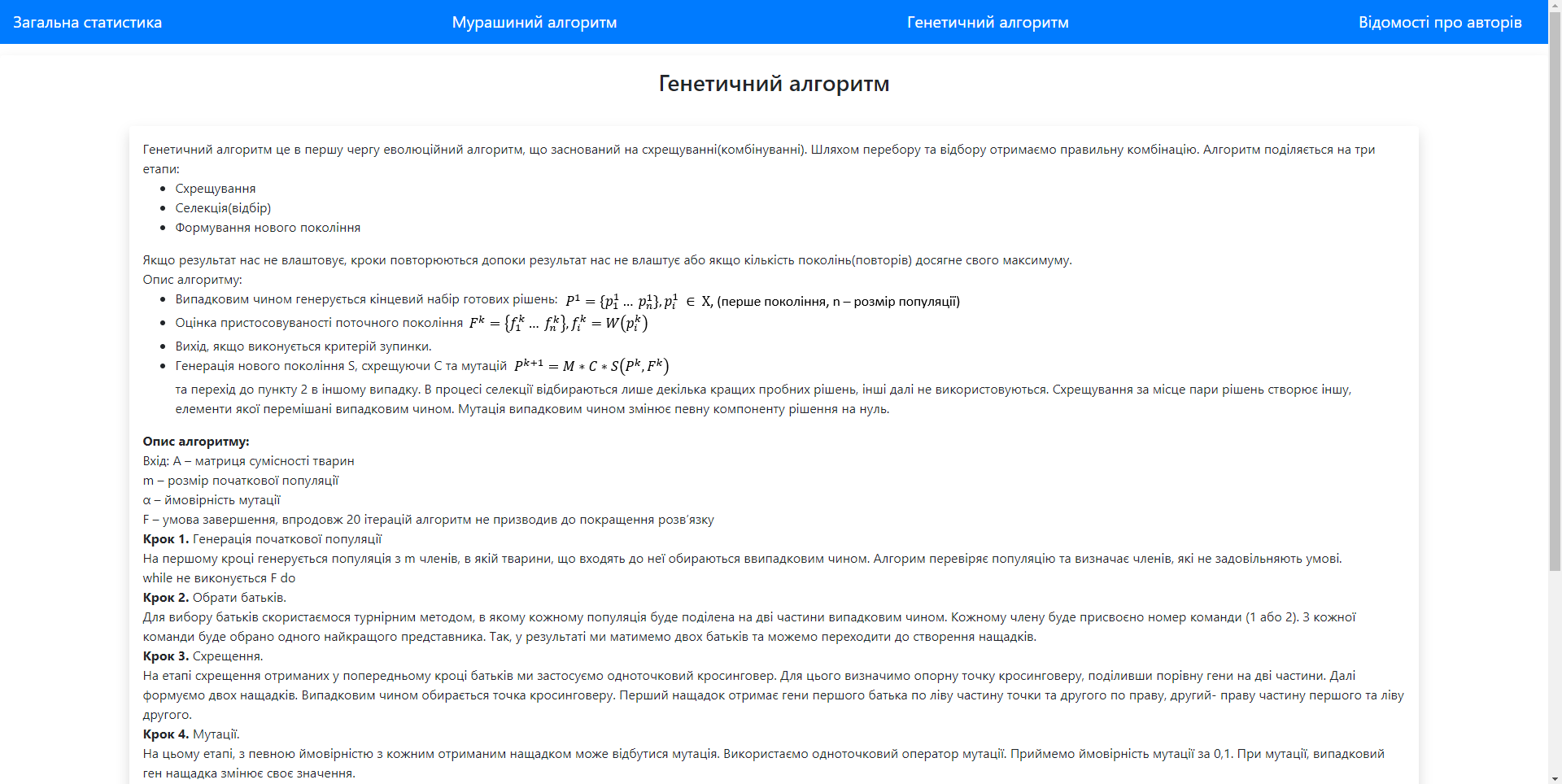


Рисунок 3.7 – Стартова сторінка генетичного алгоритму

Сторінка «відомості про авторів» містить відомості про авторів (Рисунок 3.8).

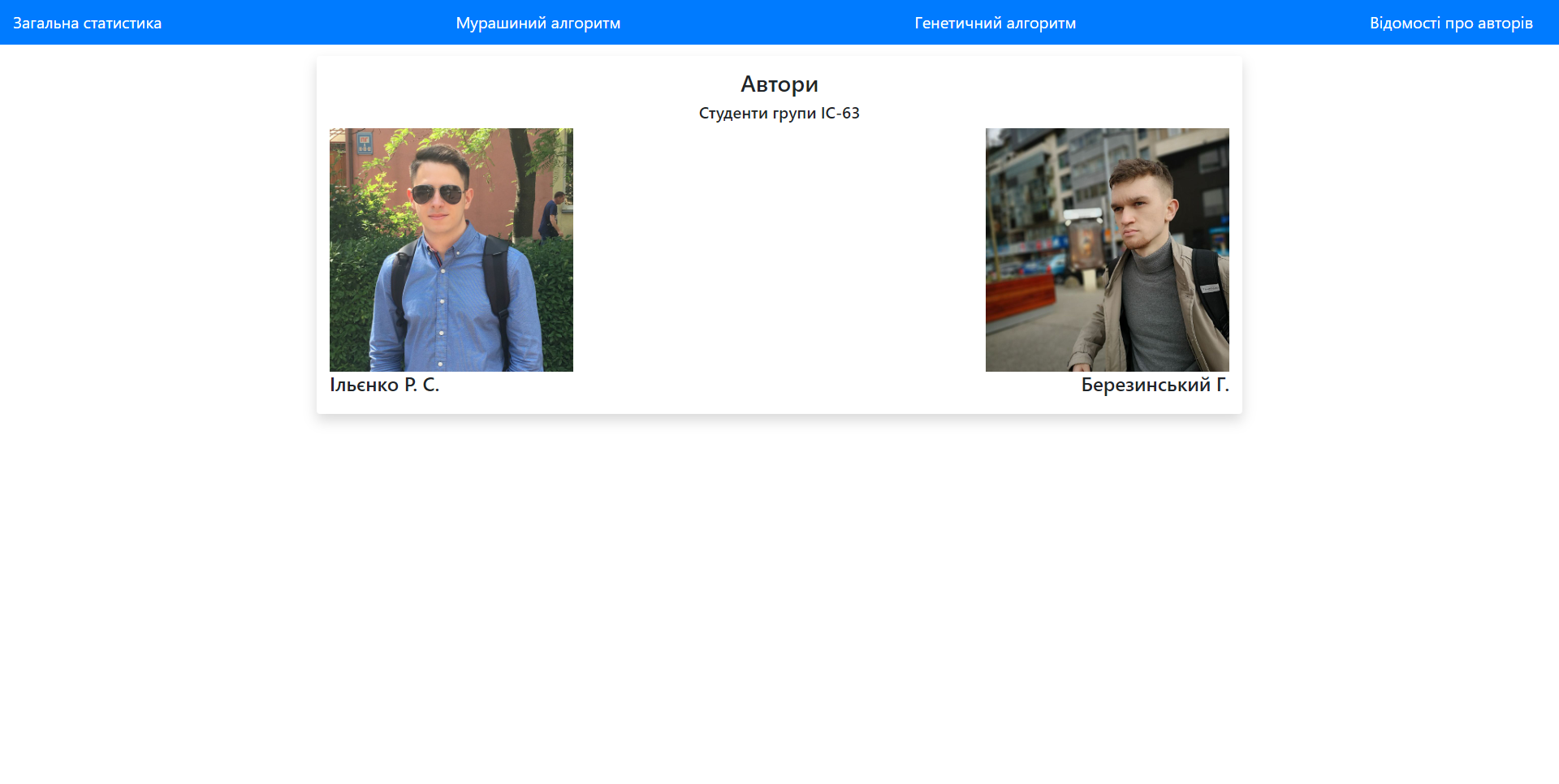


Рисунок 3.8 – Про авторів

***3.4.2 Методика випробувань***

Для випробовування часу роботи кожен з алгоритмів запускався по 10 разів на випадково згенерованих даних розмірністю віл 3 до 10. Алгоритм мурашиних колоній також тестувався на великому наборі даних – від 30 до 35. Генетичний алгоритм обмежився лише тестами невеликої розмірності через надзвичайно високу складність алгоритму, так вже на матриці розмірністю 20 алгоритм працює 16 секунд.

Для випробування точності алгоритмів, кожен запускався на вхідних даних, для яких відоме оптимальне значення ЦФ 1000 разів та вівся підрахунок правильних визначень результатів.

Дослідження алгоритму знаходиться у розділі 4.