ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ
Навчально-науковий інститут денної освіти
Форма навчання денна
Кафедра комп'ютерних наук та інформаційних технологій

Допускається до захисту
Завідувач кафедри
_____Олена ОЛЬХОВСЬКА
(підпис)
«_____»___202_ р.

КВАЛІФІКАЦІЙНА РОБОТА
на тему

РОЗРОБКА НАВЧАЛЬНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ТЕМИ «COLLECTIONS. ARRAYLIST» ДИСЦИПЛІНИ «ПРОГРАМУВАННЯ»

зі спеціальності 122 Комп'ютерні науки
освітня програма «Комп'ютерні науки»
ступеня бакалавра

Виконавець роботи Колобов Дмитро Віталійович
_____«____» _____202_ р.
(підпис)

Науковий керівник к.ф.-м.н. Олексійчук Юрій Федорович
_____«____»_____202_ р.
(підпис)

Рецензент

ПОЛТАВА 2024

РЕФЕРАТ

Записка: 41 с., 19 рис., 0 таблиць, 1 додаток, 16 джерел.

COLLECTIONS. ARRAYLIST, НАВЧАЛЬНИЙ СИМУЛЯТОР, APACHE NETBEANS

Об’єкт розробки – процес дистанційного навчання програмуванню на Java.

Мета роботи – розробка програмного забезпечення симулятору для навчання роботі з колекціями Java, зокрема з ArrayList.

Методи дослідження – методи програмування в Java, використання Java Collections Framework, середовище візуальної розробки Apache NetBeans, тестування програмного забезпечення.

В роботі проведено аналіз існуючих рішень в галузі навчальних симуляторів для програмування, виділено ключові вимоги та особливості роботи з колекціями в Java. Визначено основні переваги та недоліки існуючих систем, що стали основою для розробки власного рішення.

Розроблено алгоритм тренажеру, що включає теоретичні матеріали, практичні завдання та модуль тестування знань. Особлива увага приділяється інтерактивності та залученню студента в процес навчання через систему підказок та пояснень, що сприяє кращому засвоєнню матеріалу.

Програмна реалізація симулятора забезпечує можливість динамічної взаємодії з користувачем, адаптацію завдань залежно від рівня підготовки та ведення статистики результатів. Використання середовища Apache NetBeans дозволило ефективно інтегрувати всі компоненти програми, забезпечити їх стабільність та високу швидкість роботи.

Здійснено тестування симулятора на групі студентів, результати якого показали значне покращення рівня знань учасників у порівнянні з традиційними методами навчання. Таким чином, розроблений симулятор є ефективним інструментом у процесі освіти, що підтверджує актуальність та необхідність його подальшого розвитку та впровадження в навчальний процес.

Зміст

ВСТУП

У сучасному світі програмування відіграє ключову роль у розвитку технологій та оптимізації багатьох процесів, від промисловості до освіти. Особливо це стосується мови програмування Java, яка є однією з найпопулярніших у світовому масштабі завдяки своїй універсальності та ефективності. Важливим аспектом мови Java є робота з колекціями, зокрема з ArrayList, який дозволяє зберігати дані динамічно.

Розвиток технологій відкрив нові можливості для вдосконалення методів навчання. Освітні симулятори, які використовуються для підготовки спеціалістів у галузі програмування, забезпечують не тільки теоретичне ознайомлення з матеріалом, але й можливість його практичного застосування. Це дозволяє студентам не просто вчити нові концепції, але й відразу тестувати їх на практиці, що є надзвичайно важливим для засвоєння матеріалу.

Об’єктом даного проекту є розробка навчального програмного забезпечення на тему "Collections. ArrayList" для дисципліни "Програмування". Програма покликана допомогти студентам глибше зрозуміти та освоїти роботу з динамічними колекціями в Java, використовуючи інтерактивний підхід до навчання.

Використання симуляторів у навчальному процесі має ряд переваг, які ми детально розглянемо в даній роботі. Це включає миттєвий зворотній зв'язок для студентів, можливість самостійно контролювати свій прогрес та гнучкість у виборі тем і завдань залежно від індивідуального рівня знань.

Мета даної роботи полягає у створенні повноцінного симулятора, який би ефективно інтегрував теоретичні знання і практичні вправи з використанням ArrayList, що дозволить студентам не тільки краще засвоїти матеріал, але й застосувати його на практиці.

Протягом розробки було використано сучасні інструменти програмування, такі як середовище розробки Apache NetBeans та Java Development Kit. Результатом стало створення програми, що включає модулі навчання, практичних завдань і тестування.

1. ПОСТАНОВКА ЗАДАЧІ

Для реалізації курсової роботи необхідно створити алгоритм роботи електронного тренажеру з теми "Collections. ArrayList" для курсу "Програмування". План роботи включає наступні етапи:

- Вивчення теоретичних матеріалів з роботи з колекціями в Java, зокрема з ArrayList, та їх застосування.
- Аналіз існуючих навчальних тренажерів для роботи з колекціями та програмуванням.
- Вибір прикладів та алгоритмів, які будуть використані в тренажері.
- Розробка алгоритму тренажера, що включає теоретичний матеріал, практичні завдання та тестування знань користувача.

Тренажер має забезпечити користувачам не тільки комплексні завдання для розвитку навичок роботи з ArrayList, але й можливість ознайомлення з теоретичними матеріалами перед виконанням завдань. Інтерфейс тренажера має бути інтуїтивно зрозумілим та зручним для користувачів, включаючи:

- Стартову сторінку для вибору режиму роботи (теорія, практика, тестування).
- Модуль для читання теоретичних матеріалів з можливістю переходу до практичних завдань.
- Вікно з практичними завданнями з можливістю введення коду і отримання зворотного зв'язку.
- Модуль тестування з автоматичною перевіркою відповідей та виведенням результатів.
- Систему підказок та допомоги при виникненні помилок з прикладами або рекомендаціями для виправлення.

Завданням також є забезпечення адаптації тренажера для різних категорій користувачів, включаючи початківців та досвідчених розробників, що дозволить зробити процес навчання максимально ефективним та зручним.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Роль та місце колекцій у мові програмування Java

Колекції в Java відіграють важливу роль у розробці програмного забезпечення, надаючи потужний інструментарій для роботи з наборами даних. Вони є частиною Java Collections Framework (JCF), що впроваджений у Java 2 Platform Standard Edition (J2SE) 1.2, і з того часу є неодмінним аспектом мови програмування Java.

Колекції дозволяють зберігати, отримувати, маніпулювати та комунікувати зборами даних. Вони забезпечують уніфікований інтерфейс для роботи з різними типами даних, що спрощує розробку програм і зменшує ймовірність помилок. Рамки колекцій включають різноманітні структури даних, такі як списки, множини, карти та черги, кожна з яких має свої особливості та призначена для рішення конкретних задач.

Різновиди колекцій(див рис. 1.1):

- Списки (List): Дозволяють зберігати елементи у визначеному порядку і забезпечують доступ до них за індексом. Дублікати допускаються.
- Множини (Set): Забезпечують зберігання унікальних елементів, тобто виключають можливість дублікатів.
- Карти (Map): Дозволяють зберігати пари "ключ-значення" і забезпечують швидкий доступ до значення за ключем.
- Черги (Queue): Підтримують операції вставки, вилучення та інспекції елементів відповідно до принципу FIFO (першим прийшов — першим вийшов) або приоритетів.

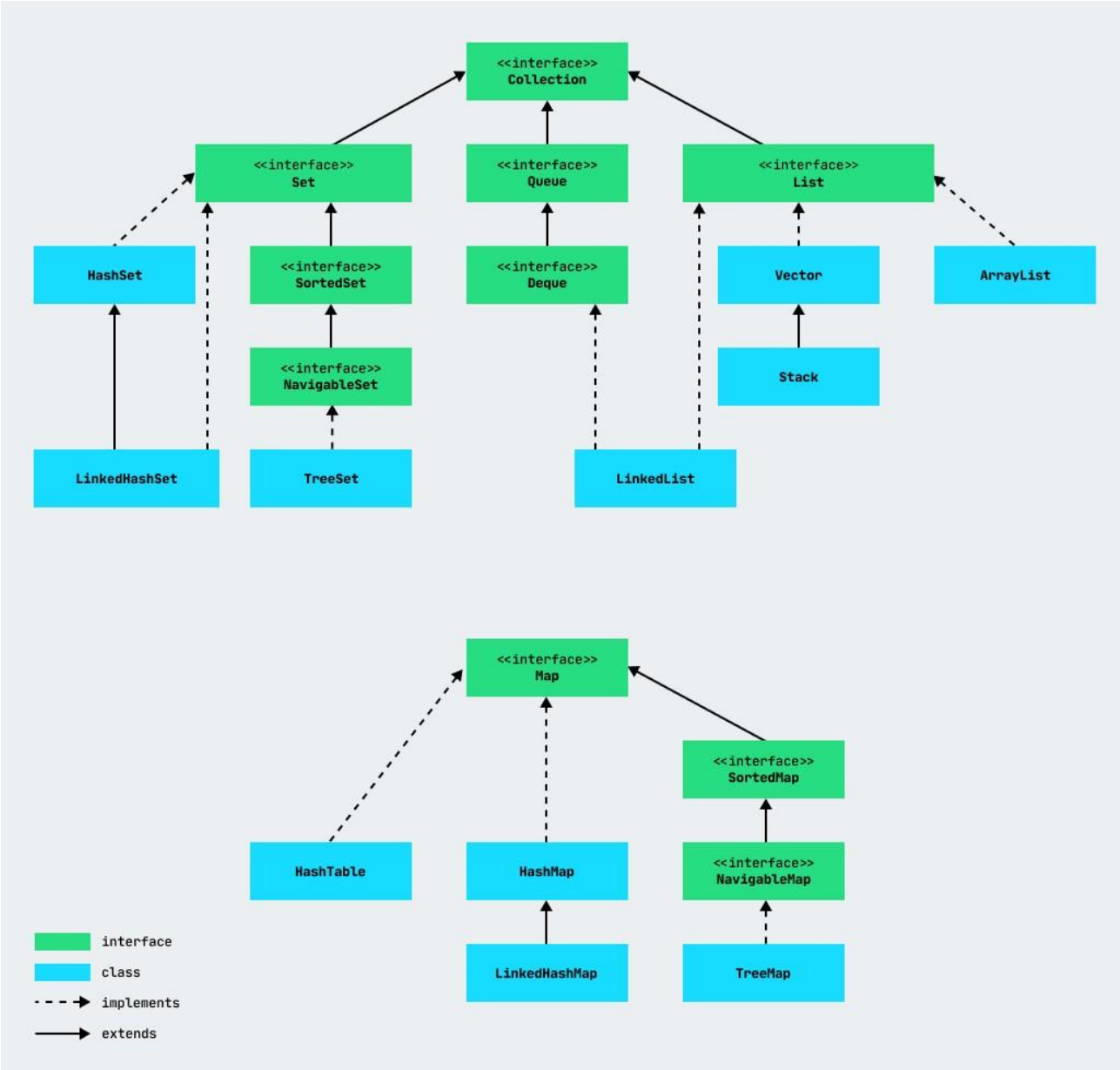

Рисунок 1.1 – Колекції Java

Використання колекцій в Java приносить ряд переваг, зокрема:

- Ефективність: Колекції оптимізовані за швидкодію та використання пам'яті, що робить їх ефективним рішенням для обробки великих обсягів даних.
- Гнучкість: Завдяки широкому спектру інтерфейсів та реалізацій, колекції можуть бути адаптовані до багатьох задач обробки даних.
- Безпека типів: Загальний використання Generics (узагальнення) з колекціями дозволяє підвищити безпеку типів під час компіляції.
- Широка підтримка: Будучи стандартною частиною Java, колекції мають велику підтримку та документацію.

Застосування

Колекції використовуються в широкому спектрі додатків Java: від простих утиліт до складних корпоративних систем. Вони є ключовими у розробці алгоритмів, управлінні даними, обробці подій, паралельному програмуванні та багатьох інших областях.

Завдяки своїй універсальності, ефективності та високому рівню інтеграції з мовою Java, колекції є незамінним інструментом для будь-якого Java-розробника.

2.2 Огляд прикладів комп'ютерних симуляторів

У сучасному світі освіти важко переоцінити значення навчальних програм і тренажерів. Ці інструменти відіграють ключову роль у процесі навчання, особливо коли йдеться про таку складну і витончену сферу, як програмування. Розглянемо тренажери, зосереджені на вивченні Java та, зокрема, на роботі з ArrayList, які стають невід'ємною частиною освітнього процесу завдяки їхнім численним перевагам:

- Доступність 24/7: Вони роблять навчальний процес безперервним та доступним з будь-якої точки світу в будь-який час. Це забезпечує студентам можливість навчатися в найзручніший для них час, перетворюючи вивчення на гнучкий і динамічний процес.
- Економія коштів: У порівнянні з традиційним навчанням, тренажери часто виявляються більш економічним варіантом, зменшуючи або навіть виключаючи потребу в фізичних навчальних матеріалах і ресурсах.
- Персоналізація навчання: Сучасні тренажери дозволяють адаптувати навчальний процес під індивідуальні потреби та рівень знань кожного студента, що робить навчання більш ефективним.
- Миттєвий зворотній зв'язок: Навчальні програми здатні швидко оцінювати відповіді студентів, надаючи їм зворотний зв'язок, що сприяє швидкому виявленню та усуненню прогалин у знаннях.
- Покрокове навчання: Дозволяють ефективно структурувати матеріал, розбиваючи його на логічні та зручні для сприйняття секції, що допомагає студентам краще засвоювати інформацію.

- Автоматизація процесу навчання: Використання дозволяє автоматизувати багато аспектів навчального процесу, звільняючи час викладачів для зосередження на більш складних та творчих завданнях.

Таким чином, тренажери для вивчення Java, особливо з фокусом на роботі з ArrayList, представляють собою цінний ресурс, який не тільки підвищує ефективність навчання, але й робить його більш доступним, інтерактивним та персоналізованим. Вони сприяють кращому розумінню та використанню ключових концепцій програмування, відіграючи важливу роль у формуванні майбутнього покоління розробників.

2.3 Огляд прикладів комп'ютерних тренажерів

У процесі розробки навчального програмного забезпечення для глибокого освоєння Java, зокрема роботи з ArrayList, критично важливим етапом є аналіз вже існуючих рішень в цій галузі. Вивчення досвіду попередників дозволяє не тільки уникнути повторення типових помилок, але й вбудувати в свій проект найбільш успішні практики та інновації.

Візьмемо до уваги наступні навчальні програми, які стали відомими завдяки своїй ефективності та популярності серед користувачів.

"Codewars" є одним з піонерів у викладанні основ програмування(див рис 1.4). Створений для надання зручної та інтуїтивно зрозумілої платформи для новачків, цей тренажер відзначається своєю здатністю поєднувати теоретичні знання з практичними завданнями. Його підхід до викладання базується на принципі "від простого до складного", забезпечуючи поетапне засвоєння матеріалу.

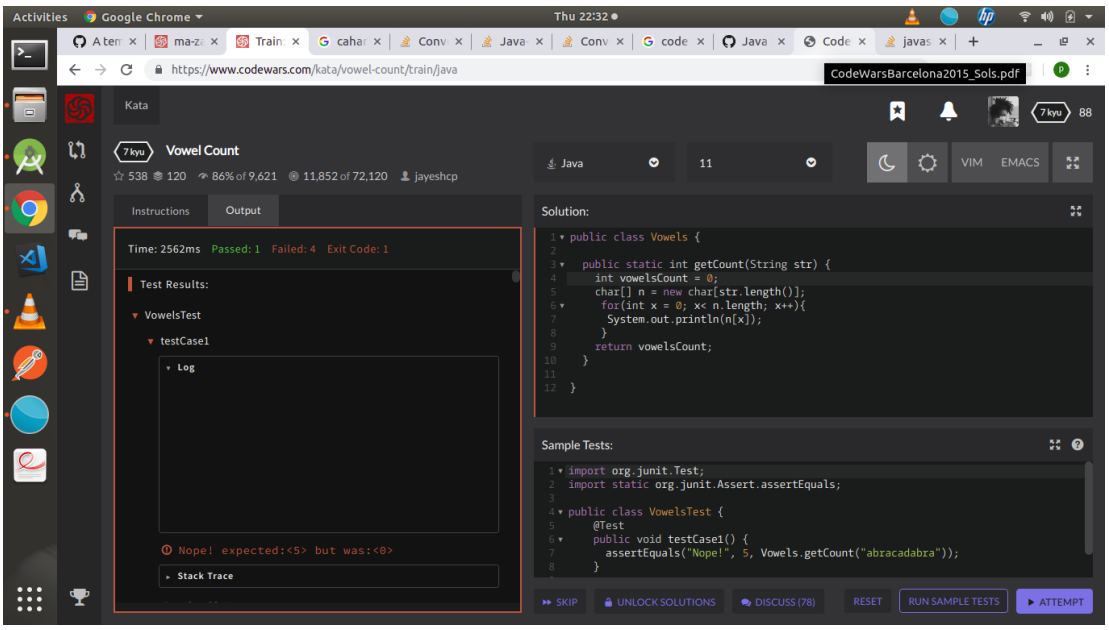

Рисунок 1.4 –Codewars

"LeetCode" славиться своїми завданнями на алгоритми та структури даних, пропонуючи великий вибір задач, що допомагають поглибити знання в Java. Завдяки своїй зосередженості на практичних аспектах програмування, "LeetCode" є чудовим інструментом для розвитку навичок роботи з різними структурами, включаючи ArrayList.

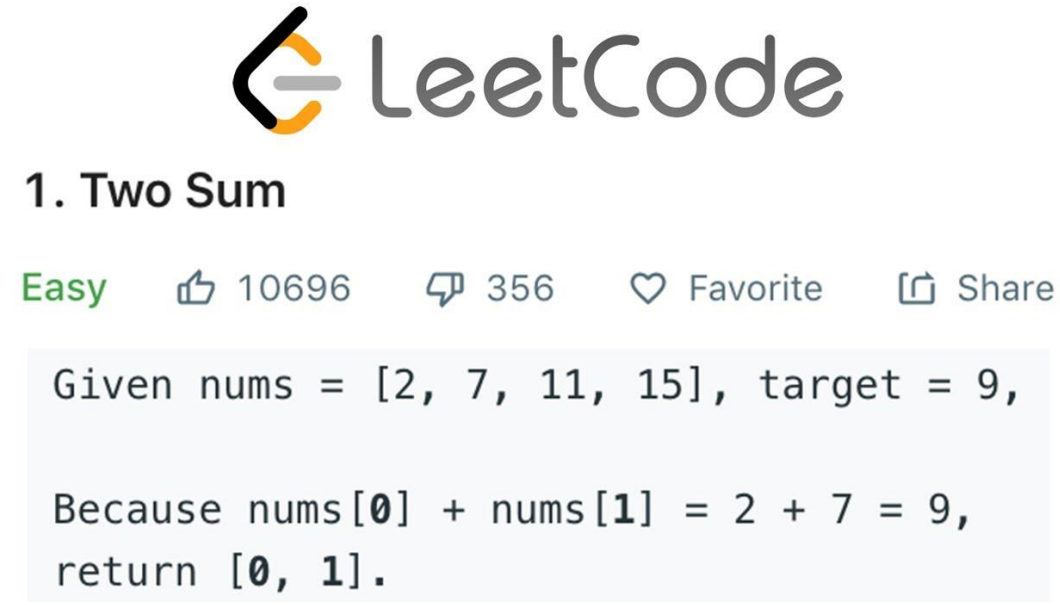

Рисунок 1.5 –LeetCode

Аналізуючи ці тренажери, можна зробити наступні висновки щодо успішної реалізації навчальної програми:

1. Поєднання теорії та практики: Навчальне програмне забезпечення повинне забезпечувати збалансований мікс теоретичних знань та практичних завдань.
2. Інтерактивність: Важливим є наявність інтерактивних елементів, що залучають користувачів у процес навчання.
3. Система зворотного зв'язку: Миттєве надання зворотного зв'язку користувачам допомагає їм ефективно навчатися на власних помилках.
4. Гнучкість та доступність: Навчальний процес повинен бути доступним для користувачів з різним рівнем знань та навичок, пропонуючи гнучкі шляхи освоєння матеріалу.

Власна навчальна програма на тему ArrayList в Java буде враховувати ці ключові елементи успіху, намагаючись створити зручний, ефективний та всебічно розвиваючий ресурс для студентів та розробників, які прагнуть поглибити свої знання та навички в роботі з колекціями в Java.

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1 Основи роботи з колекціями в Java

Колекції в Java є основним механізмом для зберігання груп об'єктів. Java Collections Framework (JCF) включає набір класів і інтерфейсів, які дозволяють роботу з різними типами колекцій. Основні інтерфейси, що входять до JCF, це List, Set, Queue та Map, кожен з яких має свої унікальні характеристики та призначення.

Серед реалізацій інтерфейсу List, ArrayList відіграє ключову роль у зберіганні послідовностей об'єктів, де кожен об'єкт має свій індекс. Це робить ArrayList ідеальним вибором для реалізації динамічних масивів, де потрібен швидкий доступ до елементів.

ArrayList заснований на звичайному масиві об'єктів, але дозволяє динамічно змінювати свій розмір завдяки автоматичному розширенню масиву, коли в нього додаються нові елементи. Основні характеристики ArrayList включають:

- Індексований доступ: Можливість швидкого доступу до будь-якого елемента за індексом.
- Зміна розміру: ArrayList збільшує свій розмір автоматично, коли до нього додаються нові елементи.
- Гнучкість: Підтримка різноманітних операцій, таких як вставка, видалення, пошук елементів.

Основні операції, які можуть бути виконані з ArrayList, включають:

- Додавання елементів: Метод add(E e) дозволяє додавати елемент в кінець списку або за вказаним індексом за допомогою add(int index, E element).
- Видалення елементів: Метод remove(Object o) та remove(int index) використовуються для видалення елементів.
- Доступ до елементів: Метод get(int index) повертає елемент за вказаним індексом.
- Зміна елементів: Метод set(int index, E element) замінює елемент на вказаній позиції.

Використання ArrayList

Приклад створення ArrayList та роботи з ним:

```
import java.util.ArrayList;
public class Main {
    public static void main(String[] args) {
        ArrayList<String> fruits = new ArrayList<>();
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Cherry");
        System.out.println("Список фруктів: " + fruits);
        // Видалення елемента
        fruits.remove("Banana");
        System.out.println("Після видалення: " + fruits);
        // Доступ до елемента
        String fruit = fruits.get(0);
        System.out.println("Перший фрукт у списку: " + fruit);
        // Зміна елемента
        fruits.set(0, "Apricot");
        System.out.println("Після заміни: " + fruits);
    }
}
```

Цей приклад ілюструє базові операції, що можуть бути виконані з ArrayList, такі як додавання, видалення, доступ до елементів та їх зміна.

3.2 Алгоритм роботи програми

Екран вітання:

"Ласкаво просимо до тренажера ArrayList! Ця програма допоможе вам опанувати одну з ключових структур даних Java, використовуючи серію теоретичних уроків, практичних завдань і тестів. Ви готові почати?"

Вибір розділу:

1. Теоретична частина

2. Практична частина
3. Тестова частина"

Теоретична частина

Крок 1. Введення в ArrayList:

"В Java, ArrayList є частиною Java Collections Framework і представляє собою динамічний масив, що забезпечує можливість зберігання елементів будь-якого типу. Відмінною особливістю A rrayList є його здатність автоматично змінювати свій розмір."

Приклад коду:

```
ArrayList<String> exampleList = new ArrayList<>();
exampleList.add("Apple");
exampleList.add("Banana");
System.out.println("Перший елемент: " + exampleList.get(0));
```

Крок 2. Структура ArrayList:

"ArrayList під капотом використовує звичайний масив. Під час додавання елементів, якщо масив заповнений, створюється новий масив більшого розміру, а елементи копіюються в новий масив."

Приклад коду:

```
ArrayList<Integer> numbers = new ArrayList<>(5);
numbers.add(1);
numbers.add(2);
System.out.println("Розмір: " + numbers.size());
```

Крок 3. Переваги та недоліки ArrayList:

"Переваги ArrayList включають легкість використання, швидкий доступ до елементів за індексом. Однак, видалення елементів або додавання в середину списку може бути дорогим, оскільки це вимагає зсуву елементів."

Крок 4. Операції з ArrayList:

"Давайте розглянемо основні операції, які можна виконувати з ArrayList:

- Додавання елементів: add(E e)
- Видалення елементів: remove(Object o)

- Доступ до елементів: get(int index)
- Розмір списку: size()"

Приклад коду:

ArrayList<String> fruits = new ArrayList<>();

fruits.add("Apple");

fruits.remove("Apple");

System.out.println("Кількість фруктів: " + fruits.size());

Заключення теоретичної частини

"Вітаємо! Ви успішно пройшли теоретичну частину ArrayList. Тепер ви знаєте основні концепції та операції, пов'язані з цією важливою структурою даних в Java. Готові перейти до практичних завдань або тестування?"

Вступ до практичної частини

На екрані вітання практичної частини тренажера користувачі зустрічаються з наступним повідомленням: "Тепер, коли ви ознайомились з теорією ArrayList, давайте перевіримо та закріпимо отримані знання на практиці. Ви виконуватимете завдання, які допоможуть вам краще зрозуміти, як працювати з ArrayList в реальних ситуаціях."

Завдання 1. Створення та додавання елементів

Текст завдання: "Створіть ArrayList імен names та додайте до нього три імені: 'John', 'Emily', 'Bob'. Виведіть список на консоль." Приклад коду для вводу:

// Місце для коду

Після виконання завдання:

- Правильна відповідь:

ArrayList<String> names = new ArrayList<>();

names.add("John");

names.add("Emily");

names.add("Bob");

System.out.println(names);

"Вітаємо! Ви правильно створили список і додали елементи. Так тримати!"

- Неправильна відповідь: "Щось пішло не так. Переконайтесь, що ви правильно

створили ArrayList і додали до нього елементи. Пам'ятайте, метод add() використовується для додавання елементів до списку. Спробуйте ще раз."

Завдання 2. Видалення елементів

Текст завдання: "Використовуючи список names з попереднього завдання, видаліть 'Emily' та виведіть оновлений список на консоль." Приклад коду для вводу:

// Місце для коду

Після виконання завдання:

- Правильна відповідь:

  names.remove("Emily");

  System.out.println(names);

  "Чудово! Ви вдало видалили елемент зі списку. Це важливий навик при роботі з динамічними колекціями."

- Неправильна відповідь: "Видалення не відбулось. Переконайтесь, що ви використовуєте метод remove(Object o) для видалення елемента за значенням. Спробуйте знову, і переконайтесь, що вказуєте правильний об'єкт для видалення."

Завдання 3. Доступ до елементів

Текст завдання: "Отримайте та виведіть на консоль другий елемент зі списку names." Приклад коду для вводу:

// Місце для коду

Після виконання завдання:

- Правильна відповідь:

  System.out.println(names.get(1));

  "Ви вірно отримали другий елемент зі списку. Використання методу get(int index) є ключовим для доступу до елементів ArrayList."

- Неправильна відповідь: "Здається, ви спробували отримати елемент неправильно. Пам'ятайте, що індексація в ArrayList починається з нуля. Переконайтесь, що вказуєте правильний індекс. Спробуйте ще раз."

Завдання 4. Копіювання списку

Текст завдання: "Створіть новий ArrayList newNames і скопіюйте в нього всі елементи зі списку names. Після копіювання виведіть новий список на консоль." Приклад коду для вводу:

// Місце для коду

Після виконання завдання:

- Правильна відповідь:

  ArrayList<String> newNames = new ArrayList<>(names);

  System.out.println("Новий список: " + newNames);

"Ви успішно скопіювали список! Копіювання списків є стандартною операцією при роботі з колекціями, коли необхідно зберегти оригінальні дані без змін."

- Неправильна відповідь: "Схоже, що копіювання не було виконано правильно. Переконайтесь, що ви використовуєте конструктор ArrayList для створення копії з існуючого списку. Спробуйте знову, і переконайтесь, що ви не випускаєте інші можливі помилки при копіюванні."

Завдання 5. Перевірка наявності елемента

Текст завдання: "Перевірте, чи містить список names ім'я 'John'. Виведіть повідомлення, що підтверджує наявність або відсутність цього імені у списку." Приклад коду для вводу:

// Місце для коду

Після виконання завдання:

- Правильна відповідь:

```
if (names.contains("John")) {
    System.out.println("Список містить ім'я John.");
} else {
    System.out.println("Список не містить ім'я John.");
}
```

"Чудово! Ви правильно перевірили наявність імені в списку. Вміння використовувати метод contains() дуже важливе при роботі з колекціями для перевірки включення певних елементів."

- Неправильна відповідь: "Переконайтесь, що використовуєте метод contains()

для перевірки наявності елемента у списку. Це дає можливість ефективно управляти даними без необхідності перегляду всіх елементів. Спробуйте ще раз, враховуючи цю пораду."

Завдання 6. Очищення списку

Текст завдання: "Очистіть весь список names та виведіть повідомлення про стан списку після очищення." Приклад коду для вводу:

// Місце для коду

Після виконання завдання:

- Правильна відповідь:

  names.clear();

  System.out.println("Список було очищено.");

"Вітаємо! Ви успішно очистили список. Використання методу clear() є ефективним способом для повного видалення всіх елементів з колекції."

- Неправильна відповідь: "Переконайтесь, що ви використовуєте метод clear() для очищення списку. Цей метод дозволяє повністю видалити всі елементи, тим самим відновлюючи початковий стан колекції. Спробуйте знову, щоб підтвердити, що ви правильно виконуєте операції зі списком."

Тестова частина тренажера з ArrayList:

Вступ до тестової частини: "Тепер, коли ви пройшли через теоретичну та практичну частини нашого тренажера ArrayList, настав час перевірити ваші знання. Ця частина містить серію тестових питань, які допоможуть вам оцінити розуміння ArrayList. Готові почати?"

Питання 1: Створення ArrayList "Який із наступних способів правильно створює ArrayList, що може зберігати об'єкти типу String?"

- A) ArrayList strings = new ArrayList();
- B) ArrayList<String> strings = new ArrayList<String>();
- C) List<String> strings = new ArrayList<>();
- D) ArrayList strings = new ArrayList<String>(); *Правильна відповідь: C*

Питання 2: Додавання елементів "Який метод використовується для додавання елемента в кінець ArrayList?"

- A) append()
- B) insert()
- C) add()
- D) push() *Правильна відповідь: C*

Питання 3: Доступ до елементів "Як отримати доступ до елемента в ArrayList за індексом?"

- A) elementAt(index)
- B) get(index)
- C) at(index)
- D) indexOf(index) *Правильна відповідь: B*

Питання 4: Видалення елемента "Як правильно видалити елемент з ArrayList по індексу?"

- A) list.removeElement(index);
- B) list.delete(index);
- C) list.remove(index);
- D) list.erase(index); *Правильна відповідь: C*

Питання 5: Перевірка наявності елемента "Який метод використовується для перевірки, чи містить ArrayList заданий об'єкт?"

- A) list.contains(object);
- B) list.has(object);
- C) list.includes(object);
- D) list.exists(object); *Правильна відповідь: A*

Питання 6: Зміна елемента "Як змінити елемент у ArrayList за вказаним індексом?"

- A) list.set(index, newValue);
- B) list.update(index, newValue);
- C) list.replace(index, newValue);
- D) list.change(index, newValue); *Правильна відповідь: A*

Питання 7: Очищення ArrayList "Як очистити ArrayList від усіх елементів?"

- A) list.clearItems();

- B) list.removeAll();
- C) list.deleteAll();
- D) list.clear(); *Правильна відповідь: D*

Питання 8: Основний інтерфейс колекцій для ArrayList "Який інтерфейс колекцій є основою для ArrayList?"

- A) Set
- B) Map
- C) List
- D) Queue *Правильна відповідь: C*

Питання 9: Перевірка порожнечі ArrayList "Як перевірити, чи порожній ArrayList?"

- A) list.isEmpty();
- B) list.isClear();
- C) list.hasItems();
- D) list.size() == 0; *Правильна відповідь: A*

Заключення тестової частини: "Ви успішно завершили тестову частину ArrayList. Ми сподіваємося, що цей тест допоміг вам краще зрозуміти, як використовувати ArrayList у вашому коді. Ви можете повернутися до теоретичної або практичної частини для поглиблення знань або спробувати тест знову."

Блок-схеми роботи програми:

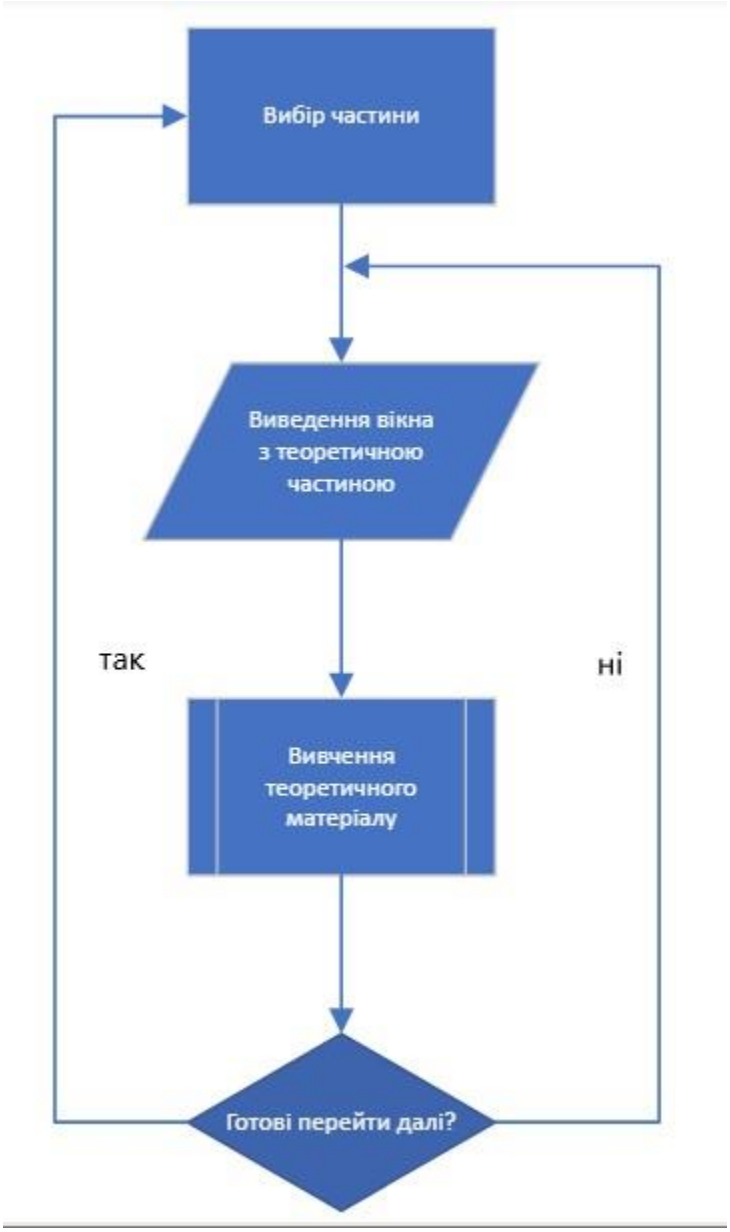

Рисунок 3.1 – Блок-схема теоретичної частини

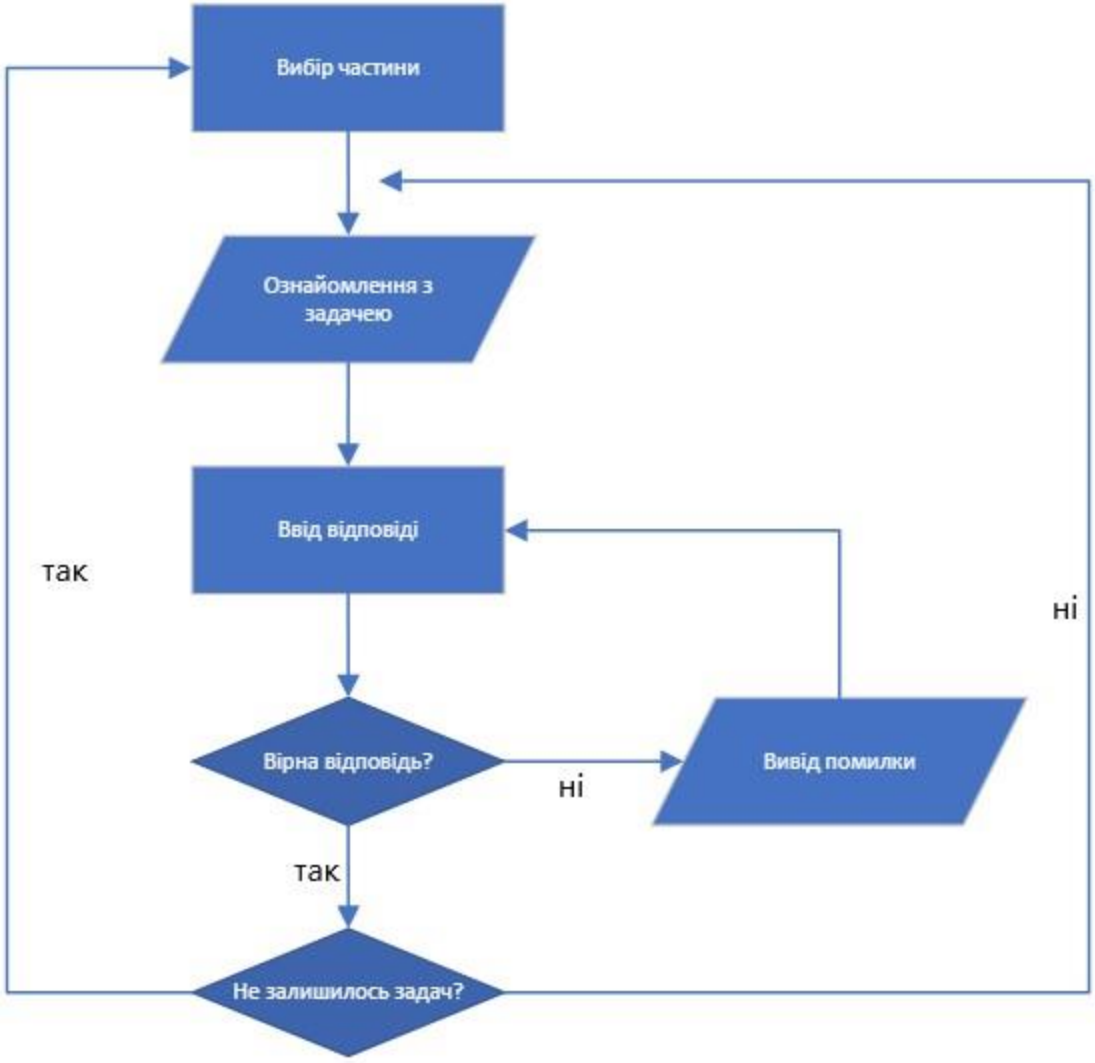

Рисунок 3.2 – Блок-схема практичної частини

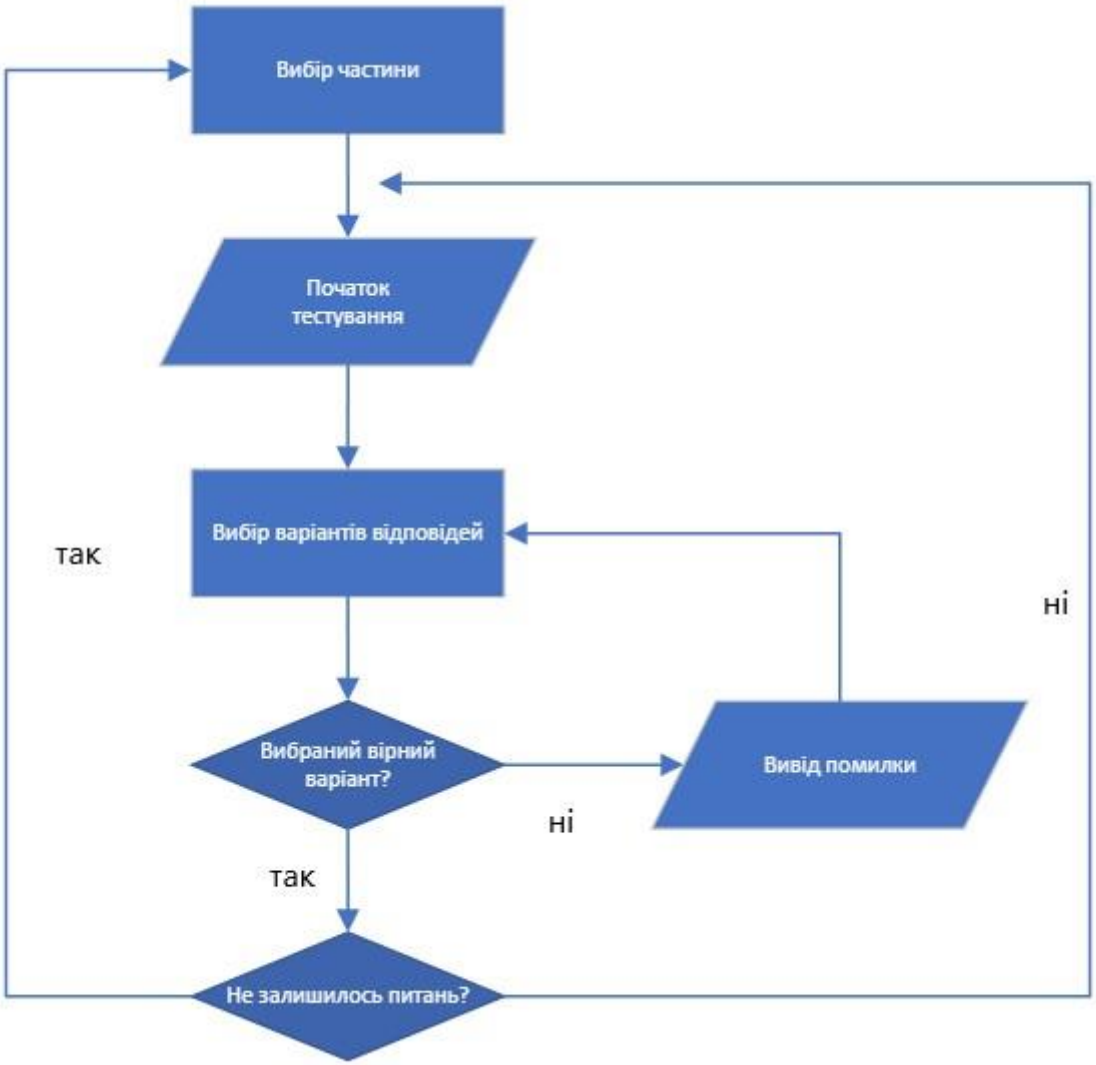

Рисунок 3.3 – Блок-схема тестової частини

4. ПРАКТИЧНА ЧАСТИНА

4.1 Опис процесу програмної реалізації

Розробка програмної реалізації тренажера з ArrayList для навчання основам роботи з колекціями в Java була виконана в декілька етапів. Ключові моменти цього процесу охоплюють планування, проектування інтерфейсу, програмування функціональності, тестування і документація.

Планування

На етапі планування було визначено основні вимоги до функціональності тренажера. Важливим аспектом стало забезпечення інтуїтивно зрозумілого інтерфейсу користувача, що дозволяє з легкістю переходити між теоретичними уроками, практичними завданнями та блоком тестування. Також було вирішено використовувати Java як основну мову програмування через її широке застосування в освітніх програмах з програмування.

Проектування інтерфейсу користувача

Процес проектування інтерфейсу для тренажера з ArrayList був зосереджений на створенні інтуїтивно зрозумілого та легкого у використанні інтерфейсу, що дозволяє користувачам ефективно взаємодіяти з навчальним матеріалом. Розробка інтерфейсу включала наступні аспекти:

Структура інтерфейсу

Інтерфейс програми було розділено на три основні секції, які відповідають основним модулям тренажера:

1. Теоретична частина - модуль для вивчення теоретичного матеріалу.
2. Практична частина - модуль для виконання практичних завдань.
3. Тестування - модуль для перевірки знань через тести з множинним вибором.

Компоненти інтерфейсу

- Головне меню дозволяє користувачу вибирати між теорією, практикою, і тестуванням. Кожен вибір веде до відповідного розділу програми.
- Текстові поля (JTextArea), використовуються для відображення інформаційного контенту і прийому відповідей від користувачів.

- Кнопки (JButton) забезпечують навігацію між різними частинами програми і виконання дій, таких як подання відповідей на завдання або перевірку тестів.
- Панелі (JPanel) використовуються для групування візуальних елементів із схожою функціональністю, таких як блоки з питаннями у тестовому модулі або групи контролів у практичній частині.

Дизайн інтерфейсу

Дизайн інтерфейсу підкреслює чіткість та простоту, з використанням зрозумілих ікон та контрольних елементів, щоб користувачі не відчували зайвого стресу від навігації по програмі. Інтерфейс адаптовано для користувачів з різним рівнем досвіду, включаючи початківців.

Візуальні елементи

Для кожного модуля використовувались різні кольорові схеми для забезпечення візуального розмежування між навчальними активностями. Використання іконок і кольорових маркерів допомагає користувачам швидше зорієнтуватися в матеріалах курсу.

Реакція на дії користувача

Інтерфейс надає негайний зворотний зв'язок на дії користувача, такі як виправлення помилок у коді, правильні або неправильні відповіді у тестах, і прогрес у навчальних модулях. Це допомагає утримувати увагу користувача та підвищує мотивацію до навчання.

Програмування функціональності

Програмування функціональності тренажера ArrayList передбачало створення набору інтерактивних компонентів та функцій, що дозволяють користувачам ефективно засвоювати теорію, практикуватись і тестувати свої знання. Основні аспекти програмування включали:

Взаємодія з користувачем

Розробка інтерактивних елементів управління забезпечувала зв'язок між користувачем і програмою, включаючи:

- Текстові поля для введення коду або відповідей.
- Кнопки для переходу між різними частинами тренажера та подання відповідей.

- Вибір завдань через меню або навігаційні панелі.

Обробка введення

Програмування логіки обробки введення користувача було критично для забезпечення адекватної реакції на дії користувача:

- Валідація відповідей в режимі реального часу, з миттєвим зворотнім зв'язком.
- Обробка помилок у коді користувача з підказками для виправлення.

Логіка тестування

Для модуля тестування було розроблено систему оцінювання з можливістю вибору відповідей та автоматичної перевірки:

- Менеджмент сесій, що дозволяє зберігати прогрес користувача між сесіями.
- Генерація звіту з результатами для оцінювання ефективності навчання користувача.

Програмування інтерфейсу

Використання бібліотеки Swing у Java для розробки графічного інтерфейсу, який включав:

- Створення інтерактивних форм для введення відповідей та навігації.
- Анімація та візуальні ефекти для поліпшення залученості та інтерактивності тренажера.

4.2 Опис програми

Програма-тренажер, створена для навчання роботі з ArrayList у мові програмування Java, є комплексним рішенням, яке забезпечує користувачам можливість вивчення теоретичних основ та практичного застосування структур даних через інтерактивний підхід. Опис програми охоплює її архітектуру, основні компоненти, а також функціональність, що вона надає.

Архітектура програми

Програма розроблена з використанням мови програмування Java, зокрема, із застосуванням бібліотеки Swing для створення графічного користувацького інтерфейсу (GUI). Архітектура програми має модульну структуру:

1. Модуль теоретичної частини - надає користувачам структуровані теоретичні матеріали про ArrayList.
2. Модуль практичних завдань - дозволяє користувачам виконувати різні завдання на основі вивченого матеріалу.
3. Тестовий модуль - забезпечує можливість перевірки знань через серію тестових питань із множинним вибором.

Основні компоненти

- Головне вікно - стартова точка програми, що надає навігацію між різними частинами тренажера.
- Інтерфейс користувача - містить елементи управління, такі як кнопки, текстові поля, меню вибору завдань і тести.
- Система зворотного зв'язку - автоматично надає користувачам повідомлення про успішне виконання завдань або помилки в рішеннях.

Функціональність

- Вивчення теорії - користувачі можуть читати теоретичні матеріали, що супроводжуються прикладами коду.
- Виконання практичних завдань - користувачі можуть писати та тестувати код прямо у програмі.
- Тестування - в кінці курсу користувачі можуть перевірити свої знання, виконуючи тести з вибором правильної відповіді.
- Звітність - здатність генерувати звіти про результати користувачів для викладачів або самооцінки.

Засоби розробки

- IDE: Apache NetBeans IDE 21
- Контроль версій: Git для управління змінами в коді.
- Тестування: JUnit для написання і виконання юніт-тестів.

4.3 Перевірка валідності

Перевірка валідності програми-тренажера з теми "ArrayList" у Java є критично важливим етапом розробки, який забезпечує коректність роботи програми та її

відповідність визначеним вимогам. Цей процес включає в себе ряд тестів і перевірок, що дозволяють виявити та усунути потенційні помилки і недоліки у програмі перед її впровадженням у навчальний процес.

Типи перевірок:

1. Функціональне тестування: Перевірка кожної функції програми на відповідність специфікаціям. Це включає в себе тестування усіх функцій, що забезпечують роботу з ArrayList, включаючи додавання, видалення елементів, доступ до елементів та їх копіювання.
2. Тестування інтерфейсу користувача: Перевірка зручності та інтуїтивності інтерфейсу, забезпечення того, що користувач легко може навігувати між різними частинами програми та ефективно використовувати її функціонал.
3. Юніт-тестування: Розробка та виконання юніт-тестів для ключових компонентів програми, зокрема для методів управління списком ArrayList. Це допомагає переконатися, що логіка програми працює належним чином.
4. Інтеграційне тестування: Перевірка взаємодії між різними модулями програми, щоб забезпечити їх правильну взаємодію та співпрацю.
5. Тестування валідності вводу: Переконатися, що програма коректно обробляє некоректні вводи, забезпечуючи належні повідомлення про помилки та запобігаючи будь-якій неправильній поведінці програми.

Методи тестування:

- Ручне тестування: Виконання тестів вручну для перевірки унікальних аспектів інтеракції з користувачем, які можуть бути важко автоматизувати.
- Автоматичне тестування: Використання інструментів автоматичного тестування, таких як JUnit, для написання та виконання тестових сценаріїв, що забезпечують широке покриття коду.

Результати тестування документуються для подальшого аналізу та перевірки. У випадку виявлення помилок розробники негайно вживають заходів для їх виправлення, перепроводячи тестування для переконання у їх повному усуненні. Завершальний етап тестування передбачає повторне проведення усіх тестів для підтвердження стабільності програми після усіх змін.

4.4 Інструкція користувача

Інструкція користувача для програми "ArrayList Trainer" призначена для навчання основам роботи зі структурою даних ArrayList у мові програмування Java. Нижче наведено поетапні інструкції з користування програмою.

Екран вітання та вибір розділів

1. При запуску програми видно головне вікно, де представлено привітання та короткий опис програми(див рис. 4.1).

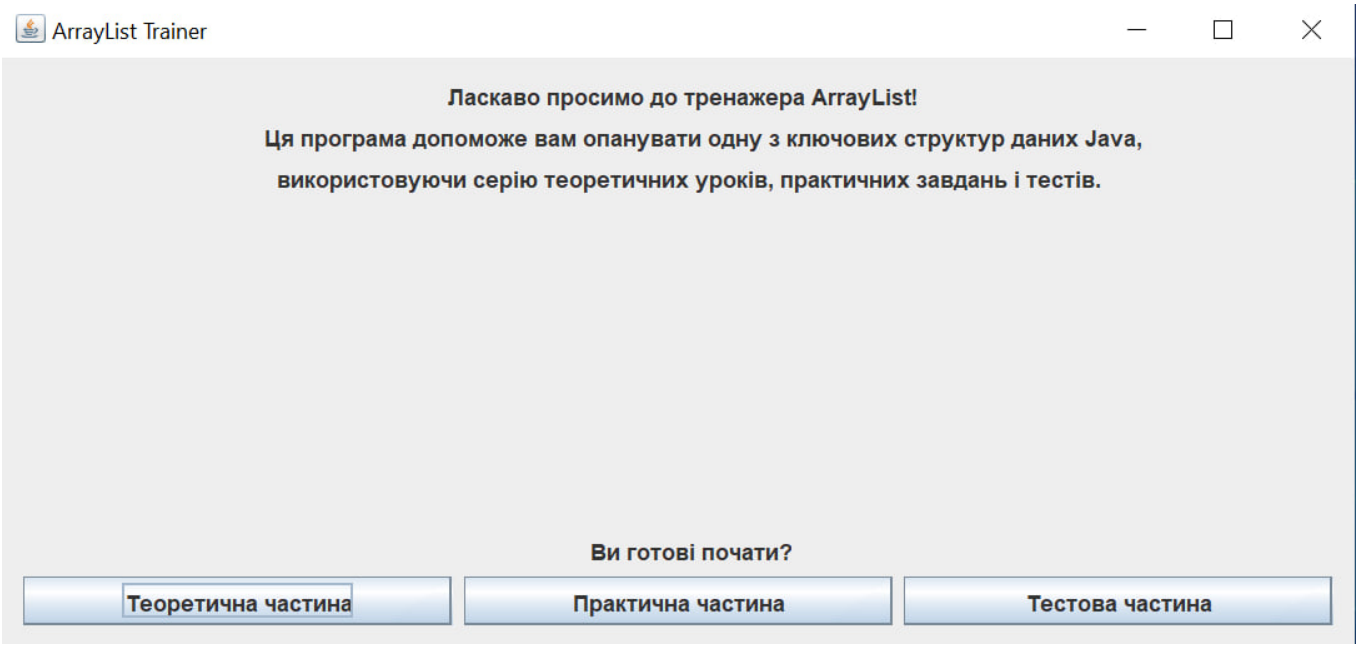

Рисунок 4.1 – Головне вікно

2. На головному екрані виберіть один із трьох розділів для роботи: "Теоретична частина", "Практична частина", "Тестова частина". Використання кнопок дозволяє швидко перейти до потрібного розділу.

Теоретична частина

1. При виборі "Теоретична частина" на екрані з’являється інтерфейс з текстовим полем, де відображається теоретичний матеріал про ArrayList(див рис. 4.2).

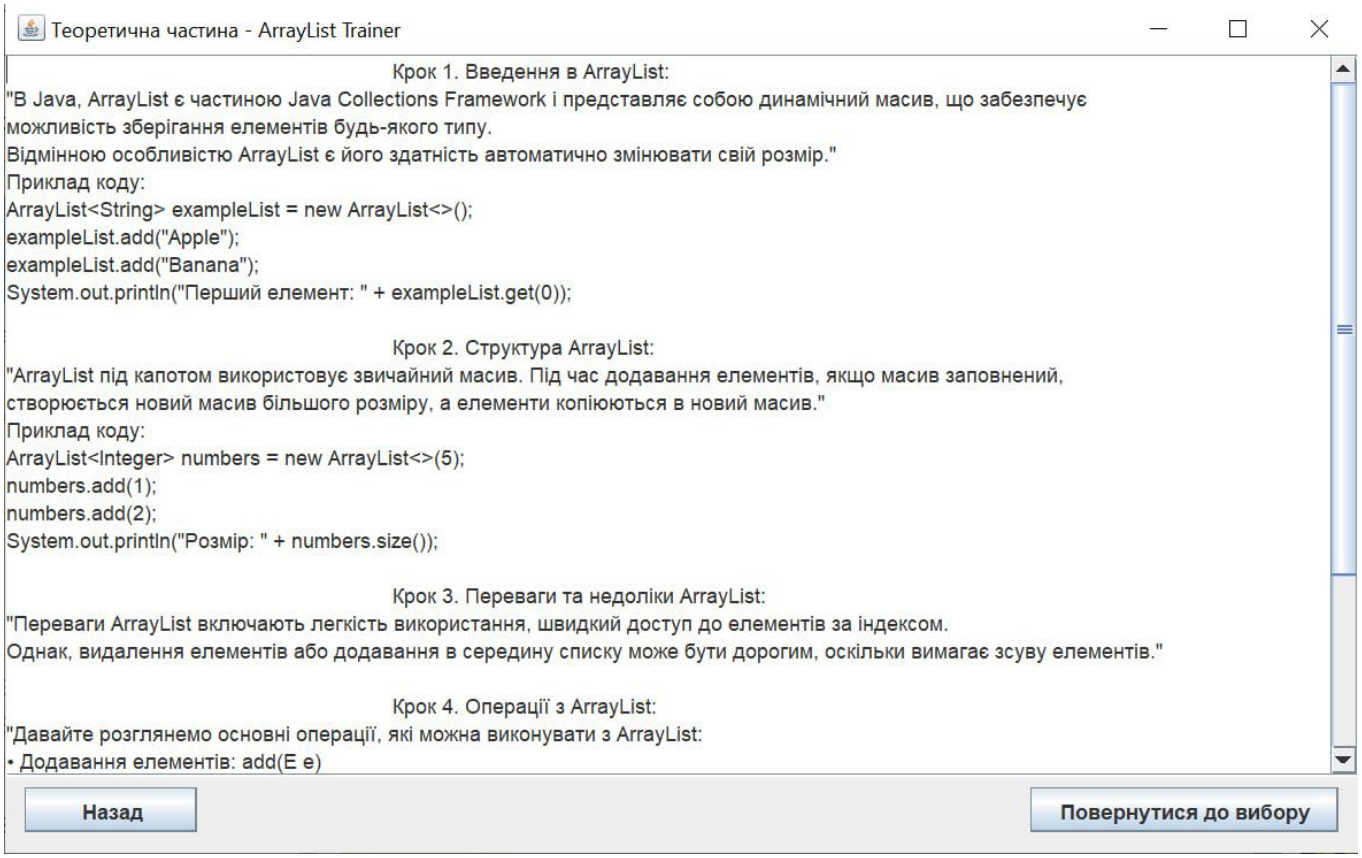

Рисунок 4.2 – Теоретична частина

2. Матеріал поділений на кілька кроків, що детально описують різні аспекти роботи з ArrayList:
    - **Крок 1:** Введення в ArrayList. Опис основних властивостей.
    - **Крок 2:** Структура ArrayList і спосіб роботи з елементами.
    - **Крок 3:** Переваги та недоліки використання ArrayList.
    - **Крок 4:** Операції з ArrayList, як додавання та видалення елементів.
3. Після проходження всіх кроків з’являється діалогове вікно з повідомленням про успішне завершення теоретичної частини та пропозицією перейти до практичної частини або тестування(див рис. 4.3).

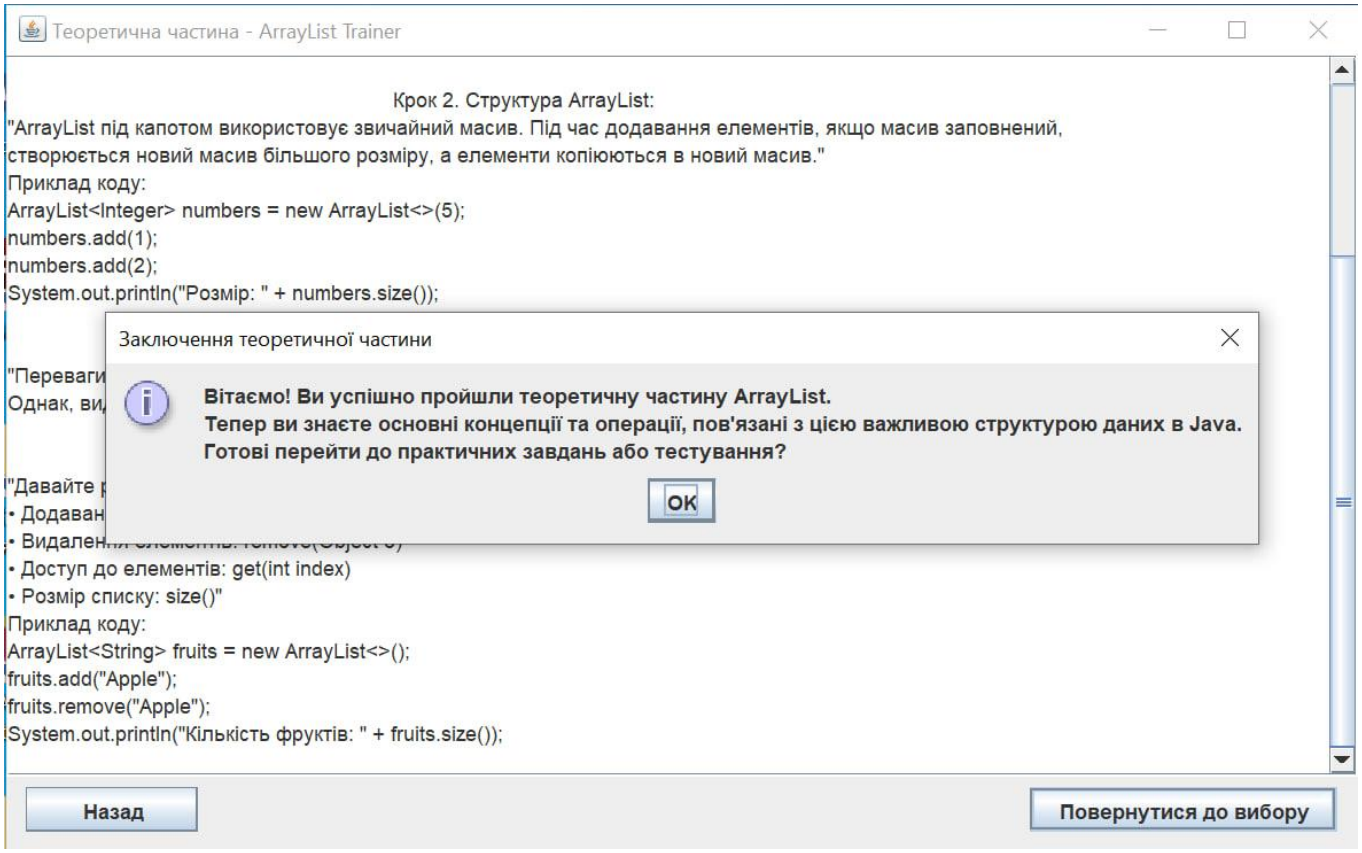

Рисунок 4.3 – Завершення теоретичної частини

Головна сторінка тренажера

Крок 1: Почніть із головної сторінки тренажера, де вам буде запропоновано вибрати між теоретичною частиною, практичною частиною, та тестовою частиною тренажера(див рис. 4.1).

Вступ до практичної частини

Крок 2: Для переходу до практичної частини натисніть на кнопку "Практична частина". З'явиться повідомлення, що мотивує перевірити теоретичні знання на практиці. Натисніть "OK" для продовження(див рис 4.4).

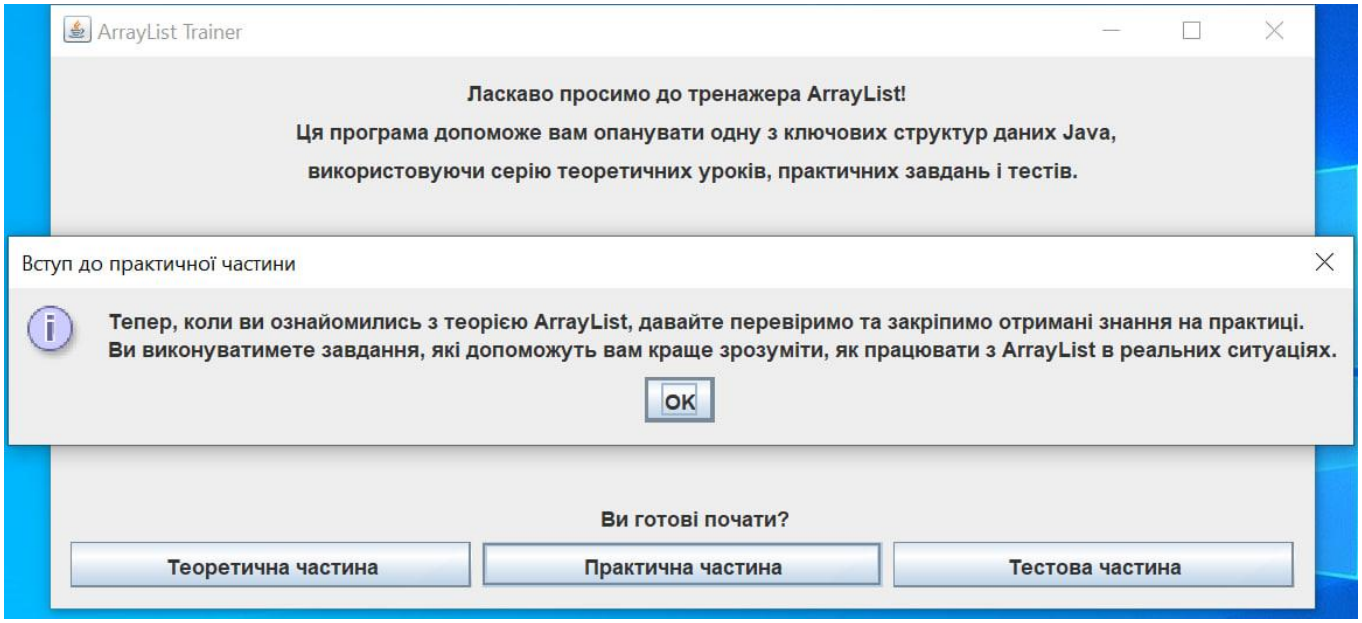

Рисунок 4.4 – Вступ до практичної частини

Завдання 1-2: Створення, додавання та видалення елементів

Кроки 3-4:

- Створіть ArrayList імен names та додайте до нього імена: 'John', 'Emily', 'Bob'. Після додавання видаліть 'Emily' із списку. Введіть ваш код у текстове поле та натисніть "Перевірити" для оцінки вашого коду(див рис. 4.5).
- Підказки:
  - Перевірте синтаксис та методи, які ви використовуєте для додавання і видалення елементів.
  - Зверніть увагу на правильне використання методу remove() для видалення елемента по значенню.

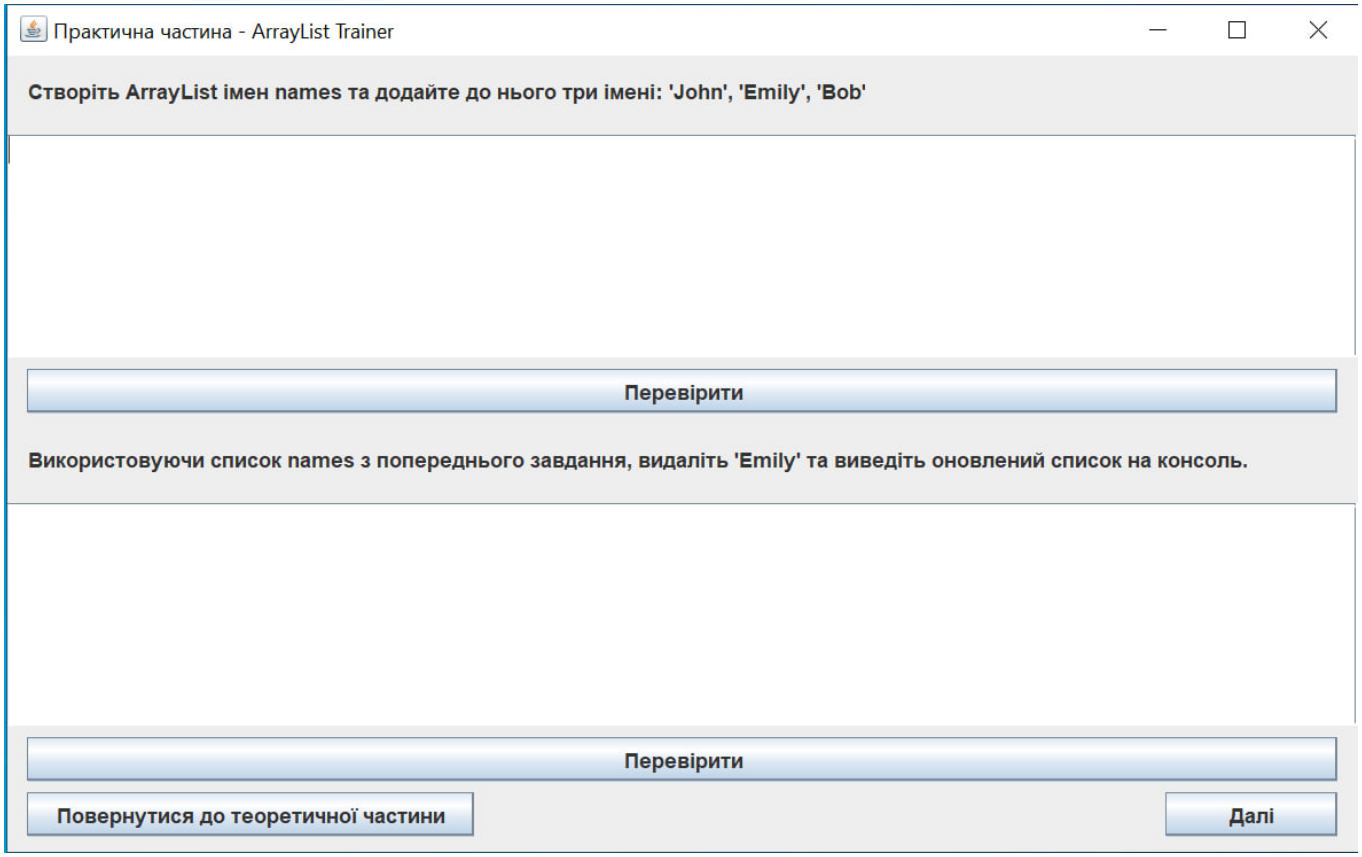

Рисунок 4.5 – Перша сторінка практичної частини

Завдання 3-4: Доступ та копіювання елементів

Кроки 5-6:

- Отримайте та виведіть другий елемент зі списку names, потім створіть новий ArrayList newNames і скопіюйте в нього всі елементи зі списку names. Введіть відповідний код у текстове поле і натисніть "Перевірити" для оцінки(див рис. 4.6).
- Підказки:
  - Використовуйте метод get() для доступу до елементів за індексом, який починається з нуля.
  - Для копіювання списку використовуйте конструктор копіювання ArrayList<>(існуючийСписок).

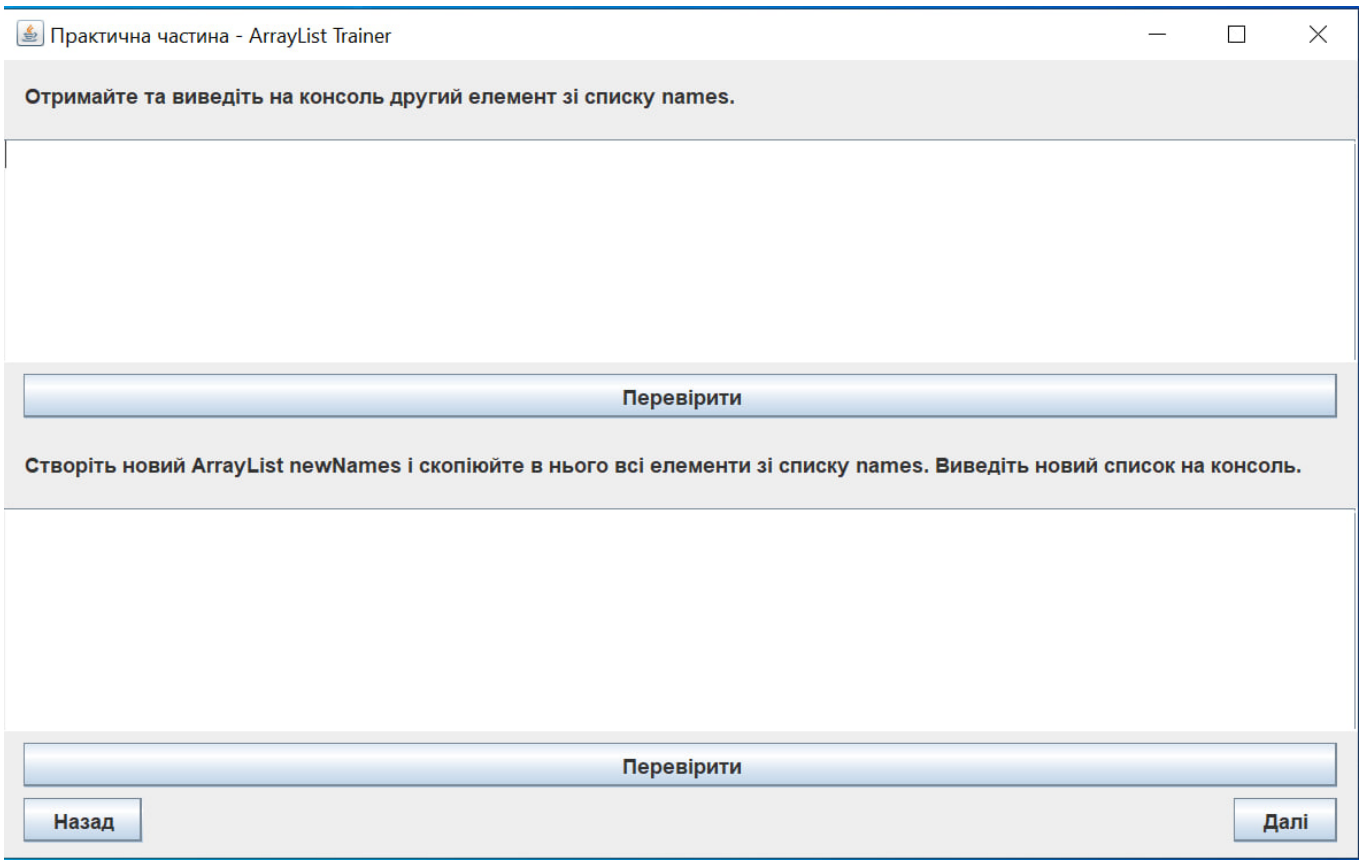

Рисунок 4.6 – Друга сторінка практичної частини

Завдання 5-6: Перевірка наявності та очищення списку

Кроки 7-8:

- Перевірте наявність імені 'John' у списку names, використовуючи метод contains(). Якщо ім'я є, виведіть "John присутній", інакше - "John відсутній". Після цього очистіть весь список за допомогою методу clear() та виведіть список після очищення(див рис. 4.7).
- Підказки:
  - Використання умовної конструкції if допоможе визначити, чи існує елемент у списку.
  - Використовуйте clear() для очищення списку та System.out.println(names) для відображення порожнього списка.

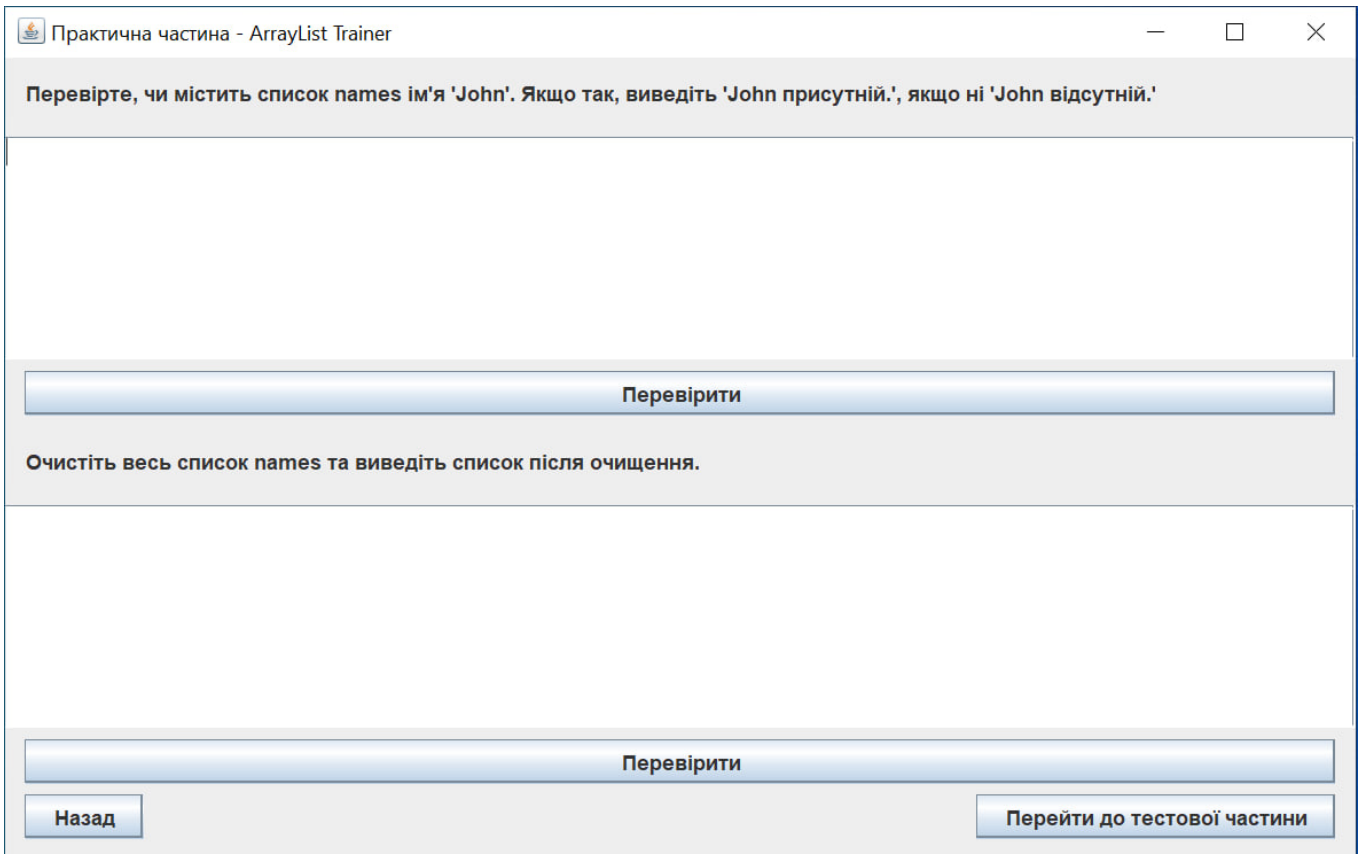

Рисунок 4.7 – Третя сторінка практичної частини

Закінчення практичної частини

Крок 9:

- Після виконання усіх завдань з'явиться повідомлення про успішне завершення практичної частини. Натисніть "OK" для переходу до тестової частини(див рис. 4.8).

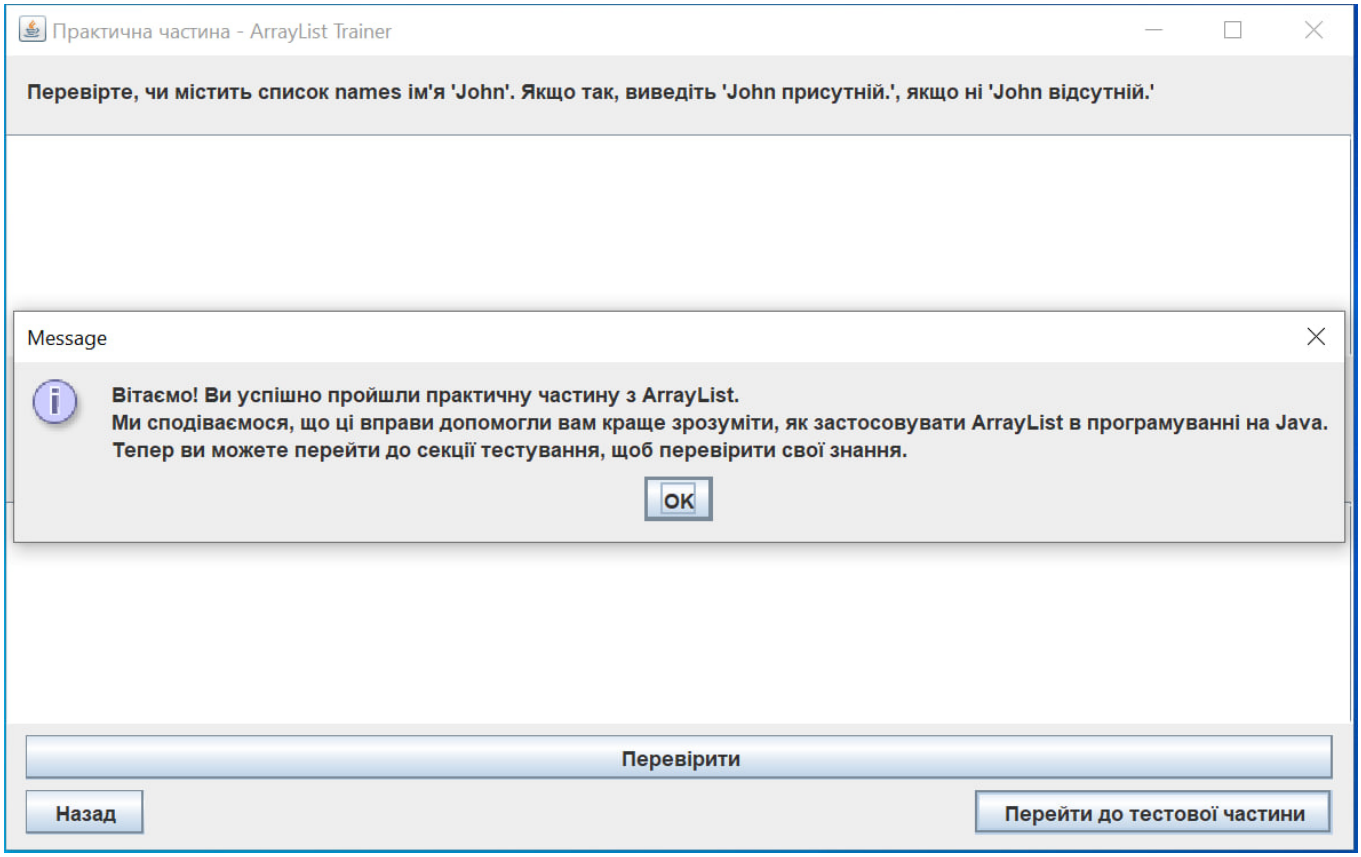

Рисунок 4.8 – Завершення практичної частини

Вступ до тестової частини

1. Перехід до тестової частини: Після завершення практичних завдань ви побачите діалогове вікно, яке повідомляє про можливість перевірити набуті знання в тестовій частині. Натисніть "ОК" для переходу до тестів(див рис. 4.9).

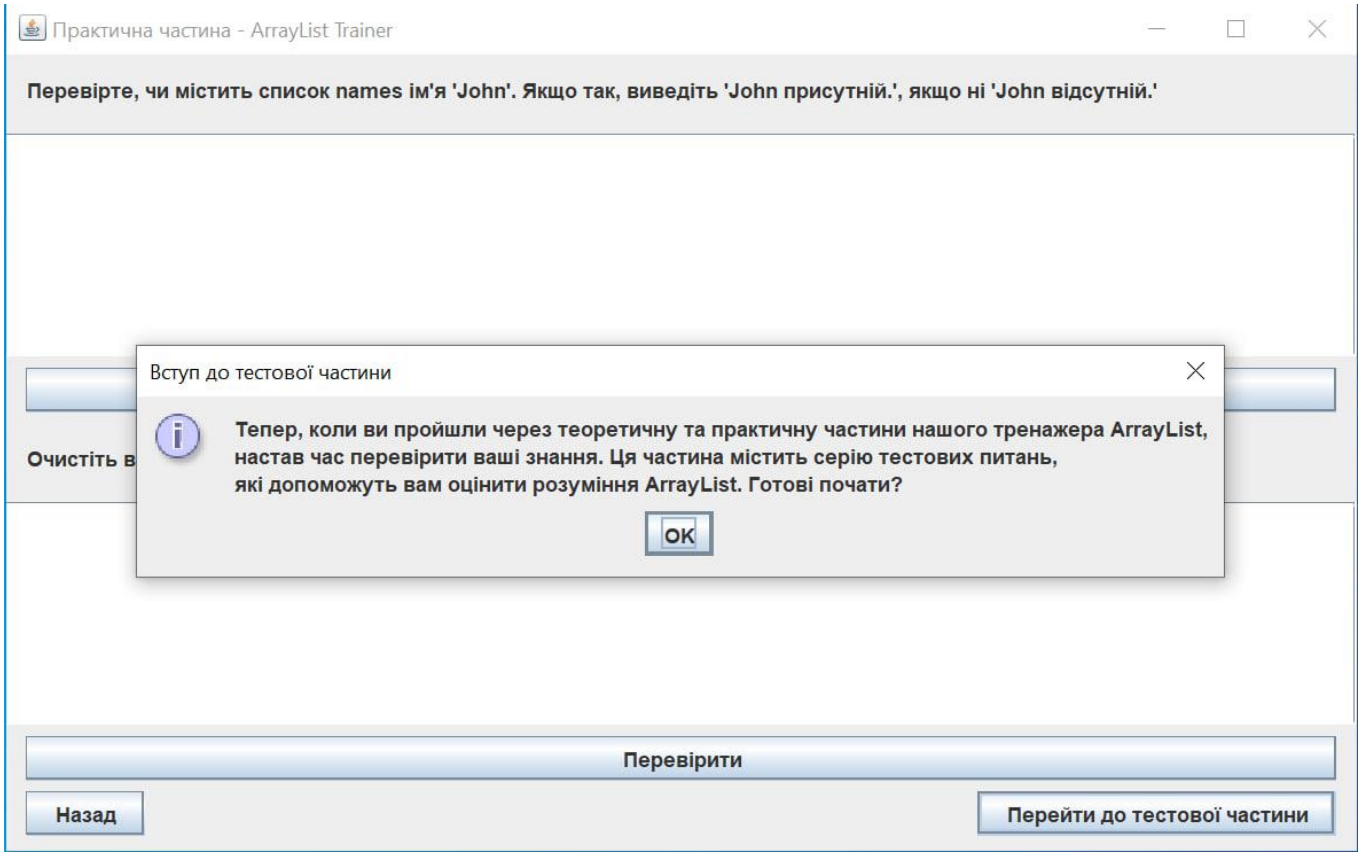

Рисунок 4.9 – Перехід до тестової частини

Проходження тесту

2. Виконання тестів:
    - На кожному етапі тестування вам будуть запропоновані питання з кількома варіантами відповідей. Виберіть відповідь, яка, на вашу думку, є правильною, натискаючи на відповідний радіобатон поруч з текстом відповіді.
    - Питання охоплюють різні аспекти роботи з ArrayList, включаючи створення списків, додавання та видалення елементів, а також перевірку вмісту і структури даних(див рис. 4.10-4.12).

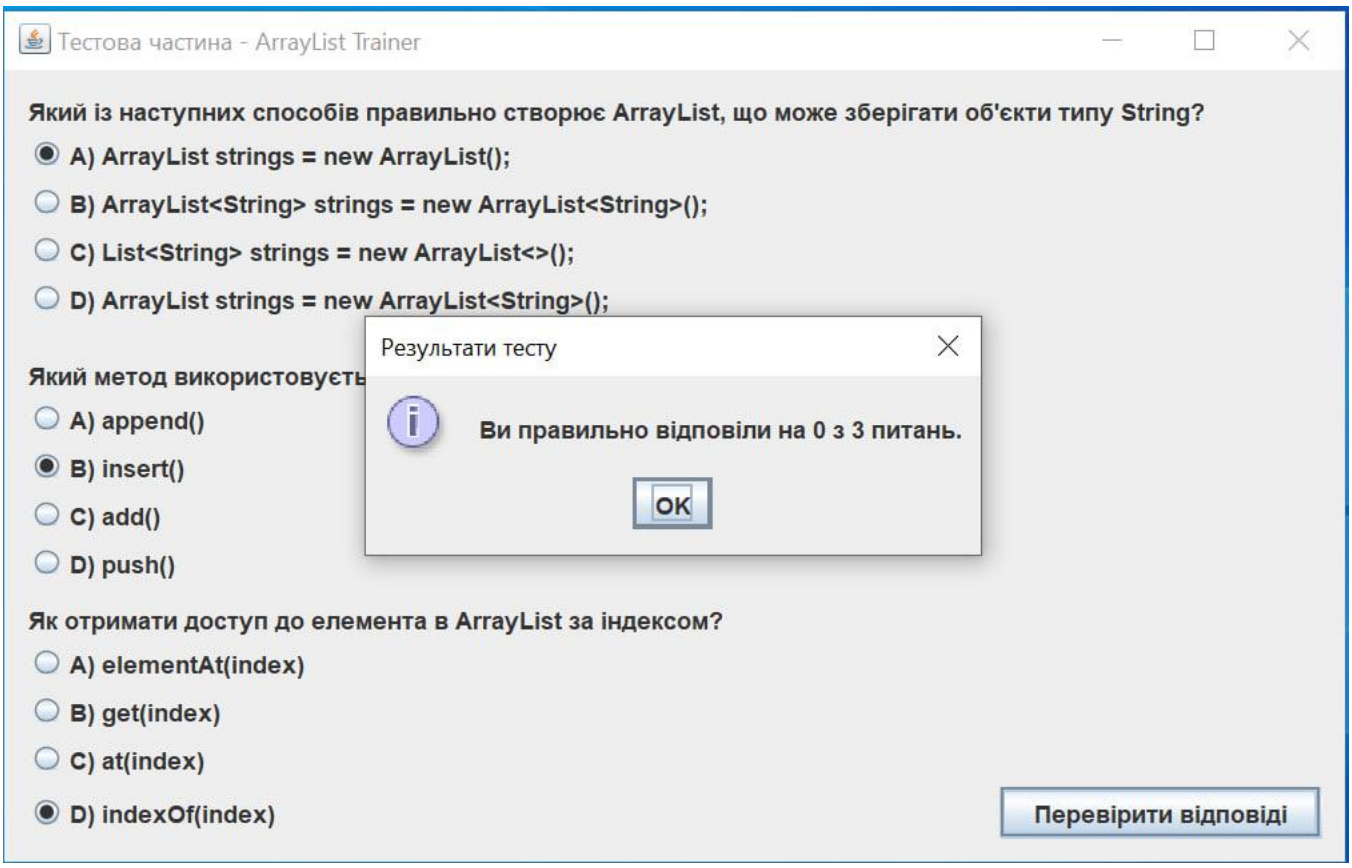

Рисунок 4.10 – Перша сторінка тестової частини

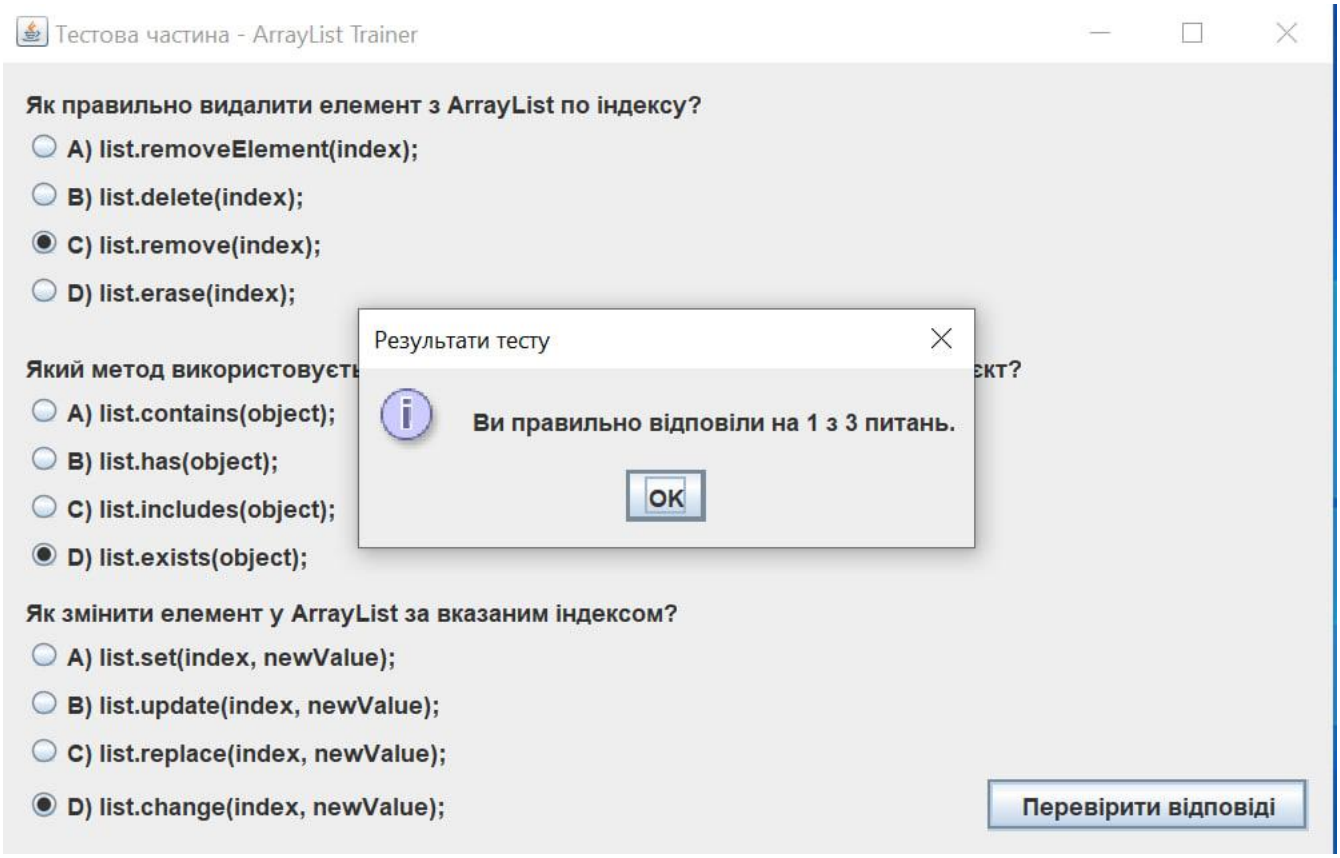

Рисунок 4.11 – Друга сторінка тестової частини

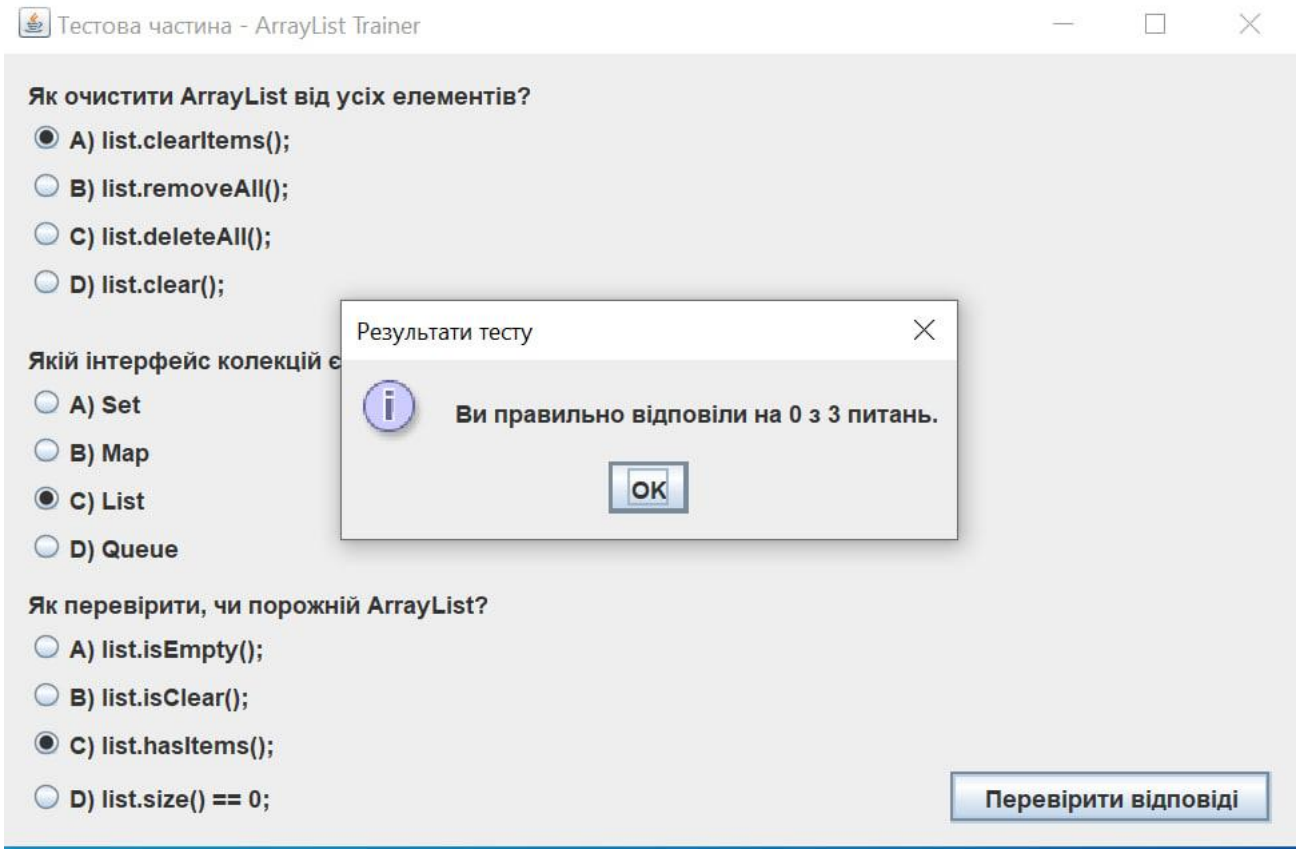

Рисунок 4.12 – Третя сторінка тестової частини

Перевірка відповідей

3. Перевірка відповідей:
    - Після вибору відповідей на всі питання натисніть кнопку "Перевірити відповіді" для отримання результатів.
    - Система автоматично оцінить ваші відповіді та надасть зворотний зв'язок, показуючи, скільки питань ви відповіли правильно.

Заключення тестової частини

4. Заключення тестової частини:
    - Після перевірки відповідей з'явиться діалогове вікно з подякою за проходження тестування та пропозицією повернутися до головної сторінки тренажера.
    - Натисніть "ОК" для завершення тестування або "Назад" для повторення тесту або перегляду теоретичних матеріалів(див рис. 4.13).

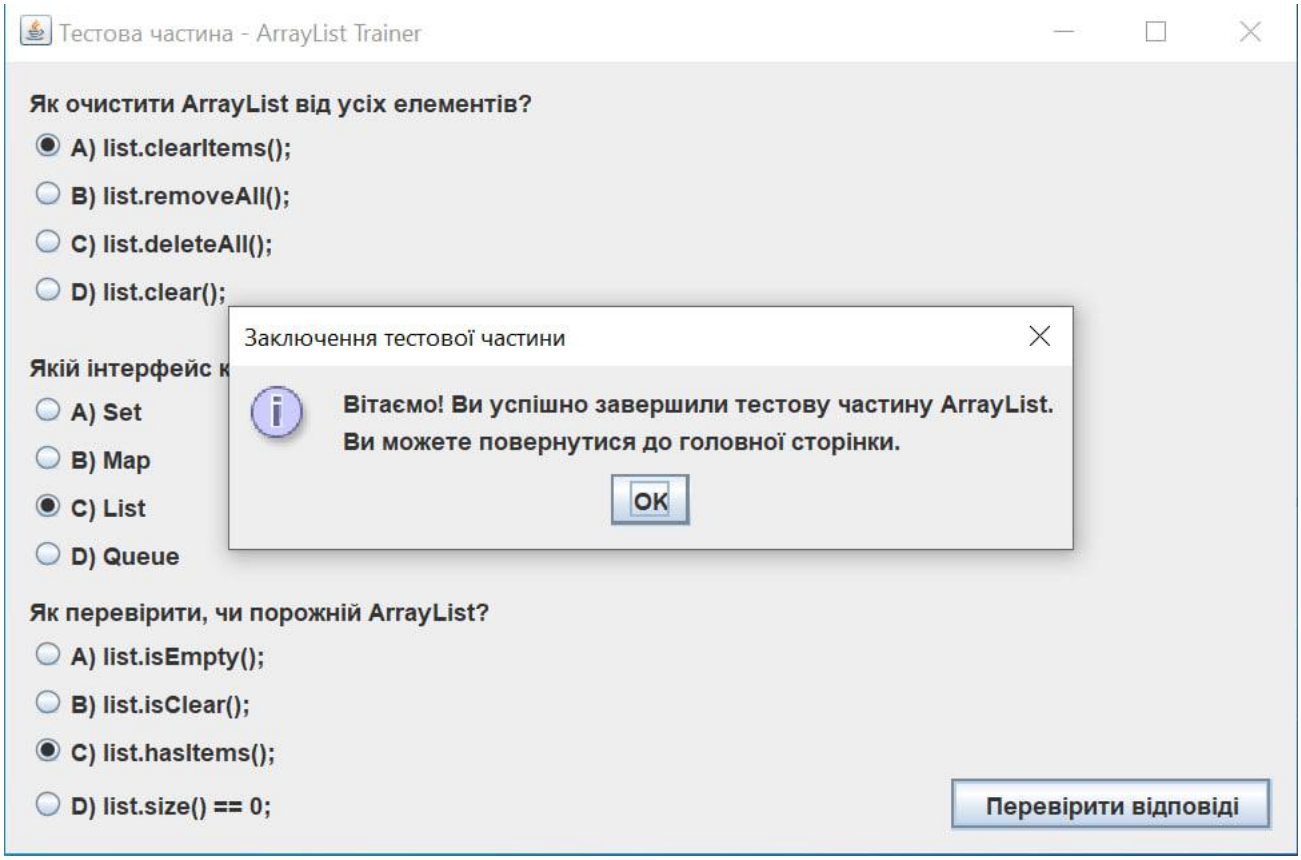

Рисунок 4.13 – Завершення тестової частини

ВИСНОВКИ

Під час виконання цієї кваліфікаційної роботи було здійснено повний цикл розробки тренажера для навчання роботи з колекціями Java, зокрема з ArrayList. Основна увага приділялася практичному застосуванню теоретичних знань з колекцій у програмуванні, що включало розробку інтерактивних завдань та тестів.

Результати роботи включають:

1. Інформаційний огляд: Було проведено аналіз існуючих методик викладання та практичного застосування колекцій у програмуванні. Розглянуто літературні джерела та електронні ресурси, що стосуються тренажерів і навчальних програм.
2. Теоретична розробка: Формулізовано основні принципи роботи з ArrayList у Java, описано можливості та особливості цієї структури даних.
3. Розробка програмної частини: Виконано програмування тренажера, який включає теоретичні матеріали, практичні завдання та блок тестування.
4. Перевірка та тестування: Реалізовано перевірку функціональності програми, що включає тестування всіх компонентів тренажера, з метою виявлення та усунення помилок.
5. Інструкція користувача: Розроблено детальну інструкцію для користувачів, що описує кроки використання тренажера, включаючи огляд функцій та поради з ефективного використання програми.

Тренажер був успішно інтегрований у навчальний процес, що дозволило забезпечити студентам зручний і ефективний інструмент для вивчення та закріплення знань по роботі з колекціями у Java. Ця робота демонструє значний потенціал застосування інтерактивних тренажерів у процесі дистанційного навчання та самостійного освоєння програмування.

СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Horstmann C. S. *Core Java Volume I – Fundamentals*. 11th ed. Prentice Hall, 2018. 928 с.
2. Bloch J. *Effective Java*. 3rd ed. Addison-Wesley Professional, 2018. 416 с.
3. Lafore R. *Data Structures and Algorithms in Java*. 2nd ed. Sams Publishing, 2002. 800 с.
4. Sedgewick R., Wayne K. *Algorithms*. 4th ed. Addison-Wesley Professional, 2011. 976 с.
5. Oracle. *The Java Tutorials* [Електронний ресурс]. – Режим доступу: https://docs.oracle.com/javase/tutorial/collections/index.html.
6. Eckel B. *Thinking in Java*. 4th ed. Prentice Hall, 2006. 1150 с.
7. Яковлєв В.П., Блинов С.В., Романчик В.П. *Основи програмування на мові Java*. Навчальний посібник. – Мінськ: БГУІР, 2015. – 405 с.
8. Шилдт Г. *Повне керівництво Java*. 10-те вид. Видавництво "Діалектика", 2018. 1488 с.
9. Фаулер М. *Рефакторинг: поліпшення існуючого коду*. 2-ге вид. Addison-Wesley Professional, 2018. 448 с.
10. McConnell S. *Code Complete*. 2nd ed. Microsoft Press, 2004. 960 с.
11. Васильєв А.Н. *Технологія розробки програмного забезпечення*. Навчальний посібник. – СПб.: БХВ-Петербург, 2004. – 608 с.
12. Паттерсон Д., Хеннессі Дж. *Архітектура комп'ютера*. 5-те вид. Видавництво "Вільямс", 2012. 856 с.
13. Martin R. C. *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall, 2008. 464 с.
14. McConnell S. *Rapid Development: Taming Wild Software Schedules*. Microsoft Press, 1996. 680 с.
15. Myers G. J., Sandler C., Badgett T. *The Art of Software Testing*. 3rd ed. Wiley, 2011. 432 с.
16. Gamma E., Helm R., Johnson R., Vlissides J. *Design Patterns: Elements of Reusable*

Object-Oriented Software. Addison-Wesley Professional, 1994. 395 c.

ДОДАТОК А.

1. MainForm

```
package com.kolobov.arraylist;

public class MainForm extends javax.swing.JFrame {

    public MainForm() {
        initComponents();
        setTitle("ArrayList Trainer");
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnPractice = new javax.swing.JButton();
        btnTest = new javax.swing.JButton();
        lblWelcome = new javax.swing.JLabel();
        btnTheory = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        btnPractice.setText("Практична частина");
        btnPractice.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnPracticeActionPerformed(evt);
            }
        });

        btnTest.setText("Тестова частина");
        btnTest.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnTestActionPerformed(evt);
```

```
        }
    });

    lblWelcome.setText("Ласкаво просимо до тренажера ArrayList!");

    btnTheory.setText("Теоретична частина");
    btnTheory.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            btnTheoryActionPerformed(evt);
        }
    });

    jLabel1.setText("Ця програма допоможе вам опанувати одну з ключових структур даних Java,");

    jLabel2.setText("використовуючи серію теоретичних уроків, практичних завдань і тестів.");

    jLabel3.setText("Ви готові почати?");

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(btnTheory, javax.swing.GroupLayout.PREFERRED_SIZE, 229,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(btnPractice, javax.swing.GroupLayout.PREFERRED_SIZE, 229,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(btnTest, javax.swing.GroupLayout.PREFERRED_SIZE, 229,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                    .addGap(0, 0, Short.MAX_VALUE)
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                    .addComponent(jLabel3)
                    .addGap(301, 301, 301))
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,  10,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel2)
                            .addGap(22, 22, 22))
                        .addComponent(jLabel1)
                        .addGroup(layout.createSequentialGroup()
                            .addGap(98, 98, 98)
                            .addComponent(lblWelcome)))
                    .addGap(114, 114, 114))))))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(lblWelcome)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 183, Short.MAX_VALUE)
                .addComponent(jLabel3)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnTheory)
                    .addComponent(btnPractice)
                    .addComponent(btnTest))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>
```

```
    private void btnPracticeActionPerformed(java.awt.event.ActionEvent evt) {
        PracticeForm practiceForm = new PracticeForm();
        practiceForm.setVisible(true);
        this.setVisible(false);
    }

    private void btnTestActionPerformed(java.awt.event.ActionEvent evt) {
        TestForm testForm = new TestForm();
        testForm.setVisible(true);
        this.setVisible(false);
    }

    private void btnTheoryActionPerformed(java.awt.event.ActionEvent evt) {
        TheoryForm theoryForm = new TheoryForm();
        theoryForm.setVisible(true);
        this.setVisible(false);
    }

    public static void main(String args[]) {

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MainForm().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnPractice;
    private javax.swing.JButton btnTest;
    private javax.swing.JButton btnTheory;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel lblWelcome;
    // End of variables declaration
}
```

2. TheoryForm:

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;

public class TheoryForm extends javax.swing.JFrame {

    public TheoryForm() {
        initComponents();
        setTitle("Теоретична частина - ArrayList Trainer");
        setSize(800, 500);
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jScrollPane2 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jTextArea1.setText("\t\t\tКрок 1. Введення в ArrayList:\n\"В Java, ArrayList є частиною Java Collections Framework і представляє собою динамічний масив, що забезпечує \nможливість зберігання елементів будь-якого типу. \nВідмінною особливістю ArrayList є його здатність автоматично змінювати свій розмір.\"\nПриклад коду:\nArrayList<String> exampleList = new ArrayList<>();\nexampleList.add(\"Apple\");\nexampleList.add(\"Banana\");\nSystem.out.println(\"Перший елемент: \" + exampleList.get(0));\n\n\t\t\tКрок 2. Структура ArrayList:\n\"ArrayList під капотом використовує звичайний масив. Під час додавання елементів, якщо масив заповнений, \nстворюється новий масив більшого розміру, а елементи копіюються в новий масив.\"\nПриклад коду:\nArrayList<Integer> numbers = new ArrayList<>(5);\nnumbers.add(1);\nnumbers.add(2);\nSystem.out.println(\"Розмір: \" + numbers.size());\n\n\t\t\tКрок 3. Переваги та недоліки ArrayList:\n\"Переваги ArrayList включають легкість
```

```
використання, швидкий доступ до елементів за індексом. \nОднак, видалення елементів або додавання в середину списку може бути дорогим, оскільки вимагає зсуву елементів.\"\n\n\t\t\tКрок 4. Операції з ArrayList:\n\"Давайте розглянемо основні операції, які можна виконувати з ArrayList:\n• Додавання елементів: add(E e)\n• Видалення елементів: remove(Object o)\n• Доступ до елементів: get(int index)\n• Розмір списку: size()\"\nПриклад коду:\nArrayList<String> fruits = new ArrayList<>();\nfruits.add(\"Apple\");\nfruits.remove(\"Apple\");\nSystem.out.println(\"Кількість фруктів: \" + fruits.size());\n");
        jTextArea1.setCaretPosition(0);
        jScrollPane2.setViewportView(jTextArea1);

        jButton1.setText("Назад");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setText("Повернутися до вибору");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(12, 12, 12)
                .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 100, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jButton2)
                .addContainerGap())
            .addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT_SIZE, 700, Short.MAX_VALUE)
        );
```

```
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT_SIZE, 465, Short.MAX_VALUE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton2)
                    .addComponent(jButton1))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
         JOptionPane.showMessageDialog(this,
        "<html><body>" +
        "<p>Вітаємо! Ви успішно пройшли теоретичну частину ArrayList.</p>" +
        "<p>Тепер ви знаєте основні концепції та операції, пов'язані з цією важливою структурою даних в Java.</p>" +
        "<p>Готові перейти до практичних завдань або тестування?</p>" +
        "</body></html>",
        "Заключення теоретичної частини", JOptionPane.INFORMATION_MESSAGE);

        this.dispose();

        MainForm mainForm = new MainForm();
        mainForm.setVisible(true);
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        MainForm mainForm = new MainForm();
        mainForm.setVisible(true);
        this.setVisible(false);
    }

    public static void main(String args[]) {
```

```
            java.awt.EventQueue.invokeLater(new Runnable() {
                public void run() {
                    new TheoryForm().setVisible(true);
                }
            });
        }

        // Variables declaration - do not modify
        private javax.swing.JButton jButton1;
        private javax.swing.JButton jButton2;
        private javax.swing.JScrollPane jScrollPane2;
        private javax.swing.JTextArea jTextArea1;
        // End of variables declaration
}
```

3. PracticeForm

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;

public class PracticeForm extends javax.swing.JFrame {

    public PracticeForm() {
        initComponents();
        setTitle("Практична частина - ArrayList Trainer");
        setSize(800, 500);
        setLocationRelativeTo(null);

        JOptionPane.showMessageDialog(this,
        "<html>Тепер, коли ви ознайомились з теорією ArrayList, давайте перевіримо та закріпимо отримані знання на практиці.<br>"
        + "Ви виконуватимете завдання, які допоможуть вам краще зрозуміти, як працювати з ArrayList в реальних ситуаціях.</html>",
        "Вступ до практичної частини",
        JOptionPane.INFORMATION_MESSAGE);
    }

    @SuppressWarnings("unchecked")
```

```
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jLabel1 = new javax.swing.JLabel();
    jScrollPane1 = new javax.swing.JScrollPane();
    jTextArea1 = new javax.swing.JTextArea();
    jButton1 = new javax.swing.JButton();
    jButton2 = new javax.swing.JButton();
    jLabel2 = new javax.swing.JLabel();
    jButton3 = new javax.swing.JButton();
    jScrollPane2 = new javax.swing.JScrollPane();
    jTextArea2 = new javax.swing.JTextArea();
    jButton4 = new javax.swing.JButton();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

    jLabel1.setText("Створіть ArrayList імен names та додайте до нього три імені: 'John', 'Emily', 'Bob'");

    jTextArea1.setColumns(20);
    jTextArea1.setRows(5);
    jScrollPane1.setViewportView(jTextArea1);

    jButton1.setText("Далі");
    jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
        }
    });

    jButton2.setText("Повернутися до теоретичної частини");
    jButton2.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton2ActionPerformed(evt);
        }
    });

    jLabel2.setText("Використовуючи список names з попереднього завдання, видаліть 'Emily' та
```

```
виведіть оновлений список на консоль.");

        jButton3.setText("Перевірити");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jTextArea2.setColumns(20);
        jTextArea2.setRows(5);
        jScrollPane2.setViewportView(jTextArea2);

        jButton4.setText("Перевірити");
        jButton4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton4ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jScrollPane1)
            .addComponent(jScrollPane2, javax.swing.GroupLayout.Alignment.TRAILING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jButton3, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jButton2)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
```

```
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel1)
                    .addComponent(jLabel2))
                .addGap(0, 23, Short.MAX_VALUE))
            .addComponent(jButton4, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jButton3)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jButton4)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 28,
Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton2)
                    .addComponent(jButton1))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        PracticeForm1 practiceForm1 = new PracticeForm1();
```

```
        practiceForm1.setVisible(true);

        this.dispose();
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        String userCode = jTextArea1.getText();
        String correctAnswer = "ArrayList<String> names = new ArrayList<>();\n" +
                               "names.add(\"John\");\n" +
                               "names.add(\"Emily\");\n" +
                               "names.add(\"Bob\");\n" +
                               "System.out.println(names);";

        if (userCode.replaceAll("\\s+","").equals(correctAnswer.replaceAll("\\s+",""))) {
            // Якщо відповідь правильна
            JOptionPane.showMessageDialog(this, "Вітаємо! Ви правильно створили список і додали елементи. Так тримати!");
            // Перехід до наступної форми або завдання
        } else {
            // Якщо відповідь неправильна
            JOptionPane.showMessageDialog(this, "Перевірте синтаксис створення списку і додавання елементів. Вам потрібно використовувати метод add для кожного нового елемента. Приклад: names.add(\\\"John\\\");");
        }
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        String userCode = jTextArea2.getText();
        String correctAnswerForTask2 = "names.remove(\"Emily\");\n" +
                                       "System.out.println(names);";

        if (userCode.replaceAll("\\s+","").contains(correctAnswerForTask2.replaceAll("\\s+",""))) {
            JOptionPane.showMessageDialog(this, "Чудово! Ви вдало видалили елемент зі списку. Це важливий навик при роботі з динамічними колекціями.");
        } else {
            JOptionPane.showMessageDialog(this, "Переконайтеся, що ви правильно вказали об'єкт або індекс для видалення. Для видалення по імені об'єкта використовуйте names.remove(\\\"Emily\\\");.\nЯкщо це не працює, перевірте, чи доданий такий елемент у список.");
```

```
        }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        TheoryForm theoryForm = new TheoryForm();
        theoryForm.setVisible(true);
        this.setVisible(false);
    }

    public static void main(String args[]) {

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new PracticeForm().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextArea jTextArea2;
    // End of variables declaration
}
```

4. PracticeFrom1

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;

public class PracticeForm1 extends javax.swing.JFrame {
```

```
    public PracticeForm1() {
        initComponents();
        setTitle("Практична частина - ArrayList Trainer");
        setSize(800, 500);
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane2 = new javax.swing.JScrollPane();
        jTextArea2 = new javax.swing.JTextArea();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel1.setText("Отримайте та виведіть на консоль другий елемент зі списку names.");

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);

        jLabel2.setText("Створіть новий ArrayList newNames і скопіюйте в нього всі елементи зі списку
names. Виведіть новий список на консоль.");

        jTextArea2.setColumns(20);
        jTextArea2.setRows(5);
        jScrollPane2.setViewportView(jTextArea2);

        jButton1.setText("Перевірити");
```

```
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

jButton2.setText("Далі");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton2ActionPerformed(evt);
    }
});

jButton3.setText("Назад");
jButton3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton3ActionPerformed(evt);
    }
});

jButton4.setText("Перевірити");
jButton4.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton4ActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING)
    .addComponent(jScrollPane2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
        .addContainerGap()
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
            .addComponent(jButton4, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                .addComponent(jButton1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
                    .addComponent(jButton3)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jButton2))
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                        .addComponent(jLabel1, javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel2, javax.swing.GroupLayout.Alignment.LEADING))
                    .addGap(0, 1, Short.MAX_VALUE)))
            .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jButton1)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jButton4)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 28,
Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton3)
                    .addComponent(jButton2))
                .addContainerGap())
        );
```

```
        pack();
    }// </editor-fold>

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        PracticeForm2 practiceForm2 = new PracticeForm2();
        practiceForm2.setVisible(true);

        this.dispose();
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        String userCode = jTextArea2.getText();
        // Приклад правильної відповіді для завдання 4
        String correctAnswerForTask4 = "ArrayList<String> newNames = new ArrayList<>(names);";
        String correctPrintStatement = "System.out.println(\"Новий список: \" + newNames);";

        if (userCode.replaceAll("\\s+", "").contains(correctAnswerForTask4.replaceAll("\\s+", "")) &&
userCode.replaceAll("\\s+", "").contains(correctPrintStatement.replaceAll("\\s+", ""))) {
            JOptionPane.showMessageDialog(this, "Ви успішно скопіювали список! Копіювання списків є
стандартною операцією при роботі з колекціями, коли необхідно зберегти оригінальні дані без змін.");
        } else {
            JOptionPane.showMessageDialog(this, "Переконайтеся, що ви використовуєте конструктор
ArrayList для створення копії списку. Він має виглядати так: new ArrayList<>(names);. \nЦе дозволить вам
створити новий список, який містить всі елементи оригінального списку names.");
        }
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        PracticeForm practiceForm = new PracticeForm();
        practiceForm.setVisible(true);
        this.setVisible(false);
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String userCode = jTextArea1.getText();
        // Приклад правильної відповіді для завдання 3
        String correctAnswerForTask3 = "System.out.println(names.get(1));";
```

```
        if (userCode.replaceAll("\\s+", "").contains(correctAnswerForTask3.replaceAll("\\s+", ""))) {
            JOptionPane.showMessageDialog(this, "Ви вірно отримали другий елемент зі списку. Використання методу get() є ключовим для доступу до елементів ArrayList.");
        } else {
            JOptionPane.showMessageDialog(this, "Переконайтеся, що ви використовуєте правильний індекс для доступу до елемента. Індексація у списку починається з нуля. \nМетод get повинен бути використаний з індексом елемента, який ви хочете отримати, наприклад, names.get(1) для другого елемента.");
        }
    }

    public static void main(String args[]) {

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new PracticeForm1().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextArea jTextArea2;
    // End of variables declaration
}
```

5. PracticeForm2

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;
```

```
public class PracticeForm2 extends javax.swing.JFrame {

    public PracticeForm2() {
        initComponents();
        setTitle("Практична частина - ArrayList Trainer");
        setSize(800, 500);
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jButton3 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane2 = new javax.swing.JScrollPane();
        jTextArea2 = new javax.swing.JTextArea();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jButton3.setText("Назад");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jButton4.setText("Перевірити");
        jButton4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton4ActionPerformed(evt);
            }
```

```
});

jLabel1.setText("Перевірте, чи містить список names ім'я 'John'. Якщо так, виведіть 'John присутній.', якщо ні 'John відсутній.'");

jTextArea1.setColumns(20);
jTextArea1.setRows(5);
jScrollPane1.setViewportView(jTextArea1);

jLabel2.setText("Очистіть весь список names та виведіть список після очищення.");

jTextArea2.setColumns(20);
jTextArea2.setRows(5);
jScrollPane2.setViewportView(jTextArea2);

jButton1.setText("Перевірити");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

jButton2.setText("Перейти до тестової частини");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton2ActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING)
    .addComponent(jScrollPane2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
        .addContainerGap()
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
```

```
                .addComponent(jButton4, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jButton1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
                    .addComponent(jButton3)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jButton2))
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                        .addComponent(jLabel1, javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel2, javax.swing.GroupLayout.Alignment.LEADING))
                    .addGap(0, 0, Short.MAX_VALUE)))
            .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jButton1)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jButton4)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 28,
Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton3)
                    .addComponent(jButton2))
```

```
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        PracticeForm1 practiceForm1 = new PracticeForm1();
        practiceForm1.setVisible(true);
        this.setVisible(false);
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        String userCode = jTextArea2.getText().replaceAll("\\s+", ""); // отримання тексту з текстового поля та видалення всіх пробільних символів для спрощення порівняння
        String correctMethodCall = "names.clear()";
        String correctPrintStatement = "System.out.println(\"Список після очищення: \" + names)";

        // Перевірка, чи введений код містить правильні виклики методів
        if (userCode.contains(correctMethodCall.replaceAll("\\s+", "")) &&
            userCode.contains(correctPrintStatement.replaceAll("\\s+", ""))) {
            JOptionPane.showMessageDialog(this, "Вітаємо! Ви успішно очистили список!", "Результат", JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(this, "Переконайтесь, що ви використовуєте метод clear() для очищення списку. Спробуйте ще раз.", "Помилка", JOptionPane.ERROR_MESSAGE);
        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String userCode = jTextArea1.getText();  // Припускаємо, що jTextArea1 це поле, де користувач вводить свій код

        // Приклад правильних відповідей для завдання 5
        String checkPresenceCode = "names.contains(\"John\")";
        String correctResponseCode = "System.out.println(\"John присутній у списку.\");";
        String incorrectResponseCode = "System.out.println(\"John відсутній у списку.\");";
```

```
        // Перевірка коду користувача
        if (userCode.replaceAll("\\s+", "").contains(checkPresenceCode.replaceAll("\\s+", ""))) {
            if (userCode.replaceAll("\\s+", "").contains(correctResponseCode.replaceAll("\\s+", "")) && userCode.replaceAll("\\s+", "").contains(incorrectResponseCode.replaceAll("\\s+", ""))) {
                JOptionPane.showMessageDialog(this, "Ви правильно перевірили наявність елемента у списку та обробили можливі варіанти відповідей!");
            } else {
                JOptionPane.showMessageDialog(this, "Переконайтесь, що ви правильно обробляєте відповідь. Використовуйте умовний оператор для виведення правильного повідомлення залежно від результату перевірки.");
            }
        } else {
            JOptionPane.showMessageDialog(this, "Ваш код не перевіряє наявність 'John' у списку names. Використовуйте метод contains() для перевірки наявності елемента.");
        }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        String message = "<html>Вітаємо! Ви успішно пройшли практичну частину з ArrayList.<br>"
        + "Ми сподіваємося, що ці вправи допомогли вам краще зрозуміти, як застосовувати ArrayList в програмуванні на Java.<br>"
        + "Тепер ви можете перейти до секції тестування, щоб перевірити свої знання.</html>";

        // Відображення діалогового вікна з заключенням практичної частини
        JOptionPane.showMessageDialog(this, message);

        // Перехід до тестової форми після ніатискання ОК у діалоговому вікні
        TestForm testForm = new TestForm();
        testForm.setVisible(true);

        // Закриття поточної форми
        this.dispose();
    }

    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
```

```
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(PracticeForm2.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(PracticeForm2.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(PracticeForm2.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(PracticeForm2.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new PracticeForm2().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
```

```
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextArea jTextArea2;
    // End of variables declaration
}
```

6. TestForm

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;

public class TestForm extends javax.swing.JFrame {

    public TestForm() {
        initComponents();
        setTitle("Тестова частина - ArrayList Trainer");
        setLocationRelativeTo(null);
        JOptionPane.showMessageDialog(this,
            "<html>Тепер, коли ви пройшли через теоретичну та практичну частини нашого тренажера ArrayList,<br>"
            + "настав час перевірити ваші знання. Ця частина містить серію тестових питань,<br>"
            + "які допоможуть вам оцінити розуміння ArrayList. Готові почати?</html>",
            "Вступ до тестової частини",
            JOptionPane.INFORMATION_MESSAGE);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        buttonGroup2 = new javax.swing.ButtonGroup();
```

```
buttonGroup3 = new javax.swing.ButtonGroup();
jRadioButton1 = new javax.swing.JRadioButton();
jRadioButton2 = new javax.swing.JRadioButton();
jRadioButton3 = new javax.swing.JRadioButton();
jRadioButton4 = new javax.swing.JRadioButton();
jRadioButton5 = new javax.swing.JRadioButton();
jRadioButton6 = new javax.swing.JRadioButton();
jRadioButton7 = new javax.swing.JRadioButton();
jRadioButton8 = new javax.swing.JRadioButton();
jRadioButton9 = new javax.swing.JRadioButton();
jRadioButton10 = new javax.swing.JRadioButton();
jRadioButton11 = new javax.swing.JRadioButton();
jRadioButton12 = new javax.swing.JRadioButton();
jLabel1 = new javax.swing.JLabel();
jLabel2 = new javax.swing.JLabel();
jLabel3 = new javax.swing.JLabel();
jButton1 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

buttonGroup1.add(jRadioButton1);
jRadioButton1.setText("A) ArrayList strings = new ArrayList();");

buttonGroup1.add(jRadioButton2);
jRadioButton2.setText("B) ArrayList<String> strings = new ArrayList<String>();");

buttonGroup1.add(jRadioButton3);
jRadioButton3.setText("C) List<String> strings = new ArrayList<>();");

buttonGroup1.add(jRadioButton4);
jRadioButton4.setText("D) ArrayList strings = new ArrayList<String>();");

buttonGroup2.add(jRadioButton5);
jRadioButton5.setText("C) add()");

buttonGroup2.add(jRadioButton6);
jRadioButton6.setText("D) push()");
jRadioButton6.addActionListener(new java.awt.event.ActionListener() {
```

```
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jRadioButton6ActionPerformed(evt);
        }
    });

    buttonGroup2.add(jRadioButton7);
    jRadioButton7.setText("A) append()");

    buttonGroup2.add(jRadioButton8);
    jRadioButton8.setText("B) insert()");

    buttonGroup3.add(jRadioButton9);
    jRadioButton9.setText("C) at(index)");

    buttonGroup3.add(jRadioButton10);
    jRadioButton10.setText("D) indexOf(index)");

    buttonGroup3.add(jRadioButton11);
    jRadioButton11.setText("A) elementAt(index)");

    buttonGroup3.add(jRadioButton12);
    jRadioButton12.setText("B) get(index)");

    jLabel1.setText("Який із наступних способів правильно створює ArrayList, що може зберігати об'єкти типу String?");

    jLabel2.setText("Який метод використовується для додавання елемента в кінець ArrayList?");

    jLabel3.setText("Як отримати доступ до елемента в ArrayList за індексом?");

    jButton1.setText("Перевірити відповіді");
    jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
```

```
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jRadioButton10)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jButton1))
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel2)
                            .addComponent(jRadioButton6)
                            .addComponent(jRadioButton7)
                            .addComponent(jRadioButton5)
                            .addComponent(jRadioButton8)
                            .addComponent(jRadioButton3)
                            .addComponent(jRadioButton2)
                            .addComponent(jRadioButton1)
                            .addComponent(jLabel1)
                            .addComponent(jRadioButton4)
                            .addComponent(jRadioButton11)
                            .addComponent(jRadioButton9)
                            .addComponent(jRadioButton12)
                            .addComponent(jLabel3))
                        .addGap(0, 59, Short.MAX_VALUE)))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
            .addComponent(jRadioButton2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton4)
            .addGap(18, 18, 18)
            .addComponent(jLabel2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton7)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton8)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton5)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton6)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(jLabel3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton11)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton12)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton9)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jRadioButton10)
                .addComponent(jButton1))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

    pack();
}// </editor-fold>

private void jRadioButton6ActionPerformed(java.awt.event.ActionEvent evt) {
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    int correctAnswersCount = 0;
```

```
    if (jRadioButton3.isSelected()) correctAnswersCount++;
    if (jRadioButton5.isSelected()) correctAnswersCount++;
    if (jRadioButton12.isSelected()) correctAnswersCount++;

    JOptionPane.showMessageDialog(this,
        "Ви правильно відповіли на " + correctAnswersCount + " з 3 питань.",
        "Результати тесту",
        JOptionPane.INFORMATION_MESSAGE);

    // Переходимо до наступної форми для показу додаткових питань або завершення
    TestForm1 testForm1 = new TestForm1();
    testForm1.setVisible(true);
    this.setVisible(false);  // Закриваємо поточне вікно тестування
}


public static void main(String args[]) {

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new TestForm().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.ButtonGroup buttonGroup1;
private javax.swing.ButtonGroup buttonGroup2;
private javax.swing.ButtonGroup buttonGroup3;
private javax.swing.JButton jButton1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JRadioButton jRadioButton1;
private javax.swing.JRadioButton jRadioButton10;
private javax.swing.JRadioButton jRadioButton11;
private javax.swing.JRadioButton jRadioButton12;
```

```
    private javax.swing.JRadioButton jRadioButton2;
    private javax.swing.JRadioButton jRadioButton3;
    private javax.swing.JRadioButton jRadioButton4;
    private javax.swing.JRadioButton jRadioButton5;
    private javax.swing.JRadioButton jRadioButton6;
    private javax.swing.JRadioButton jRadioButton7;
    private javax.swing.JRadioButton jRadioButton8;
    private javax.swing.JRadioButton jRadioButton9;
    // End of variables declaration
}
```

7. TestForm1

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;

public class TestForm1 extends javax.swing.JFrame {

    public TestForm1() {
        initComponents();
        setTitle("Тестова частина - ArrayList Trainer");
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        buttonGroup2 = new javax.swing.ButtonGroup();
        buttonGroup3 = new javax.swing.ButtonGroup();
        jRadioButton10 = new javax.swing.JRadioButton();
        jRadioButton11 = new javax.swing.JRadioButton();
        jRadioButton1 = new javax.swing.JRadioButton();
        jRadioButton12 = new javax.swing.JRadioButton();
        jRadioButton2 = new javax.swing.JRadioButton();
        jLabel1 = new javax.swing.JLabel();
        jRadioButton3 = new javax.swing.JRadioButton();
        jLabel2 = new javax.swing.JLabel();
```

```
jRadioButton4 = new javax.swing.JRadioButton();
jLabel3 = new javax.swing.JLabel();
jRadioButton5 = new javax.swing.JRadioButton();
jButton1 = new javax.swing.JButton();
jRadioButton6 = new javax.swing.JRadioButton();
jRadioButton7 = new javax.swing.JRadioButton();
jRadioButton8 = new javax.swing.JRadioButton();
jRadioButton9 = new javax.swing.JRadioButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

buttonGroup3.add(jRadioButton10);
jRadioButton10.setText("D) list.change(index, newValue);");

buttonGroup3.add(jRadioButton11);
jRadioButton11.setText("A) list.set(index, newValue);");

buttonGroup1.add(jRadioButton1);
jRadioButton1.setText("A) list.removeElement(index);");

buttonGroup3.add(jRadioButton12);
jRadioButton12.setText("B) list.update(index, newValue);");

buttonGroup1.add(jRadioButton2);
jRadioButton2.setText("B) list.delete(index);");

jLabel1.setText("Як правильно видалити елемент з ArrayList по індексу?");

buttonGroup1.add(jRadioButton3);
jRadioButton3.setText("C) list.remove(index);");

jLabel2.setText("Який метод використовується для перевірки, чи містить ArrayList заданий об'єкт?");

buttonGroup1.add(jRadioButton4);
jRadioButton4.setText("D) list.erase(index);");

jLabel3.setText("Як змінити елемент у ArrayList за вказаним індексом?");
```

```
buttonGroup2.add(jRadioButton5);
jRadioButton5.setText("C) list.includes(object);");

jButton1.setText("Перевірити відповіді");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

buttonGroup2.add(jRadioButton6);
jRadioButton6.setText("D) list.exists(object);");
jRadioButton6.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jRadioButton6ActionPerformed(evt);
    }
});

buttonGroup2.add(jRadioButton7);
jRadioButton7.setText("A) list.contains(object);");

buttonGroup2.add(jRadioButton8);
jRadioButton8.setText("B) list.has(object);");

buttonGroup3.add(jRadioButton9);
jRadioButton9.setText("C) list.replace(index, newValue);");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jRadioButton10)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jButton1))
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel2)
                            .addComponent(jRadioButton6)
                            .addComponent(jRadioButton7)
                            .addComponent(jRadioButton5)
                            .addComponent(jRadioButton8)
                            .addComponent(jRadioButton3)
                            .addComponent(jRadioButton2)
                            .addComponent(jRadioButton1)
                            .addComponent(jLabel1)
                            .addComponent(jRadioButton4)
                            .addComponent(jRadioButton11)
                            .addComponent(jRadioButton9)
                            .addComponent(jRadioButton12)
                            .addComponent(jLabel3))
                        .addGap(0, 144, Short.MAX_VALUE)))
                .addContainerGap())
            );
            layout.setVerticalGroup(
                layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(jLabel1)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jRadioButton1)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jRadioButton2)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jRadioButton3)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jRadioButton4)
                    .addGap(18, 18, 18)
                    .addComponent(jLabel2)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jRadioButton7)
```

```
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton8)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton5)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton6)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jLabel3)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton11)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton12)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jRadioButton9)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jRadioButton10)
                    .addComponent(jButton1))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            );

            pack();
        }// </editor-fold>

        private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
            int correctAnswersCount = 0;
            String resultsMessage;

            if (jRadioButton3.isSelected()) correctAnswersCount++;
            if (jRadioButton7.isSelected()) correctAnswersCount++;
            if (jRadioButton11.isSelected()) correctAnswersCount++;

            resultsMessage = "Ви правильно відповіли на " + correctAnswersCount + " з 3 питань.";

            JOptionPane.showMessageDialog(this, resultsMessage, "Результати тесту",
JOptionPane.INFORMATION_MESSAGE);

            TestForm2 testForm2 = new TestForm2();
```

```
    testForm2.setVisible(true);
    this.dispose();
}

private void jRadioButton6ActionPerformed(java.awt.event.ActionEvent evt) {

}

public static void main(String args[]) {

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new TestForm1().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.ButtonGroup buttonGroup1;
private javax.swing.ButtonGroup buttonGroup2;
private javax.swing.ButtonGroup buttonGroup3;
private javax.swing.JButton jButton1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JRadioButton jRadioButton1;
private javax.swing.JRadioButton jRadioButton10;
private javax.swing.JRadioButton jRadioButton11;
private javax.swing.JRadioButton jRadioButton12;
private javax.swing.JRadioButton jRadioButton2;
private javax.swing.JRadioButton jRadioButton3;
private javax.swing.JRadioButton jRadioButton4;
private javax.swing.JRadioButton jRadioButton5;
private javax.swing.JRadioButton jRadioButton6;
private javax.swing.JRadioButton jRadioButton7;
private javax.swing.JRadioButton jRadioButton8;
private javax.swing.JRadioButton jRadioButton9;
// End of variables declaration
```

```
}
```

8. TestForm2:

```
package com.kolobov.arraylist;

import javax.swing.JOptionPane;

public class TestForm2 extends javax.swing.JFrame {

    public TestForm2() {
        initComponents();
        setTitle("Тестова частина - ArrayList Trainer");
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jRadioButton10 = new javax.swing.JRadioButton();
        jButton1 = new javax.swing.JButton();
        jRadioButton11 = new javax.swing.JRadioButton();
        jRadioButton6 = new javax.swing.JRadioButton();
        jRadioButton1 = new javax.swing.JRadioButton();
        jRadioButton7 = new javax.swing.JRadioButton();
        jRadioButton12 = new javax.swing.JRadioButton();
        jRadioButton8 = new javax.swing.JRadioButton();
        jRadioButton2 = new javax.swing.JRadioButton();
        jRadioButton9 = new javax.swing.JRadioButton();
        jLabel1 = new javax.swing.JLabel();
        jRadioButton3 = new javax.swing.JRadioButton();
        jLabel2 = new javax.swing.JLabel();
        jRadioButton4 = new javax.swing.JRadioButton();
        jLabel3 = new javax.swing.JLabel();
        jRadioButton5 = new javax.swing.JRadioButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jRadioButton10.setText("D) list.size() == 0;");
```

```
jButton1.setText("Перевіриmи відповіді");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

jRadioButton11.setText("A) list.isEmpty();");

jRadioButton6.setText("D) Queue");
jRadioButton6.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jRadioButton6ActionPerformed(evt);
    }
});

jRadioButton1.setText("A) list.clearItems();");

jRadioButton7.setText("A) Set");

jRadioButton12.setText("B) list.isClear();");

jRadioButton8.setText("B) Map");

jRadioButton2.setText("B) list.removeAll();");

jRadioButton9.setText("C) list.hasItems();");

jLabel1.setText("Як очистити ArrayList від усіх елементів?");

jRadioButton3.setText("C) list.deleteAll();");

jLabel2.setText("Якій інтерфейс колекцій є основою для ArrayList?");

jRadioButton4.setText("D) list.clear();");

jLabel3.setText("Як перевірити, чи порожній ArrayList?");
```

```
        jRadioButton5.setText("C) List");

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jRadioButton10)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jButton1))
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel2)
                            .addComponent(jRadioButton6)
                            .addComponent(jRadioButton7)
                            .addComponent(jRadioButton5)
                            .addComponent(jRadioButton8)
                            .addComponent(jRadioButton3)
                            .addComponent(jRadioButton2)
                            .addComponent(jRadioButton1)
                            .addComponent(jLabel1)
                            .addComponent(jRadioButton4)
                            .addComponent(jRadioButton11)
                            .addComponent(jRadioButton9)
                            .addComponent(jRadioButton12)
                            .addComponent(jLabel3))
                        .addGap(0, 319, Short.MAX_VALUE)))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
```

```
            .addComponent(jLabel1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton4)
            .addGap(18, 18, 18)
            .addComponent(jLabel2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton7)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton8)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton5)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton6)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(jLabel3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton11)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton12)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton9)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jRadioButton10)
                .addComponent(jButton1))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        int correctAnswersCount = 0;

        if (jRadioButton4.isSelected()) correctAnswersCount++;
        if (jRadioButton7.isSelected()) correctAnswersCount++;
        if (jRadioButton11.isSelected()) correctAnswersCount++;

        JOptionPane.showMessageDialog(this,
            "Ви правильно відповіли на " + correctAnswersCount + " з 3 питань.",
            "Результати тесту",
            JOptionPane.INFORMATION_MESSAGE);

        JOptionPane.showMessageDialog(this,
            "Вітаємо! Ви успішно завершили тестову частину ArrayList.\nВи можете повернутися до головної сторінки.",
            "Заключення тестової частини",
            JOptionPane.INFORMATION_MESSAGE);
        this.dispose();

        MainForm mainForm = new MainForm();
        mainForm.setVisible(true);
    }

    private void jRadioButton6ActionPerformed(java.awt.event.ActionEvent evt) {

    }

    public static void main(String args[]) {

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new TestForm2().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
```

```
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JRadioButton jRadioButton1;
    private javax.swing.JRadioButton jRadioButton10;
    private javax.swing.JRadioButton jRadioButton11;
    private javax.swing.JRadioButton jRadioButton12;
    private javax.swing.JRadioButton jRadioButton2;
    private javax.swing.JRadioButton jRadioButton3;
    private javax.swing.JRadioButton jRadioButton4;
    private javax.swing.JRadioButton jRadioButton5;
    private javax.swing.JRadioButton jRadioButton6;
    private javax.swing.JRadioButton jRadioButton7;
    private javax.swing.JRadioButton jRadioButton8;
    private javax.swing.JRadioButton jRadioButton9;
    // End of variables declaration
}
```