**Міністерство освіти і науки України**

**Луцький національний технічний університет**
(повне найменування закладу вищої освіти)

**Факультет комп'ютерних та інформаційних технологій**
(повне найменування факультету)

**Кафедра комп'ютерної інженерії та кібербезпеки**
(повне найменування кафедри)

# КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

# ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ ТРЕНУВАННЯ УВАГИ ТА ПЕРЕВІРКИ ЗНАНЬ ЗАСОБАМИ C# WINFORMS

# A SOFTWARE PACKAGE FOR TRAINING ATTENTION AND CHECKING KNOWLEDGE USING C# WINFORMS

спеціальність 123 Комп'ютерна інженерія
(шифр і назва спеціальності)

освітня програма Комп'ютерна інженерія
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи КІ-41
Климець Владислав Сергійович

(підпис)

Керівник:
к.т.н., доцент
Пех Петро Антонович

(підпис)

Кваліфікаційну роботу
допущено до захисту
« 07 » червня 2024 р.
Гарант освітньої програми:
к.т.н., доцент
Лавренчук Світлана Василівна

(підпис)

Луцьк – 2024 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *комп'ютерної інженерії та кібербезпеки*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *123 Комп'ютерна інженерія*
Освітня програма: *«Комп'ютерна інженерія»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ проф. Н.Чернящук
« 10 » 01 2024 р.

# ЗАВДАННЯ
# НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

*Климцю Владиславу Сергійовичу*
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *Програмний комплекс для тренування уваги та перевірки знань засобами C# WinForms*

Керівник роботи *к.т.н., доцент Пех Петро Антонович*

затверджені наказом закладу вищої освіти від «30» грудня 2023 року № 459/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи 11.06.2024р.

3. Вихідні дані до роботи *Джерелом розробки є науково-технічна література та публікації в періодичних виданнях з даного питання, опубліковані зарубіжні та вітчизняні роботи в даній області та різні інтернет-ресурси технічного спрямування*

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
*Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз і вибір засобів проектування, опис функціонального наповнення об'єкта проектування, розробка й обгрунтування системного наповнення, оцінка ергономічних та надійнісних параметрів проектованої Системи, функціонально-структурна схема роботи об'єкта проектування.*

5. Перелік графічного (ілюстративного) матеріалу:

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| *Аналіз проблеми за темою роботи та постановка завдань дослідження* | *Пех П.А. ,доцент* | | |
| *Теоретичне дослідження та практична реалізація* | *Пех П.А. ,доцент* | | |
| *Експериментальне дослідження системи* | *Пех П.А. ,доцент* | | |
| *Нормоконтроль* | *Багнюк Н.В., доцент* | | |
| *Гарант ОП* | *Лавренчук С.В., доцент* | | |
| *Показник запозичень тексту* | ***___%*** | | |
| *Академічна доброчесність* | *Міскевич О.І.,асистент* | | |

7. Дата видачі завдання 10.01.2024 р.

КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1. | *Провести огляд літератури по темі кваліфікаційної роботи* | до 15.02.2024 р. | Виконано |
| 2. | *Провести аналіз загальної проблеми і вибір напрямків дослідження* | до 15.03.2024 р. | Виконано |
| 3. | *Розробити функціональну схему роботи програмного продукту* | до 04.05.2024 р. | Виконано |
| 4. | *Формування висновків* | до 07.05.2025 р. | Виконано |
| 5. | *Формування списку використаних джерел* | до 10.05.2024 р. | Виконано |
| 6. | *Формування додатків* | до 15.05.2024 р. | Виконано |
| 7. | *Оформлення ілюстративного матеріалу* | до 20.05.2024 р. | Виконано |
| 8. | *Нормоконтроль* | до 01.06.2024 р. | Виконано |
| 9. | *Інструментальна перевірка на академічний плагіат* | до 04.06.2024 р | Виконано |
| 10. | *Представлення кваліфікаційної роботи бакалавра до захисту* | до 11.06.2024 р. | Виконано |

**Здобувач вищої освіти** _____ Климовець В.С.
(підпис) (прізвище, ініціали)

**Керівник кваліфікаційної роботи** _____ Пех П.А.
(підпис) (прізвище, ініціали)

# АНОТАЦІЯ

Климець В.С. Програмний комплекс для тренування уваги та перевірки знань засобами C# WinForms.

Кваліфікаційна робота бакалавра ОП «Комп'ютерна інженерія» спеціальності 123 Комп'ютерна інженерія. Луцький національний технічний університет. Луцьк, 2024.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків та списку використаних джерел та додатків.

У роботі виконано огляд програмних засобів для розробки ігор та тестів для тренування уваги та перевірки знань. Дано детальну характеристику інструментів для розробки програмного комплексу засобами C# WinForms. Розроблено проект у складі наступних програм: програми-гри «Знайдіть пару» на сітці розмірами 4*4; програми-гри «Знайдіть пару» на сітці розмірами 4*5; програми-гри «Математичний тест» для перевірки правильності виконання арифметичних операцій; програми-тесту на базі компонента RadioButton для вибору однієї правильної відповіді з кількох; програми-тесту на базі компонента CheckBox для вибору більше однієї правильної відповіді з кількох; програми-тесту на базі компонента ListBox для встановлення відповідності між двома множинами тверджень та програми-тесту на базі компонента ComboBox для вибору правильних відповідей екзаменаційного білета.

Ключові слова: мова програмування C#, середовище C#Winforms, програма-гра, програма-тест, тренування уваги, перевірка знань

# ANNOTATION

V.S. Klymets A software complex for training attention and testing knowledge using C# WinForms.

Bachelor's qualifying thesis of the EP «Computer Engineering» specialty 123 Computer Engineering. Lutsk National Technical University. Lutsk, 2024.

The bachelor's qualification work consists of an introduction, three sections, conclusions, a list of used sources and appendices.

The paper reviews software tools for developing games and tests for training attention and testing knowledge. A detailed description of the tools for developing a software complex using C# WinForms tools is given. The developed project is consists of the following programs: «Find a pair» game-program on a 4*4 grid; game-program «Find a pair» on a 4*5 grid; game-program «Mathematical test» to check the correctness of arithmetic operations; test-program based on the RadioButton component for choosing one correct answer from several; test-program based on the CheckBox component for selecting more than one correct answer from several; test-program based on the ListBox component for establishing correspondence between two sets of statements and test-program based on the ComboBox component for selecting the correct answers of the exam ticket.

Keywords: C# programming language, C# winforms environment, program-game, program-test, attention training, knowledge test

# ЗМІСТ

# ВСТУП

Актуальність теми. У час стрімкого проникнення комп'ютерно-інформаційних технологій в усі сфери діяльності людини зростає роль програм ігрового та тестувального характеру. Серед таких програм чільне місце займають програми для тренування уваги та перевірки знань.

Тренування уваги за допомогою ігрових програм забезпечує:

– виконання будь-якої роботи у коротші терміни і з вищою якістю;

– швидке реагування на ситуацію, яка динамічно змінюється, а концентрація на головному та вміння приймати правильні рішення є дуже важливими, наприклад, для керівника;

– тренування пам'яті; чим уважніше слухаєте або читаєте, тим більше можете запам'ятати.

Перевірка знань за допомогою програм-тестів також вкрай необхідний елемент сучасного навчального процесу. Саме тому запропонована тема щодо розробки програмного комплексу для тренування уваги та перевірки знань засобами C# WinForms є актуальною на теперішній час.

Мета дослідження полягає у розробці комлексу програм для тренування уваги та перевірки знань засобами C# WinForms.

Об'єкт дослідження – засоби та процес розробки програм ігрового характеру для тренування уваги та програм-тестів для перевірки знань.

Предмет дослідження – програми ігрового характеру для тренування уваги та програми-тести для перевірки знань. Дослідження передбачає вибір та детальне вивчення інструментів для вирішення завдань розробки програм ігрового характеру для тренування уваги та програм-тестів для перевірки знань та безпосереднє створення комплексу таких програм. Розроблений комплекс програм розглядається як результат виконання кваліфікаційної роботи.

Завдання, які необхідно виконати:

1. Виконати огляд програмних засобів для розробки ігор та тестів для тренування уваги та перевірки знань.

2. Дати детальну характеристику інструментів для розробки програмного комплексу засобами C# WinForms.

3. Розробити програму-гру «Знайдіть пару» різного ступеня складності на сітках розмірами 4*4 та 4*5.

4. Розробити програму-гру «Математичний тест» для перевірки правильності виконання арифметичних операцій.

5. Розробити програми-тести різного рівня складності на базі компонент RadioButton, CheckBox, ListBox та ComboBox.

Методи досліджень. Теоретичні основи дослідження базуються на сучасних технологіях програмування, зокрема, на технології об’єктно-орієнтованого програмування.

Наукова новизна одержаних результатів полягає у розробці програм-ігор та програм-тестів сучасними засобами середовища C# WinForms.

Практичне значення одержаних результатів. Розроблений програмний проект (комплекс програм) може бути використаний для тренування уваги та перевірки математичних знань дітей, а також для розроблення повноцінних тестів різного рівня складності для оцінювання знань здобувачів освіти.

Особистий внесок здобувача. Комплекс програм, запропонований у кваліфікаційній роботі, розроблений здобувачем самостійно.

Апробація результатів**.**

Публікації. P. Pekh, N. Khrystinets, R. Gubish, V. Shulgach. Simulation of two-stage temperature regulation system. // Computer-integrated technologies: education. science, production: Scientific journal. Lutsk: LNTU, 2023. Issue No. 53. Article 50-57. https://doi.org/10.36910/6775-2524-0560-2023-51-14.

# РОЗДІЛ 1
# ОГЛЯД ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РОЗРОБКИ ІГОР ТА ТЕСТІВ ДЛЯ ТРЕНУВАННЯ УВАГИ ТА ПЕРЕВІРКИ ЗНАНЬ

## 1.1 Середовища розробки програм та мови програмування

Розробники програмних продуктів перш за все вирішують важливу задачу вибору середовища та мови програмування [1]. Тому варто з'ясувати, які саме мови програмування використовуються у теперішній час для розроблення програм взагалі та ігор і тестів зокрема.

«Мова програмування – це система позначень для опису алгоритмів і структур даних, певна штучна формальна система, засобами якої можна виражати алгоритми. Мову програмування визначає набір лексичних, синтаксичних і семантичних правил, що задають зовнішній вигляд програми та дії, які виконує виконавець (комп'ютер) під її управлінням» [2].

Наведемо найпопулярніші на сьогодні мови програмування і їх спрямованість:

– Python – це мова високого рівня загального призначення. Python почав користуватися популярністю протягом останніх років. Це сталося тому, що університети США та деяких інших західних країн почали викладати цю мову, яка відрізняється від інших мов програмування своєю простотою.

– Java – це також мова високого рівня загального призначення. Вона використовується для розробки і backend, і front-end сайтів, як і в багатьох інших різнопланових сферах, але є достатньо складною.

– C/C++ – це теж мова високого рівня загального призначення. Вивчення цієї мови дає людині базові знання з програмування. Вивчається у більшості технічних університетів світу.

– C# – це мова, яка є наступною сходинкою після мов C/C++. Варто також зазначити, що ця мова прогамування динамічно розвивається, що забезпечує їй хороші перспективи.

– JavaScript (JS) – це мова для розробки веб-сайтів, мобільних програм,

серверів та багато іншого. Знати JS потрібно і фронтенд- розробникам, і бекенд-розробникам, і фулстек-спеціалістам. Мова JavaScript була створена для браузера завдяки технології, яка називається WebAssembly, що дозволяє трансформувати код і виконувати його в браузері. Відповідно, вона працює на всіх комп'ютерах та смартфонах.

– PHP – це мова програмування, за допомогою якої створюються веб – сайти. Цією мовою написана також і програма WordPress, яка сама є системою управління контентом сайту (CMS), тобто самостійною комплексною програмою, за допомогою якої розробляються сайти. Зокрема, і сайт Facebook. То ж не дивно, що ця мова була і є досить затребуваною.

– SQL/MySQL – це мова для роботи з базами даних. Глибоке знання лише однієї цієї мови достатнє для працевлаштування у сфері IT.

– TypeScript – це мова, яка є удосконаленням та конкурентом мови JavaScript.

– Elixir – це функціональна мова, як і подібні їй мови F++, Haskel. Функціональні мови не дуже поширені серед програмістів.

– Kotlin – це мова, що намагається перехопити нішу, яку займає мова програмування Java. Однак остання суттєво не здає своїх позицій.

Про популярність мов можна судити за їх рейтингами, складеними шляхом опитування розробників програмного забезпечення або шляхом аналізу ринку програм. Один з таких рейтингів [3] наведено на рисунку 1.1.

Як бачимо з цього рейтингу, мова C# займає достатньо високе місце серед інших мов програмування.

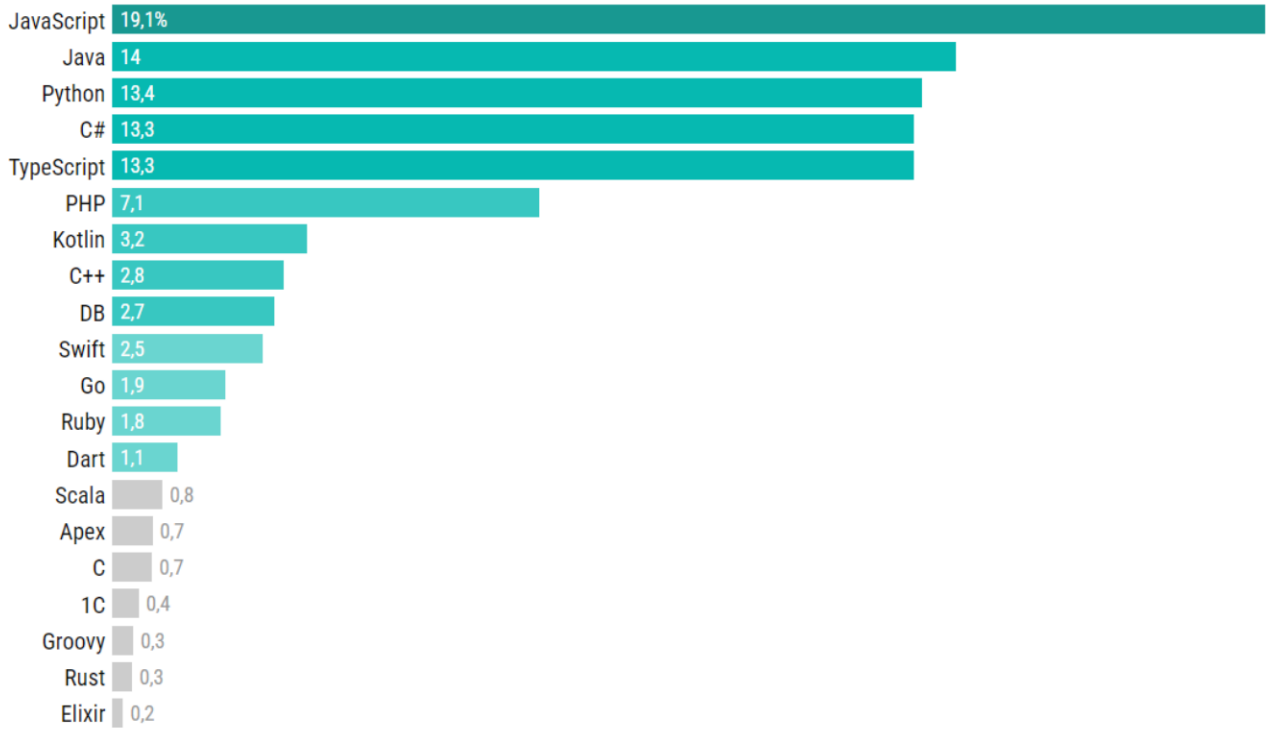


Рисунок 1.1 – Рейтинг мов програмування серед українських розробників

## 1.2 Огляд середовищ програмування

«Візуальне середовище програмування – інтегроване середовище розробки програмних засобів (IDE – integrated development environment), яке містить редактор вихідного коду, компілятор чи/або інтерпретатор, засоби автоматизації збірки та засоби спрощення розробки графічного інтерфейсу користувача. Середовища для візуального програмування також надають змогу конструювати програми шляхом оперування графічними об’єктами. Багато сучасних візуальних середовищ програмування використовуються для реалізації принципів об'єктно-орієнтованого підходу у розробці програмного забезпечення» [4].

Далі наводимо відомі середовища програмування за спаданням їх популярності:

– Visual Studio – це інтегроване безкоштовне середовище розробки, яке підтримує багато мов програмування, включаючи C#, JavaScript, F++, TypeScript, Python та багато інших.

– IntelliJ IDEA – це також популярне середовище розробки, особливо для Java-проектів. Воно підтримує такі мови програмування, як Java, Kotlin, Groovy, Scala та багато інших.

– Eclipse є популярним середовищем розробки для Java-проектів. Воно підтримує такі мови програмування, як C++, Python, PHP та інші.

– PyCharm є середовищем розробки для Python-проектів. підтримує Django та Flask, що дозволяє розробникам Python створювати веб-додатки та інші проекти.

– Atom підтримує багато мов програмування, включаючи JavaScript, HTML, CSS, Python та інші.

– Sublime Text підтримує Python, HTML, CSS та інші мови програмування.

– Vim є популярним серед програмістів Linux та Unix. Він має потужну систему редагування тексту та може підтримувати багато мов програмування.

– NetBeans є інтегрованим середовищем розробки для Java, підтримує також C/C++, HTML, PHP та інші мови програмування.

– Xcode є інтегрованим середовищем розробки для створення додатків для macOS та iOS.

– Code: Blocks є відкритим середовищем розробки для C++ та інших мов програмування.

– CodeLite підтримує мови програмуавння C++, PHP та інші.

– Brackets підтримує такі мови програмування, як HTML, CSS, JavaScript та інші.

У наведеному списку представлені ряд середовищ розробки програм, які використовуються сьогодні програмістами з усього світу. Всі вони мають свої переваги та недоліки. Вибір середовища розробки програм залежить від багатьох факторів. Серед них – вибрана мова програмування, категорія та складність проекту, уподобання розробника програмного продукту.

### 1.3 Вибір мови та середовища програмування

Вибір мови та середовища програмування в кінцевому випадку залишається за розробником програми. Для розробки комплексу програм кваліфікаційної роботи ми вибрали середовище Visual Studio та мову C#. Такий вибір пояснюється наступним.

Поняття мова C# та технології платформи .NET (Windows Forms, Xamarin, WPF, ASP.NET) тісно пов'язані. Тобто, коли ми говоримо .NET, то маємо на увазі C#, і навпаки. Однак ототожнювати ці два поняття все ж не варто. Мова C# була створена спеціально для роботи з фреймворком .NET, проте саме поняття .NET дещо ширше. Щодо платформи .NET зазначимо наступне.

«Основою платформи .NET є загальномовне середовище виконання Common Language Runtime (CLR), завдяки чому .NET підтримує кілька мов: поряд із C# це також VB.NET, C++, F#, і навіть різні діалекти інших мов, прив'язані до .NET, наприклад, Delphi.NET. При компіляції код на будь-якій із цих мов компілюється у збірку мовою CIL (Common Intermediate Language) – свого роду асемблер платформи .NET. Тому за певних умов ми можемо зробити окремі модулі однієї програми на окремих мовах.

.NET є платформою, що переноситься (з деякими обмеженнями). Наприклад, остання версія платформи на даний момент – .NET 7 підтримується на більшості сучасних ОС Windows, MacOS, Linux. Використовуючи різні технології на платформі .NET, можна розробляти програми мовою C# для різних платформ – Windows, MacOS, Linux, Android, iOS, Tizen.

.NET представляє єдину для всіх підтримуваних мов бібліотеку класів. І який би додаток ми не збиралися писати на C# – текстовий редактор, гру, чат або складний веб-сайт – так чи інакше ми використовуємо бібліотеку класів .NET.

Загальномовне середовище виконання CLR та базова бібліотека класів є основою цілого стеку технологій, які розробники можуть задіяти під час

побудови тих чи інших додатків. Наприклад, для роботи з базами даних у Ацьому стеку технологій призначено технологію ADO.NET та Entity Framework Core. Для побудови графічних додатків із багатим насиченим інтерфейсом – технологія WPF та WinUI, для створення більш простих графічних додатків – Windows Forms. Для розробки кросплатформних мобільних та десктопних програм – Xamarin/MAUI. Для створення веб-сайтів та веб-додатків – ASP.NET і т.д. До цього варто додати активний Blazor-фреймворк, що розвивається і набирає популяності, який працює поверх .NET і який дозволяє створювати веб-додатки як на стороні сервера, так і на стороні клієнта. А в майбутньому підтримуватиме створення мобільних додатків та, можливо, десктоп-додатків.

Згідно з результатами тестування веб-програми на .NET у ряді випадків сильно випереджають веб-програми, побудовані за допомогою інших технологій. Програми на .NET в принципі відрізняються високою продуктивністю.

Також слід відзначити таку особливість мови C# і фреймворку .NET, як автоматичне складання сміття. А це означає, що нам здебільшого не доведеться, на відміну від C++, дбати про звільнення пам’яті. Вищезгадане загальномовне середовище CLR саме викличе збирач сміття та очистить пам’ять» [5].

## 1.4 Огляд засобів розробки програм ігрового характеру для тренування уваги

Результати навчання студента великою мірою залежать від того, наскільки він уважно слухає матеріал на заняттях. Однак увага потрібна не лише у навчанні, а й у повсякденному житті. Потрібно бути уважними на дорозі, щоби не попасти в аварію або не спричинити її самому. Розсіяна увага часто є причиною жахливих неприємностей: можна просто зіткнутися з ким-небудь, можна обпектися від нагрівального приладу, можна спіткнутися об якусь перешкоду, або навіть впасти в люк.

Тренування уваги потрібне для:

– виконання будь-якої роботи у коротші терміни і до того ж більш якісно, адже ви будете зосереджені на виконанні основного завдання;

– швидкого реагування на ситуацію, яка динамічно змінюється, а концентрація на головному та вміння приймати правильні рішення є дуже важливими, наприклад, для керівника;

– тренування пам'яті; чим уважніше ви читаєте, тим більше в змозі запам’ятати.

Цей список можна продовжувати і продовжувати, але й з наведеного зрозуміло, чому увага є надто важливою і чому увагу потрібно тренувати. Саме тому у теперішній час розроблено багато програм ігрового характеру, направлених на тренування уваги. У таблиці 1.1 дано коротку характеристику деяких популярних ігор для дітей, які мають за мету тренування уваги.

Таблиця 1.1 – Програми ігрового характеру для тренування уваги дітей

| № п/п | Назва гри | Піктограма гри | Здібності, які розвиває гра |
|---|---|---|---|
| *1* | *2* | *2* | *3* |
| 1 | Нумізмат |  | Увага, швидкість мислення |
| 2 | Звуквар |  | Увага, слухова пам’ять |
| 3 | Шлях-ніндзя |  | Увага, точність пам’яті, обсяг пам’яті |
| 4 | Звіздар |  | Увага, обсяг пам’яті, усний рахунок |
| 5 | Світлячки |  | Увага, периферійний зір |
| 6 | Шкарлупки |  | Увага, обсяг пам’яті |
| 7 | Особливе завдання |  | Увага, швидкість мислення |

Розроблення ігор на сьогоднішій день – це велика і самостійна галузь, яка швидко, постійно і динамічно розвивається, для чого використовуються різноманітні програмні засоби [6-15].

## 1.5 Огляд програмних засобів розробки тестів для перевірки знань

Перевірка знань студентів за допомогою тестування є поширеною практикою в українських ВНЗ і використовується не лише під час онлайн навчання. Тестування дозволяє швидко перевірити рівень засвоєння студентами матеріалу та автоматично отримувати показники успішності. Підбираючи матеріали та готуючи завдання для тестів, слід враховувати наступні рекомендації:

– обсяг, інтенсивність та рівень складності завдань має бути вдвічі менший, ніж на звичайних заняттях;

– необхідно постійно оцінювати складність завдань та за потреби замінювати їх на легші та доступніші;

– варто структурувати навчальний матеріал, створюючи папки із завданнями на гугл-диску чи інших ресурсах;

– слід використовувати різні типи завдань – тести, розгорнуті відповіді, вікторини, ігри, спільні проєкти тощо.

Для розробки тестів використовуються потужні програмні засоби [16-19], частина з яких наведена у таблиці 1.2.

Таблиця 1.2 – Програмні засоби для розробки тестів

| № п/п | Назва програми або онлайн-сервісу | Коротка характеристика програмного засобу або онлайн-сервісу |
|---|---|---|
| *1* | *2* | *3* |
| 1 | MyTest | Система програм у складі редактора тестів, тестувальника та сервера для проведення тестів у локальній мережі |

Продовження таблиці 1.2

| *1* | *2* | *3* |
|---|---|---|
| 3 | Hot Potatoes | Програма для створення інтерактивних тестів |
| 4 | Rich Test | Комплекс програм з вільним доступом |
| 5 | eTest | Програмний комплекс для підготовки тестів |
| 6 | PikaTest | Система програм для створення дворівневих тестів |
| 7 | TextDel | Сукупність програм для тестування знань школярів |
| 8 | EeasyQuizzy | Програма для створення та редагування тестів |
| 9 | SanRav TestOfficepro | Кожен тест є незалежною програмою |
| 10 | AnsTester | Програмне середовище для контролю знань |
| 11 | Google Forms | Онлайн-сервіс для підготовки тестів та проведення тестування |
| 12 | Microsoft Forms | Онлайн-сервіс для тестування знань, що входить до складу пакету Office 365 |
| 13 | Online Test Pad | Онлайн-сервіс для створення тестів та проведення тестування |
| 14 | OpenTest | Онлайн-сервіс з вільним доступом |
| 15 | Moodle | Онлайн-сервіс платформа для організації навчання, підготовки тестів та проведення тестування |
| 16 | Kahoot | Онлайн-сервіс для організації навчання, підготовки тестів та проведення тестування |
| 17 | Classtime | Онлайн-сервіс орієнтований на використання бази готових запитань для проведення тестування |

Додатки-ігри та програми-тести, розроблені у даній кваліфікаційній роботі, не є конкурентами до наведених у таблиці 1.1 потужних автономних програмних засобів та онлайн-сервісів, адже наша мета інша, а саме – продемонструвати уміння здобувача розробляти подібного роду програми.

# РОЗДІЛ 2
# ІНСТРУМЕНТИ ДЛЯ РОЗРОБКИ ПРОГРАМНИХ ПРОЕКТІВ ЗАСОБАМИ C# WINFORMS

## 2.1 Головне вікно C#WinForms-проекту та його елементи

Програмні додатки у середовищі C# WinForms, як правило, створюються у вигляді проектів [20-25]. Порожній C# WinForms проект створюється засобами середовища Visual Studio. Власне сам процес створення проекту закінчується відкриттям головного вікна. У вікні обов'язково будуть головне меню команд, панель інструментів та інші засоби створення проекту, але зараз ми більш детально зупинимося на вікнах середовища, оскільки з ними найчастіше приходиться працювати під час розробки проектів.

Перш за все нам стає доступним вікно головної форми проекту Form1 (рис. 2.1). Більш детально форми ми розглянемо трохи пізніше. Зазначимо лише, що проект, як правило, складається з багатьох форм, але одна (головна) форма у ньому є обов'язковою. Саме на форми ми переносимо готові компоненти, з яких і компонуємо проект, тому форми є самі по собі дуже важливими компонентами.

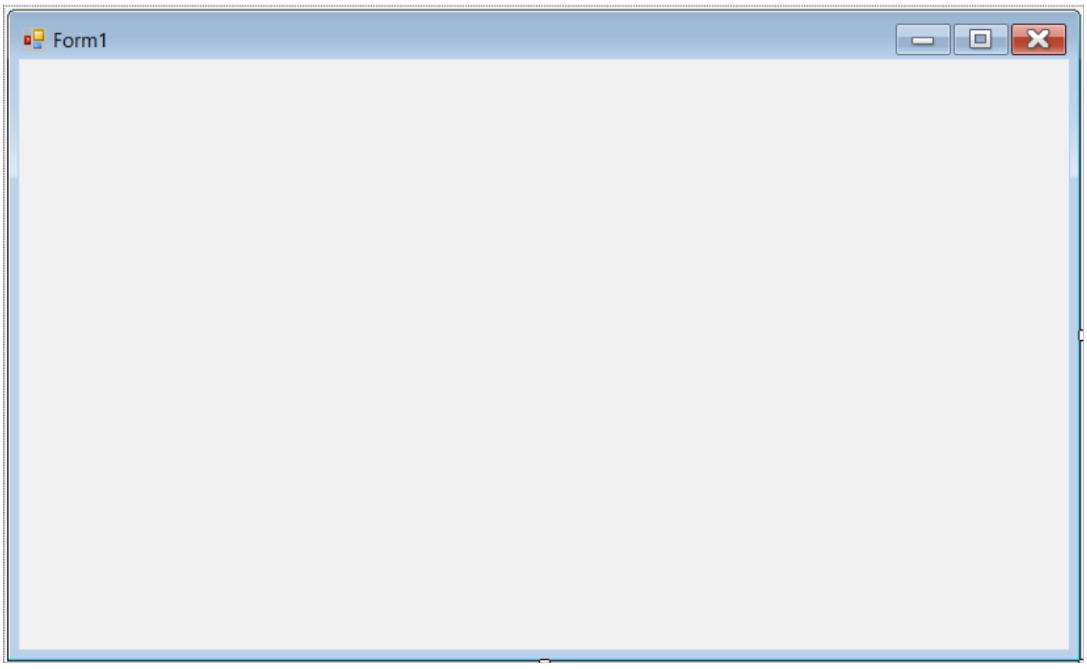


Рисунок 2.1 – Вигляд головної форми Form1 проекту

Компоненти, які необхідні для створення проекту, ми можемо вибрати за допомогою вікна Панель інструментів (Toolbox) (рис. 2.2). Всі компоненти цього вікна поділені на такі категорії:

- всі компоненти (All Windows Forms);
- найбільш поширені компоненти (Common Controls);
- компоненти для створення меню (Menus & Toolbars);
- компоненти-контейнери (Containers);
- компоненти для роботи з даними (Data);
- компоненти різнопланового призначення (Components);
- компоненти для організації друку документів (Printing);
- компоненти, що базуються на діалогових вікнах (Dialogs);
- компоненти для роботи з формами WPF (WPF Interoperability);
- компоненти загального призначення (General).

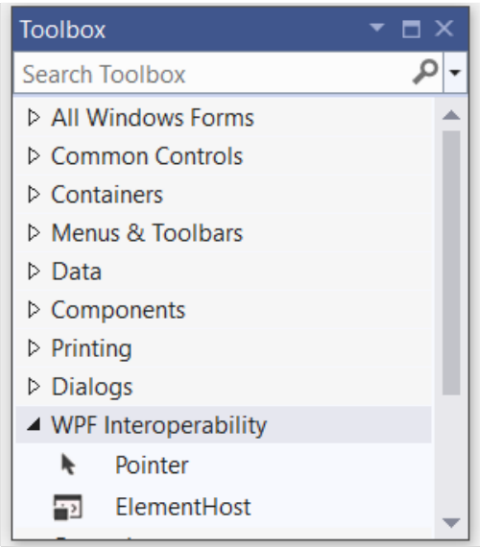


Рисунок 2.2 – Панель інструментів

Клацнувши по вкладці Common Controls, отримаємо доступ до компонентів цієї вкладки (рис. 2.3). Саме з цієї вкладки ми будемо найчастіше вибирати компоненти для нашого проекту. Зокрема, це компоненти Button (командна кнопка), Label (мітка), TextBox (текстове поле), RadioButton

(радіокнопка) та багато інших.

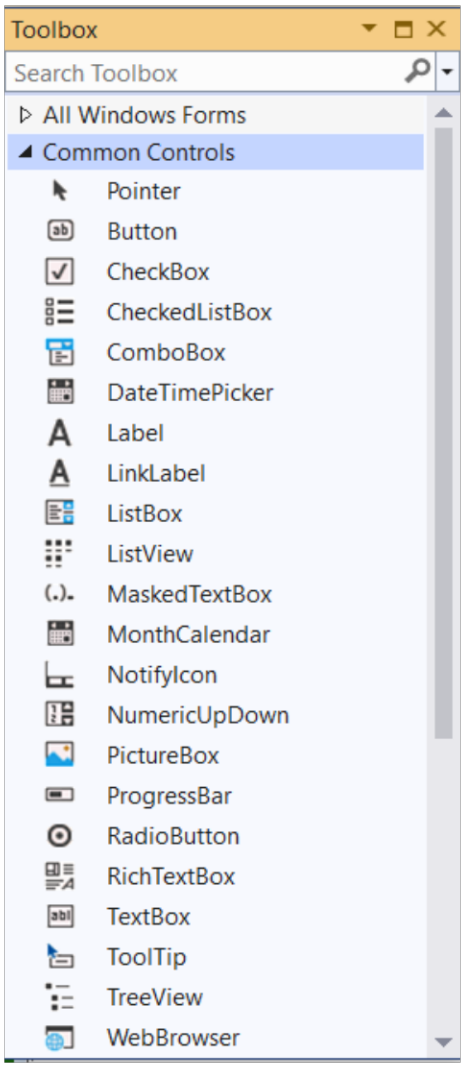


Рисунок 2.3 – Вкладка Common Controls

Для того, щоб перенести компонент з Панелі елементів на форму, потрібно клацнути по ньому, і, утримуючи натиснутою кнопку миші, клацнути по формі у місці, де цей компонент має бути розташований, а далі відпустити кнопку миші. У результаті буде створений об’єкт відповідного класу. Якщо, наприклад, на вкладці Common Controls виділити компонент Label і перенести його на форму, то буде створений об’єкт label1, який успадковує всі властивості класу Label. З об’єктами тісно пов’язані такі поняття, як Властивості та Події.

Властивості об’єкта можна змінювати, підлаштовуючи їх під свої потреби. Для цього потрібно налаштувати цей об’єкт, використовуючи вікно

Властивості (рис. 2.4). У верхній частині вікна розміщені важливі піктограми, які використовуються під час налаштування об’єктів. Перші дві піктограми (зліва направо) використовується для налаштування властивостей об’єкта, групуючи їх відповідно або за категоріями, або за алфавітом. Дві наступні піктограми використовуються під час роботи з подіями. Вони групують події відповідно або за категоріями, або за алфавітом.

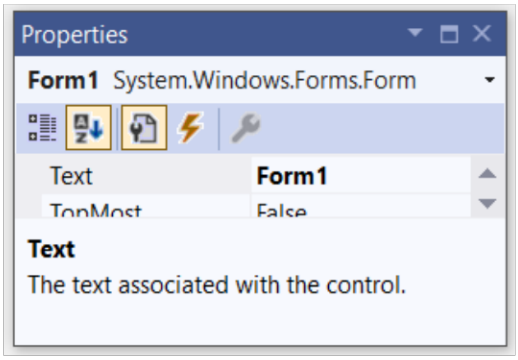


Рисунок 2.4 – Вікно Властивості (Properties)

Для того, щоб налаштувати властивість об’єкта, потрібно у лівому стовпчику вікна Властивості виділити потрібну властивість, і напроти у правому стовпчику задати її значення. У такий спосіб ми можемо змінити початкові налаштування об’єкта, які пропонує відповідний клас, на потрібні для нас. При цьому не потрібно винаходити велосипед, ми просто використовуємо заздалегідь розроблені компоненти (класи). Це велика перевага технології об’єктно-орієнтованого програмування. Завдяки технології ООП значно полегшується процес розробки проектів.

Для того, щоб запрограмувати обробник певної події, потрібно у лівому стовпчику вікна Події виділити потрібну подію і напроти у правому стовпчику вибрати зі списку і клацнути по імені відповідного обробника події. З’явиться заготовка функції-обробника, яку належить відповідним чином запрограмувати. Обробник події повинен забезпечувати той функціонал, який з цією подією пов’язаний.

Важливим є вікно браузера проектів (Solution Explorer) (рис. 2.5), яке

дуже часто використовується для роботи зі складовими елементами проекту, тобто формами, файлами, класами і т.п. Наприклад, якщо потрібно додати до проекту ще одну форму, варто клацнути правою кнопкою миші по назві проекту; з'явиться спадне меню, у якому належить клацнути по вкладці Add (додати), з'явиться ще одне спадне меню, у якому вибрати Form (Windows Forms), і клацнути по кнопці Ok. У проект буде додано форму Form2.

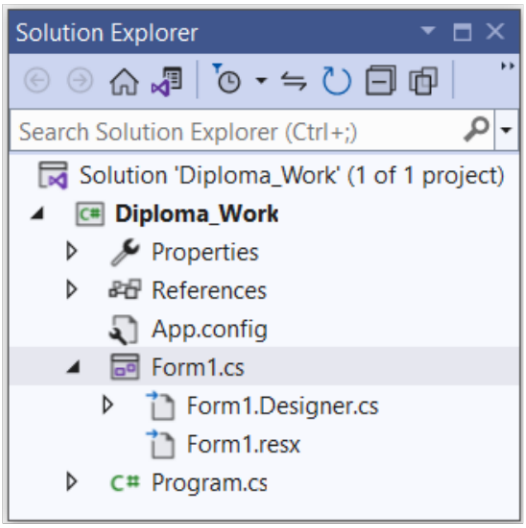


Рисунок 2.5 – Вікно браузера проектів (Solution Explorer)

Проект, як правило, містить багато форм, файлів, класів [26-30]. Тому браузер проектів є вкрай необхідним інструментом, за допомогою якого ми викликаємо потрібний нам елемент проекту. Зауважимо, що з формою Form1 пов'язані три файли: Form1.cs, Form1.cs (Design) та Form1.Designer.cs. Файл Form1.cs містить код форми, файл Form1.cs (Design) безпосередньо відображує форму на екрані, файл Form1.Designer.cs містить код конструктора форми. З кожною наступною формою проекту пов'язані подібні файли. Щоб їх активувати, потрібно у браузері проектів клацнути по значку-трикутнику навпроти потрібної форми, і у списку доступних файлів вибрати потрібний.

У спосіб, описаний вище, можна виконувати велику множину маніпуляцій з проектом: перейменовувати, переміщати чи вилучати елементи і багато інших.

Під час відлагодження проекту корисними є ще два вікна – Список

помилок (Error List) та Результати (Output) (рис. 2.6).

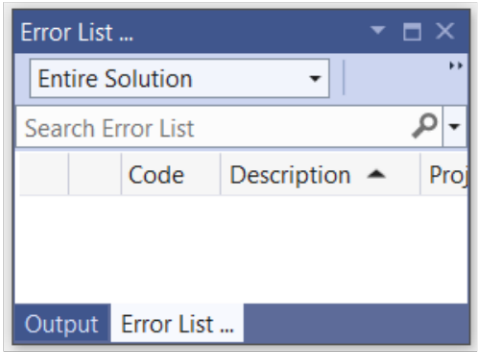


Рисунок 2.6 – Вікна Список помилок (Error List) та Властивості (Properties)

## 2.2 Призначення та характеристика компонента Form

Форма – це компонент проекту, призначений для розміщення на ньому всіх інших компонентів, необхідних для розв'язання тих чи інших завдань проекту. Кожна форма є нащадком класу Form, а тому успадковує всі його властивості і має доступ до його методів. Перша форма проекту називається ще головною, і бажано, щоб ім'я файла головної форми завжди було доступним на панелі задач. З огляду на важливість форм, розглянемо детально її властивості. Вони наведені у таблиці 2.1.

Таблиця 2.1 – Властивості компоненти Form

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| *1* | *2* | *3* |
| 1 | Name | Ім'я форми та змінної відповідного об'єкта, яка використовується у коді для доступу до властивостей об'єкта або виклику його методів. |
| 2 | Cursor | Дозволяє вибрати вигляд курсора, який буде відображатися у момент, коли курсор миші знаходиться над поверхнею форми. Тип курсора вибирається зі спадного списку, у якому відображаються усі курсори, що пропонуються. |

Продовження таблиці 2.1

| *1* | *2* | *3* |
|---|---|---|
| 3 | BackgroundColor | Колір фону форми. Ця властивість має тип класу Color. Якщо клацнути у полі напроти даної властивості, то з'явиться кнопка спадного списку. Якщо клацнути по ній, з'явиться панель для вибору кольору фону. На панелі знаходяться три вкладки: Custom (Довільні кольори), Web (Web кольори), System (Системні кольори). Для вікон бажано вибрати один із системних кольорів, наприклад, колір типу Control для діалогових вікон та колір типу Window для вікон, які призначені для відображення даних. |
| 4 | Enabled | Форма стає недоступною і не буде реагувати на дії користувача, якщо ця властивість дорівнює false. |
| 5 | Font | Шрифт, який використовується для виведення тексту. Дана властивість набагато більш актуальна для тих об'єктів, які містять надписи. Є два способи зміни шрифта.<br>Перший спосіб: виділити стрічку зі властивістю Font і у клацнути по кнопці з трьома крапками на ній; з'явиться вікно для задання властивостей шрифта.<br>Другий спосіб: клацнути по кнопці з хрестиком лівворуч слова Font; також з'явиться вікно для задання властивостей шрифта.<br>Шрифт задається такими параметрами: Ім'я (Name), Розмір (Size), Жирний (Bold), Курсив (Italic), Закреслений (Strikecut), Підкреслений (Underline). |

Продовження таблиці 2.1

| *1* | *2* | *3* |
|---|---|---|
| 6 | ForeColor | Колір переднього плану, який найчастіше використовується як колір тексту. |
| 7 | FormBorderStyle | Стиль контура форми. Вибирається зі спадного списку, який є переліком таких значень: None – контура не буде; FixedSingle – тоненький контур, який не дозволяє змінювати розміри форми; Fixed3D – тоненький трьохмірний контур, який не дозволяє змінювати розміри форми; FixedDialog – тоненький контур, який не дозволяє змінювати розміри форми і у заголовку форми не буде піктограми; Sizable – стандартний контур, який дозволяє змінювати її розміри. |
| 8 | Icon | Задає піктограму (значок) для форми. Тут є кнопка для відкриття файла, що містить зображення значка. Збережений значок знаходитеметься у файлі .resx. |
| 9 | Location | Визначає положення форми відносно верхнього лівого кута екрана. Задається значеннями графічних координат X та Y. |
| 10 | MainMenuStrip | Використовується для створення меню команд форми. Якщо на формі встановити компонент MenuStrip, то властивість MainMenuStrip можна вибрати зі спадного списку цього компонента. |
| 11 | MaximizeBox | Має логічне значення. Якщо воно дорівнює true, то вікно розтягується до максимальних розмірів. |
| 12 | MaximumSize | Задає максимальні ширину та висоту форми. |
| 13 | MinimumSize | Задає максимальні ширину та висоту форми. |

Продовження таблиці 2.1

| *1* | *2* | *3* |
|---|---|---|
| 14 | MinimizeBox | Має логічне значення. Якщо воно дорівнює true, то вікно зменшується до мінімальних розмірів. |
| 15 | Opacity | Відсоток непрозорості форми. За замовчуванням, вікно абсолютно непрозоре, тобто даний параметр дорівнює 100%. |
| 16 | Padding | Відступи від меж форми. |
| 17 | ShowIcon | Логічне значення, яке показує, чи потрібно відображати у заголовку форми її піктограму. |
| 18 | ShowInTaskBar | Визначає, чи потрібно форму відображати на панелі задач. Рекомендується включати для головної форми проекту. |
| 19 | Size | Ширина та висота форми. |
| 20 | StartPosition | Початкова позиція форми на екрані. Вибирається зі спадного списку із п’яти значень. |
| 21 | Tag | Має тип Object, тому у цій властивості можна зберегти будь-яку інформацію. |
| 22 | Text | Текст, що відображається у заголовку форми |
| 23 | TopMost | Логічне значення, яке визначає, чи має дана форма розташовуватися зверху над іншими вікнами. |
| 24 | WindowState | Статус вікна. Якщо вказати значення Normal, то форма отримає нормальний статус; значення Maximized розтягне форму на весь екран; значення Minimized зменшить форму до мінімальних розмірів. |

Клас Form містить ряд методів, які успадковуються усіма об’єктами цього класу. До цих методів належать:

– Show() – Відображує створене вікно немодально. Немодальне вікно не блокує виконання батківського вікна (в якому воно було створене). Це означає, що обидва вікна можуть працювати паралельно. Це дає змогу переключатися між вікнами.

– ShowDialog() – Відображує вікно модально. Інші вікна не можна відкрити, поки не закрите дане вікно. Це означає, що два вікна не можуть працювати паралельно.

– Hide() – Сховати вікно, не видаляючи його. Має смисл для немодальних вікон.

– Close() – Закрити форму. У цьому випадку об’єкт форми вилучається.

– Invalidate() – Перемалювати форму.

Кожен проект має реагувати на певні події, у C# проекті ми повинні передбачити обробку цих подій. Щодо форм, то з ними можуть бути пов’язані такі події:

– Load – завантаження;

– Activate – активація;

– VisibleChanged – зміна видимості;

– Reactived – деактивація;

– Closing – закриття (ще можна відмінити);

– Close – закриття.

## 2.3 Призначення та характеристика деяких компонент C# WinForms

### 2.3.1 Компонент Button

Для чого потрібні кнопки? Хоча би для того, щоб по них клацати. Клацнувши по формі, ми розраховуємо на певну реакцію проекту. Але ця реакція можлива, якщо подію клацання по кнопці певним чином запрограмувати. Наприклад, обробник події Click кнопки button1 збільшить значення змінної var_first на 10.

```
private void button1_Click(j,ject sender, EventArgs e)
```

```
{ var_first += 10;}
```

Якщо, наприклад, ми хочемо, щоб клацання по кнопці button1 закривало форму Form1 і відкривало форму Form2, обробник події клацання по формі виглядатиме так:

```
private void button1_Click(j,ject sender, EventArgs e)
{    this.Hide();
     Form2 f2 = new Form2();
     f2.Show();
}
```

Але за допомогою кнопки можна багато чого зробити, зовсім не клацаючи по ній. Наприклад, за допомогою властивості Image кнопки ми можемо викликати на кнопку певну картинку чи фото з попередньо підготовленого файлу. Тоді на кнопці ми побачимо відповідне зображення. Наведемо деякі властивості компонента Button (табл. 2.2).

Таблиця 2.2 – Властивості комонента Button

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта |
| 2 | BackColor | Колір заднього фону |
| 3 | Font | Параметри шрифта |
| 4 | Size | Розміри кнопки |
| 5 | Text | Текст, що відображається на формі |
| 6 | TextAllign | Спосіб вирівнювання тексту |

### 2.3.2 Компонент TextBox

Компонент TextBox служить для введення даних користувачем. Якщо нам потрібно ввести число, стрічку чи символ, ми використовуємо саме цей компонент. Текст, який ми ввели у текстове поле, відобразиться на екрані. Тут доречно акцентувати увагу на властивості Lines, яка виводить вікно для введення у нього потрібного для компонента TextBox. Цей текст у згаданому

вікні можна також редагувати, щоб отримати текст потрібного вигляду. У таблиці 2.3 наведені ще деякі властивості компонента TextBox.

Таблиця 2.3 – Властивості компонента TextBox

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |
| 2 | AutoSize | Якщо значення властивості дорівнює true, розміри текстового поля встановлюються за замовчуванням. Інакше розміри можна змінювати |
| 3 | Lines | Викликає вікно, у якому вводиться багатострічковий текст. |
| 4 | Miltiline | Ящо значення дорівнює true, то встановлюється режим багатострічкового текстового документа. |
| 5 | Size | Розміри компонента. |
| 6 | UseSystemPassword | Текст, що вводиться, відображається на формі крапками. |
| 7 | TextAllign | Спосіб вирівнювання тексту. |

### 2.3.3 Компонент TableLayoutPanel

Компонент TableLayoutPanel відносится до категорії контейнерів і служить для групування і розміщення на ньому у вигляді таблиці групи компонентів, наприклад, міток. Розміри стрічок та стовпчиків таблиці TableLayoutPanel можна задавати як у абсолютних, так і у відносних одиницях. Таблиця TableLayoutPanel може бути притиснутою до низу чи верху форми, до її лівого чи правого боку. У таблиці 2.4 наведені деякі властивості компонента Таблиця TableLayoutPanel.

Таблиця 2.4 – Властивості комонента TableLayoutPanel

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |

### 2.3.4 Компонент NumericUpDown

Компонент NumericUpDown відноситься до категорії Common Controls і служить для відображення на ньому певних числових даних, наприклад, результатів виконання арифметичних операцій. У таблиці 2.5 наведені деякі властивості компонента NumericUpDown.

Таблиця 2.5 – Властивості комонента NumericUpDown

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта |
| 2 | AutoSize | Якщо значення властивості дорівнює true, розміри компонента встановлюються за замовчуванням. |
| 3 | BackColor | Колір заднього фону |
| 4 | Font | Параметри шрифта |
| 5 | Size | Розміри компонента |

### 2.3.5 Компонент GroupBox

Компонент GroupBox відноситься до категорії контейнерів і служить для групування і розміщення на ньому таких компонентів, як RadioButton, CheckBox, ListBox, ComboBox. У таблиці 2.6 наведені деякі властивості компонента GroupBox.

Таблиця 2.6 – Властивості комонента GroupBox

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |
| 2 | AutoSize | Якщо значення властивості дорівнює true, розміри компонента встановлюються за замовчуванням. |
| 3 | BackColor | Колір заднього фону |
| 4 | Font | Параметри шрифта |
| 5 | Text | Заголовок групи елементів |

2.3.6 Компонент RadioButton

Компонент RadioButton відносится до категорії Common Controls і служить для вибору лише одного компонента із групи таких самих компонентів. У таблиці 2.7 наведені деякі властивості компонента RadioButton.

Таблиця 2.7 – Властивості комонента RadioButton

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |
| 2 | Appearance | Значення, що визначає зовнішній вигляд об’єкта. |
| 3 | AutoCheck | Значення, що визначає чи буде автоматично змінюватися значення Checked. |
| 4 | AutoEllipsis | Значення, яке показує, чи відобразиться знак (…). |
| 5 | CanFocus | Значення, яке показує, чи може об’єкт отримувати фокус. |

2.3.7 Компонент CheckBox

Компонент CheckBox відносится до категорії Common Controls і служить для вибору одного або кількох компонентів із групи таких самих компонентів. У таблиці 2.8 наведені деякі властивості компонента CheckBox.

Таблиця 2.8 – Властивості компонента CheckBox

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |
| 2 | Checked | Повертає значення true або false, залежно від того чи чи є прапорець в компоненті. |
| 3 | CheckedState | Повертає значення Checked або Unchecked залежно від значення властивості ThreeState. |

2.3.8 Компонент ListBox

Компонент ListBox відносится до категорії Common Controls і служить

для для організації вибору одного елемента списку із множини таких елементів. У таблиці 2.9 наведені деякі властивості компонента ListBox.

Таблиця 2.9 – Властивості комонента ListBox

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |
| 2 | Items | Зберігає набір всіх елементів списку |
| 3 | Count | Зберігає ількість елементів списку. |

2.3.9 Компонент ComboBox

Компонент ComboBox відносится до категорії Common Controls і служить для організації вибору одного елемента списку із множини таких елементів. У таблиці 2.10 наведені деякі властивості компонента ComboBox.

Таблиця 2.10 – Властивості комонента ComboBox

| № п/п | Назва властивості | Опис властивості |
|---|---|---|
| 1 | Name | Ім’я кнопки та змінної відповідного об’єкта. |
| 2 | Count | Кількість елементів у списку. |
| 3 | SelectedIndex | Повертає ціле значення, яке відповідає вибраному елементу списку. |
| 4 | SelectedItem | Повертає сам вибраний елементу списку. |

# РОЗДІЛ 3
# РОЗРОБКА КОМПЛЕКСУ ПРОГРАМ ДЛЯ ТРЕНУВАННЯ УВАГИ ТА ПЕРЕВІРКИ ЗНАНЬ

## 3.1 Структура комплексу програм та навігація по ньому

Комплекс програм, розроблений у кваліфікаційній роботі, реалізований у вигляді C# WinForms проекту з назвою DiplomaWork. Цей проект складається з дев’яти форм, перша і п’ята з яких зображені на рисунках 3.1 та 3.2. Вони мають інформаційний характер, а на решті форм реалізовані власне програми комплексу. Призначення кожної форми проекту наведені у таблиці 3.1. Фактично комплекс програм реалізований у вигляді двох ігор та чотирьох тестів. Перша гра «Знайдіть пару» реалізована на формах Form2 та Form3, друга гра «Математичний тест» реалізована на формі Form4, а тести різної складності та спрямування – на формах Form6, Form7, Form8 та Form9.

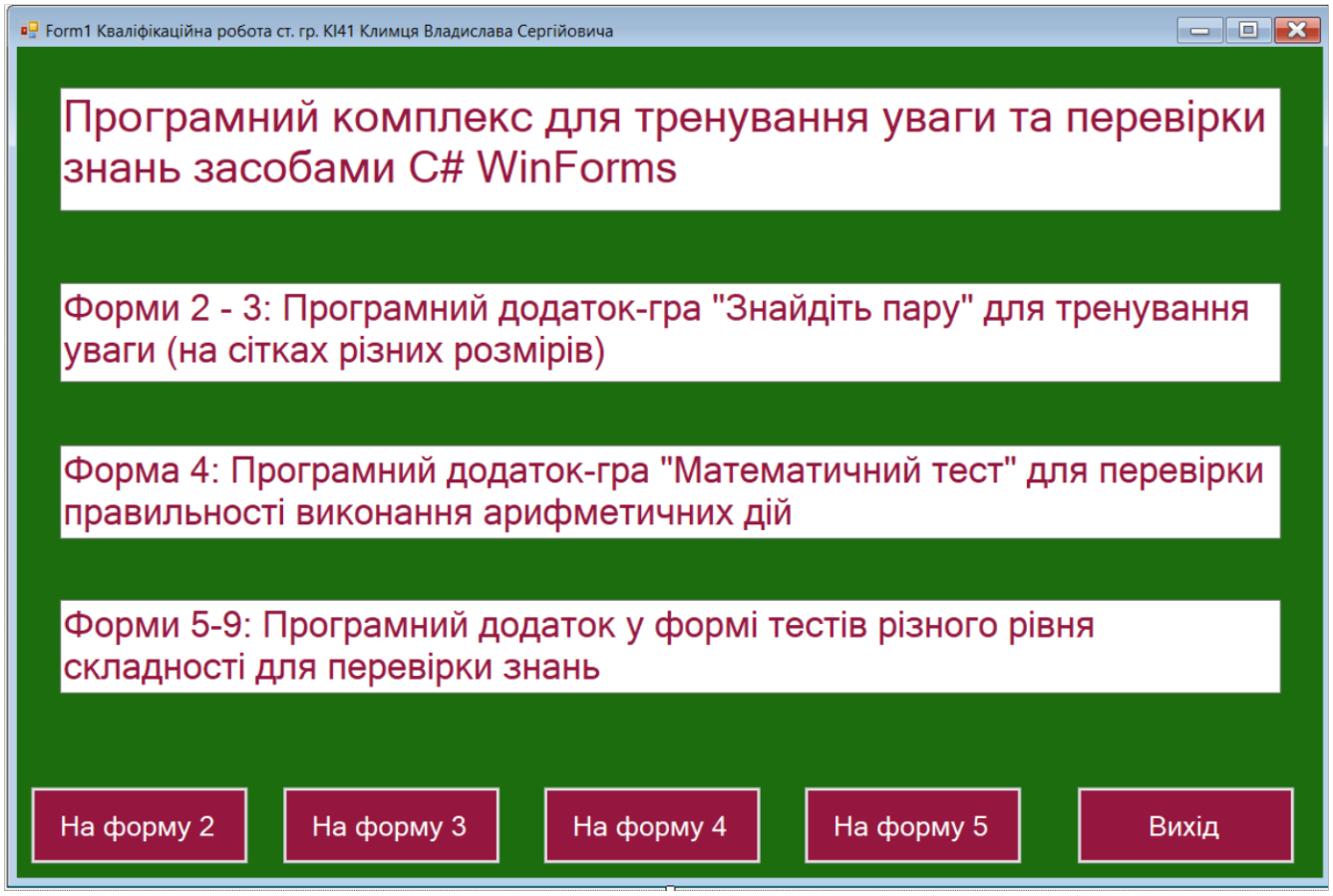


Рисунок 3.1 – Вигляд головної форми Form1 проекту

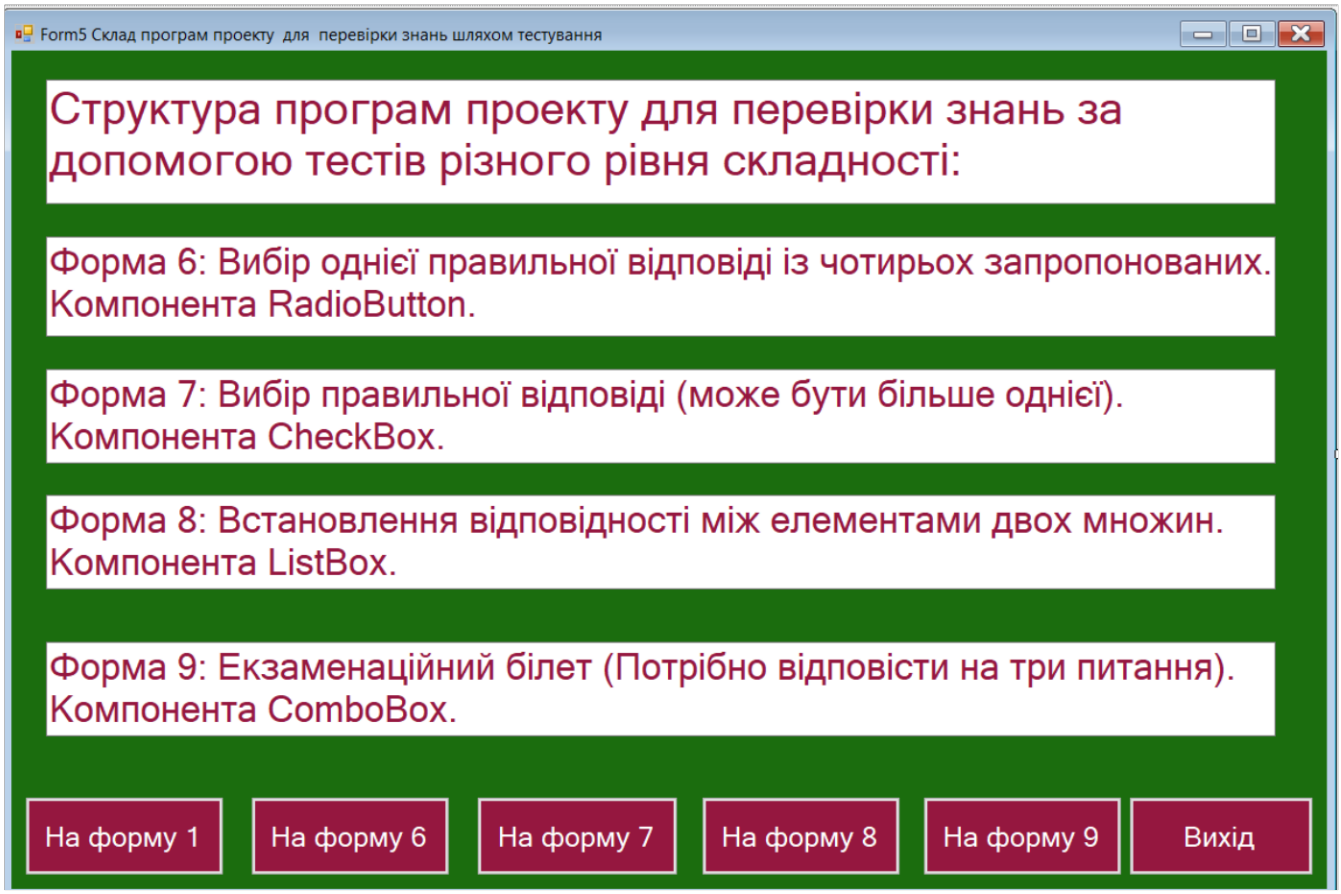


Рисунок 3.2 – Вигляд інформаційної форми Form5 проекту

Таблиця 3.1 – Призначення форм проекту

| № п/п | Назва форми | Призначення форми |
|---|---|---|
| 1 | Form1 | Інформація про вміст форм Form1 – Form5 |
| 2 | Form2 | Додаток-гра «Знайдіть пару 4*4» |
| 3 | Form3 | Додаток-гра «Знайдіть пару 4*5» |
| 4 | Form4 | Додаток-гра «Математчний тест» |
| 5 | Form5 | Інформація про вміст форм Form6 – Form9 |
| 6 | Form6 | Тест на базі компонент RadioButton |
| 7 | Form7 | Тест на базі компонент CheckButton |
| 8 | Form8 | Тест на базі компонент ListBox |
| 9 | Form9 | Тест на базі компонент ComboBox |

Перш за все потрібно вирішити питання навігації по формах проекту. Це досягається за допомогою командних кнопок Button, розташованих внизу кожної форми. Для кожної з них властивість Text замінена відповідним текстом. Цей текст підказує призначення кнопки. Так, наприклад, на першій кнопці форми Form1 надпис «На форму 2» означає, що, клацнувши по цій формі, ми перейдемо на форму Form2. Таким чином, з головної форми проекту можна перейти на всі інші форми та повернутися назад. Крім того, передбачено перехід з неголовних форм на сусідні форми, а також завершення роботи проекту за допомогою кнопок типу «Вихід», які є на всіх формах.

Для того, щоб забезпечити описану вище навігацію по формах проекту, кожна з них має бути відповідним чином запрограмована. Так, щоб перейти з форми Form1 на форму Form5, кнопка button4 форми Form1 має містити код, наведений на рисунку 3.3. Перша команда коду закриває форму Form1, друга – створює новий об’єкт класу Form5, за допомогою якого третя команда викликає метод Show(), що відкриває форму Form5. Навпаки, щоб із форми Form5 перейти на форму Form1, кнопка button1 форми Form5 має містити код, наведений на рисунку 3.4. Кнопки button з надписом «Вихід» на усіх формах забезпечують вихід з проекту з будь-якої форми завдяки коду, наведеному на рисунку 3.5.

```
private void button4_Click_1(object sender, EventArgs e)
{
    this.Hide();
    Form5 f5 = new Form5();
    f5.Show();
}
```

Рисунок 3.3 – Код командної кнопки button4 форми Form1

```
private void button1_Click(object sender, EventArgs e)
{
    this.Hide();
    Form1 f1 = new Form1();
    f1.Show();
}
```

Рисунок 3.4 – Код командної кнопки button1 форми Form5

```
private void button2_Click(object sender, EventArgs e)
{
    Application.Exit();
}
```

Рисунок 3.5 – Код командної кнопки button з надписом «Вихід» усіх форм проекту

### 3.2 Додаток-гра «Знайдіть пару 4*4» для тренування уваги

#### 3.2.1 Макет гри «Знайдіть пару 4*4» для тренування уваги

Програмний додаток для тренування уваги «Знайдіть пару 4*4» - це гра. Макет нашої гри – це прямокутна сітка (матриця), що складається з чотирьох стрічок та чотирьох стовпчиків, на перетині яких утворюється 16 клітин. У кожній клітині сітки має бути один елемент гри (значок), що обов'язково має мати пару в якійсь іншій клітині. У якій саме клітині сітки знаходиться значок-пара, ми не знаємо. Клацнувши по першій клітині сітки, ми бачимо зображення значка, що в ній знаходиться. Далі клацаємо по іншій клітині. Якщо у цій клітині знаходиться пара, зображення в обох клітинах вже не зникатимуть з екрана. Якщо ж пари не знайдено, то з екрана зникнуть зображення обох значків. Сутність гри полягає у тому, щоб віднайти пару для всіх значків за якомога коротший проміжок часу. Для цього слід уважно стежити за тим, які саме значки і в яких клітинах сітки відкривалися раніше, щоб у потрібний момент ми могли їх активувати. Ось у цьому і полягає суть тренування уваги.

Отже, спочатку створюємо макет гри. Для створення прямокутної сітки клацнемо по формі Form2, щоб викликати конструктор Windows Forms, тобто файл Form2.cs [Design]. У вікні Властивості встановлюємо такі значення властивостей форми Form2, як наведені у таблиці 3.2. Для того, щоб задати властивість BackColor за допомогою сайту «colorpicker.me», у вікно пошуку браузера Google вводимо текст «color picker», і зі запропонованих варіантів сайтів вибираємо і запускаємо сайт «colorpicker.me». З'явиться вікно сайту (рис. 3.6), у якому за допомогою повзуна візуально підбираємо підходящий

колір. У даному випадку ми вибрали для форми Form2 ніжноблакитний колір,

Таблиця 3.2 – Властивості форми Form2

| № з/п | Назва властивості | Значення параметра |
|---|---|---|
| 1 | Text | Текст «Знайдіть пару 4*4». Цей текст відображається у верхній частині вікна гри |
| 2 | Size | 1200; 800 (це ширина і висота форми, які можна встановити також перетягуючи кут форми до тих пір, поки не отримаємо форму потрібних розмірів) |
| 3 | BackColor | Встановлюємо за допомогою сайту «colorpicker.me». |

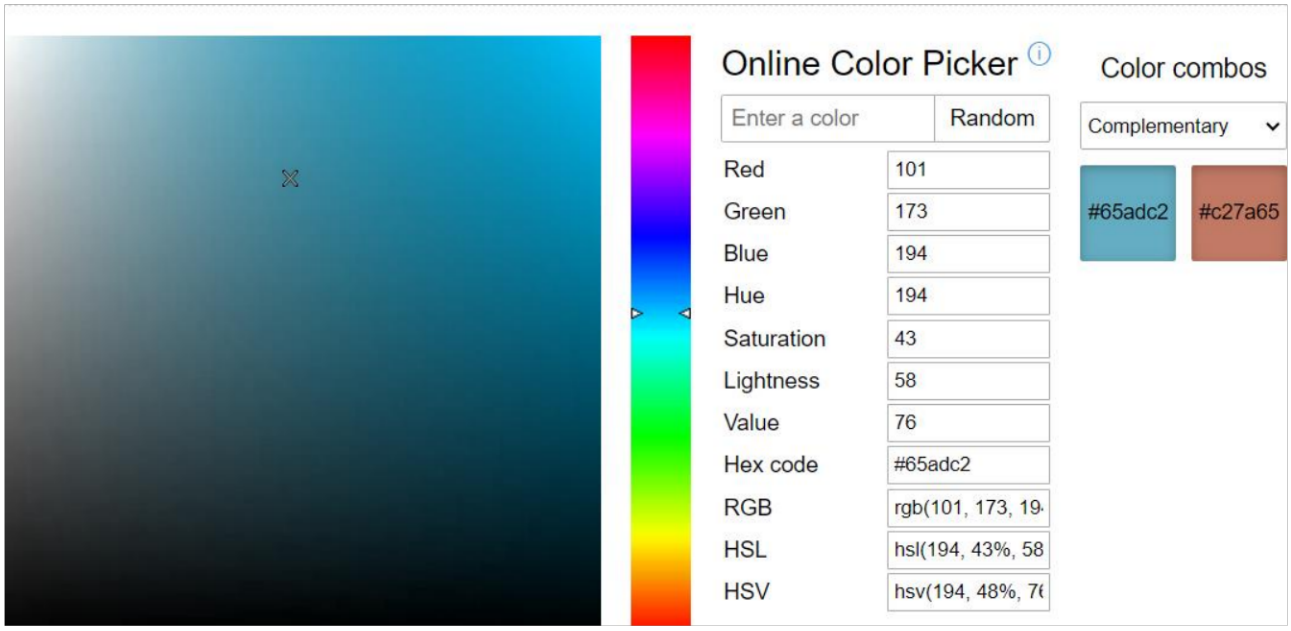


Рисунок 3.6 – Головне вікно сайту «colorpicker.me»

шістнадцятковий код #65adc2 якого копіюємо. Далі клацаємо по формі Form2, і у вікні Properties активуємо властивість BackColor (рис. 3.7), для якої вставляємо скопійоване значення коду кольору. До речі, вибраний колір відповідає трійці значень (101; 173; 194) за системкою RGB. Такий вибір кольору є набагато більш ефективним, ніж той, що пропонує програмне середовище C#.

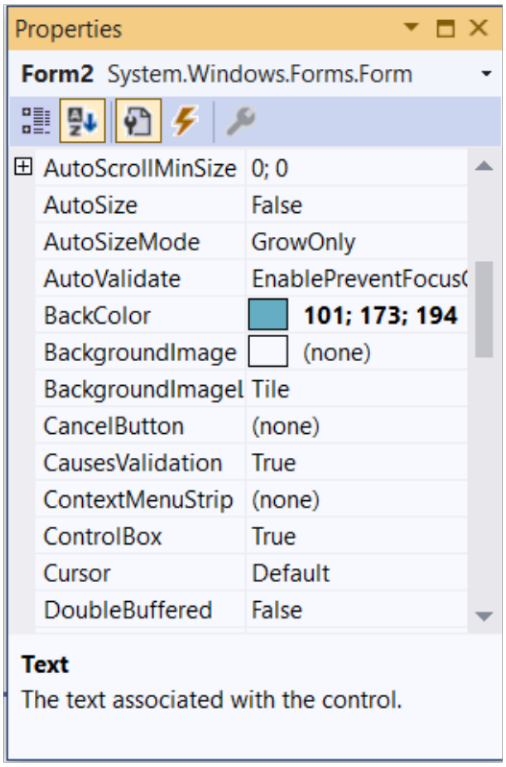


Рисунок 3.7 – Задання властивості BackColor форми Form2

Далі командою Вид>Панель елементів викликаємо вікно Панель елементів (Toolbox). Перетягуємо на форму Form2 компонент TableLayoutPanel із вкладки Контейнери. На формі Form2 з'являється сітка (рис. 3.8, а), що складається з двох стрічок та двох стовпців. Натискаємо трикутну кнопку у верхньому правому куті панелі, щоб відобразити меню завдань цієї панелі (рис. 3.8, б). У меню команд панелі клацаємо двічі по пункту Додати рядок та двічі по пункту Додати стовпець, щоб отримати сітку розміром 4*4 (рис. 3.8, с). Далі у меню завдань активуємо команду Правка, з допомогою якої встановлюємо для стовпців відносний розмір 25%, а для стрічок встановлюємо відносний розмір 20% (рис. 3.9) і натискаємо на кнопку ОК.

У вікні Властивості встановлюємо такі значення властивостей панелі tableLayoutPanel1, як наведені у таблиці 3.2. У результатів виконання всіх дій ми отримали макет гри «Знайдіть пару 4*4» (рис. 3.10) у вигляді прямокутної сітки (матриці), у клітинах якої мають бути розміщені значки, кожен з яких повинен мати пару. Саме тому макет гри може містити лише парну кількість клітин, причому чим більші розміри макету, тим складнішою є гра.

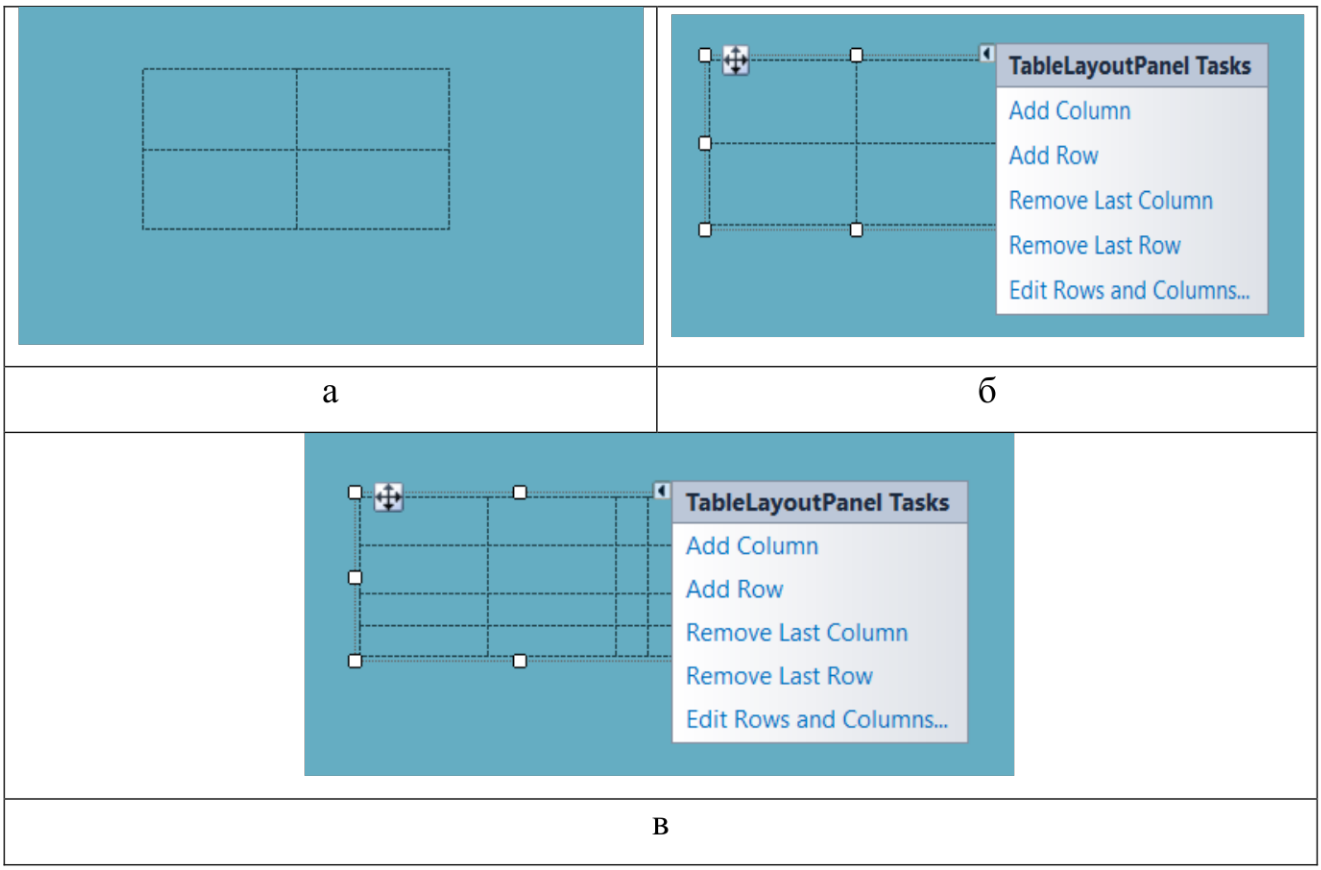


Рисунок 3.8 – Створення макету гри: а) Перенесення компоненти на форму; б) Відображення меню команд панелі; в) Додавання стовпчиків та стрічок

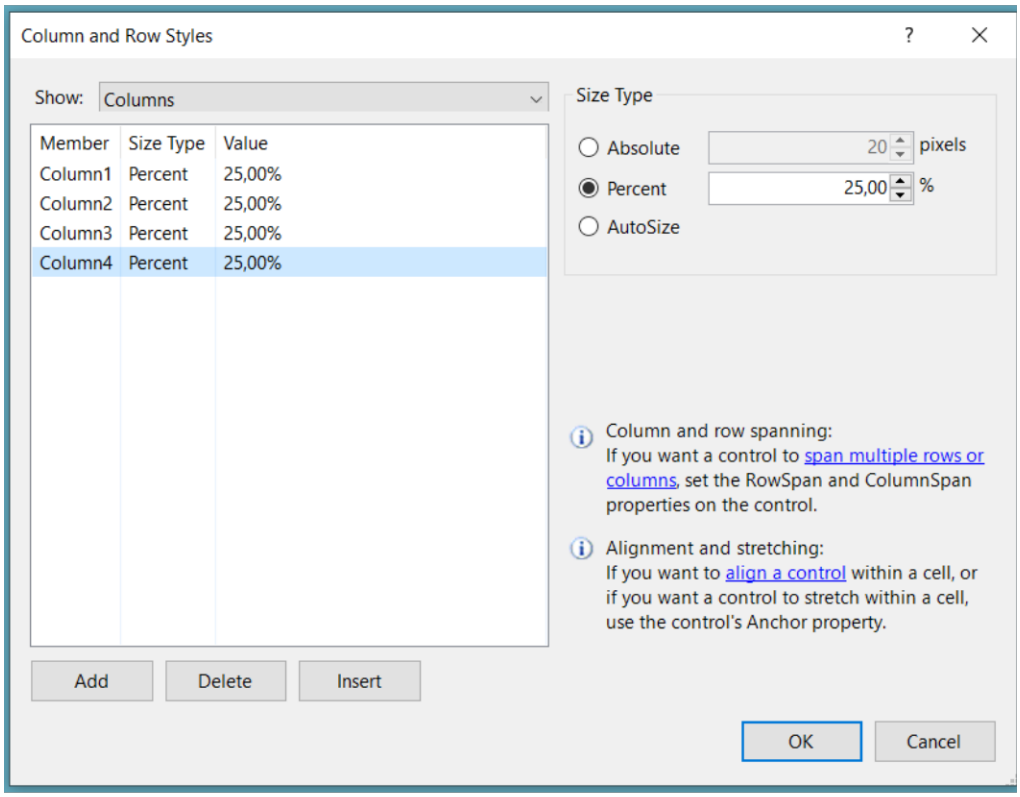


Рисунок 3.9 – Задання відносних розмірів стрічок та стовпчиків

Таблиця 3.3 – Властивості панелі tableLayoutPanel1

| № з/п | Назва властивості | Значення параметра |
|---|---|---|
| 1 | Dock | Top (цим ми притискуємо панель до верхньої межі форми) |
| 2 | Size | 1180; 580 (цим ми гарантуємо, що нижня частина форми не буде закрита панеллю) |
| 3 | BackColor | Встановлюємо за допомогою сайту «colorpicker.me». |
| 4 | CeilBorderStile | Inset (клітини матриці будуть розділені світлою смужкою). |


Рисунок 3.10 – Остаточний вигляд макету гри «Знайдіть пару 4*4»

### 3.2.2 Додавання міток для відображення значків

Вибираємо у вікні Панель елементів категорію Стандартні елементи керування. Елемент керування Label перетягуємо у верхню ліву комірку панелі tableLayoutPanel1. Встановлюємо для елемента label1 такі властивості, як наведені у таблиці 3.4. Далі копіюємо елемент label1 і вставляємо його

послідовно у всі клітини сітки. Тепер у кожній клітині макету відобразилися квадрати (рис. 3.11). Зазначимо, що шрифт Webdings постачається з операційною системою Windows. Символу ’c’ у цьому шрифті відповідає значок квадрат, тому він і відображується на макеті; знаку оклику відповідає павук, великій літері N – око, комі – перець чилі, і т.п.

Таблиця 3.4 – Властивості мітки label1

| № з/п | Назва властивості | Значення параметра |
|---|---|---|
| 1 | AutoSize | False |
| 2 | Dock | Fill |
| 3 | BackColor | Встановлюємо за допомогою сайту «colorpicker.me». |
| 4 | Font | Type – Webdings; Style – Bold; Size – 48. |
| 5 | Text | ‘c’ |
| 6 | TextAllign | MiddleCenter |

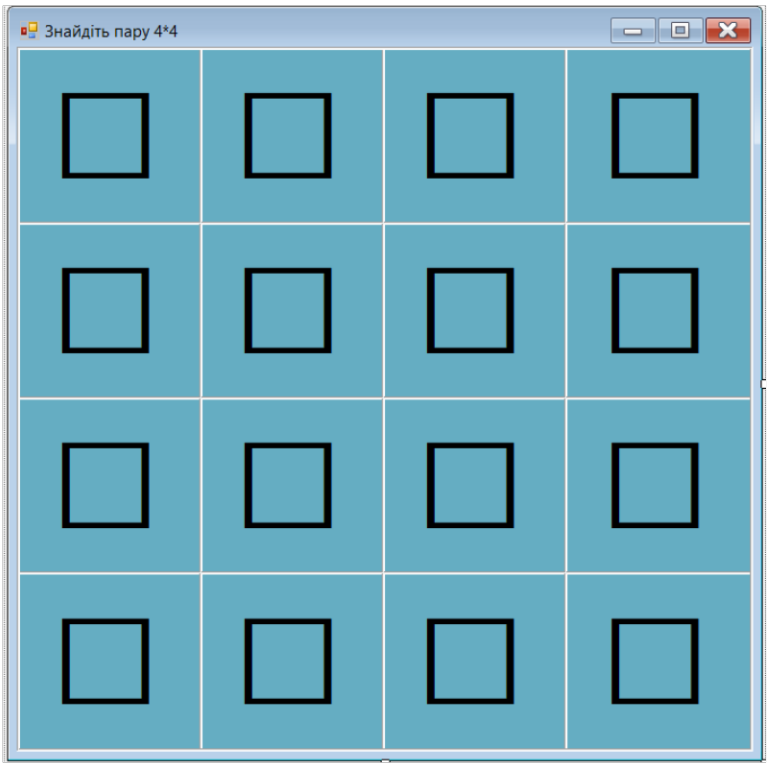


Рисунок 3.11 – Макет гри 4*4з квадратами-мітками

3.2.3 Додавання випадкового об’єкта і списку значків

Далі зосередимося на розробленні програмного коду для вирішення поставлених раніше завдань гри. Генерування випадкового об’єкта з метою подальшого вибору випадкового значка та створення списку літер шрифта Webdings. у якому кожна літера зустрічається двічі, забезпечує код, наведений на рисунку 3.12.

```
public partial class Form1 : Form
{

        Random random = new Random();

        List<string> icons = new List<string>()
        {
            "!", "!", "N", "N", ",", ",", "k", "k",
            "b", "b", "v", "v", "w", "w", "z", "z"
        };
};
```

Рисунок 3.12 – Код додавання випадкового об’єкта і списку значків

3.2.4 Призначення клітинам макету випадкових значків

Панель tableLayoutPanel1 містить 16 клітин, а список піктограм має 16 піктограм. Код функції AssignIconsToSquares(), наведений на рисунку 3.13, виконує призначення клітинам макету значення випадкових значків.

Метод AssignIconsToSquares() за допомогою інструкції foreach виконує перегляд усіх міток панелі tableLayoutPanel1. Він виконує одні й самі оператори стосовно кожної мітки. Ці оператори послідовно додають у клітини сітки значки зі списку.

Перший рядок викликає мітку з ім'ям iconLabel.

Другий рядок – це умовний оператор if. Якщо не всі мітки переглянуті, перегляд міток виконується, інакше умовний оператор if пропускається.

Перший рядок в операторі if формує змінну randomNumber, яка є випадковим числом, що відповідає одному з елементів списку значків. Метод Next() об'єкта Random повертає довільне ціле число. Властивість Count списку значків задає діапазон, з якого вибирається випадкове число.

Другий рядок присвоює властивості Text поточної мітки значення літери,

якій відповідає певна піктограма зі списку значків.

Третій рядок приховує піктограми. Цей рядок поки що нами закоментований, щоб мати змогу перевірити частину коду, що залишилася, перш ніж продовжити роботу.

Четвертий рядок в if інструкції видаляє значок, доданий у форму зі списку.

```
private void AssignIconsToSquares()
{
    foreach (Control control in tableLayoutPanel1.Controls)
    {
        Label iconLabel = control as Label;
        if (iconLabel != null)
        {
            int randomNumber = random.Next(icons.Count);
            iconLabel.Text = icons[randomNumber];
            // iconLabel.ForeColor = iconLabel.BackColor;
            icons.RemoveAt(randomNumber);
        }
    }
}
```

Рисунок 3.13 – Код призначення клітинам макету випадкових значків

Функція AssignIconsToSquares() ініціалізується під час запуску форми Form2 (рис. 3.14). Запустивши проект на виконання, отримаємо макет, заповнений випадковими значками (рис. 3.15). Маємо змогу пересвідчитися, чи вірно відображуються на панелі піктограми значків. Однак, якщо зняти коментар з третього рядка оператора if, значки з панелі зникнуть.

```
public Form1()
{
    InitializeComponent();
    AssignIconsToSquares();
}
```

Рисунок 3.14 – Код ініціалізації функції AssignIconsToSquares()


Рисунок 3.15 – Результат роботи методу AssignIconsToSquares()

3.2.5 Додавання обробників подій міток

Якщо клацнути по мітці (клітині), на екрані має з’явитися зображення значка; якщо ж клацнути по клітині, значок якої вже відображений, жодних дій не повинно відбуватися. Такий функціонал забезпечують обробники подій міток (рис. 3.16).

```
private void label1_Click(object sender, EventArgs e)
{
    Label clickedLabel = sender as Label;
    if (clickedLabel != null)
    {
        if (clickedLabel.ForeColor == Color.Black)
            return;
        clickedLabel.ForeColor = Color.Black;
    }
}
```

Рисунок 3.16 – Код додавання обробників подій міток

Наприклад, обробник клацання по першій мітці label1_Click( object sender, EventArgs e) має два аргументи: sender – ім’я мітки та e – ім’я події. Для

того, щоб надати такий функціонал усім міткам форми, натискаємо першу мітку, і, утримуючи клавішу Ctrl натиснутою, послідовно виділяємо всі мітки. Далі у вікні Властивості у категорії Події знаходимо подію Click, клацаємо по ній, і вводимо код обробника події label1_Click( object sender, EventArgs e). Тепер кожна мітка форми реагуватиме на клацання по ній у спосіб, описаний нами вище. Запустивши проект на виконання, отримаємо порожній макет. Якщо клацати по його клітинах, будуть відображатися і залишатися на екрані випадкові значки (рис. 3.17).

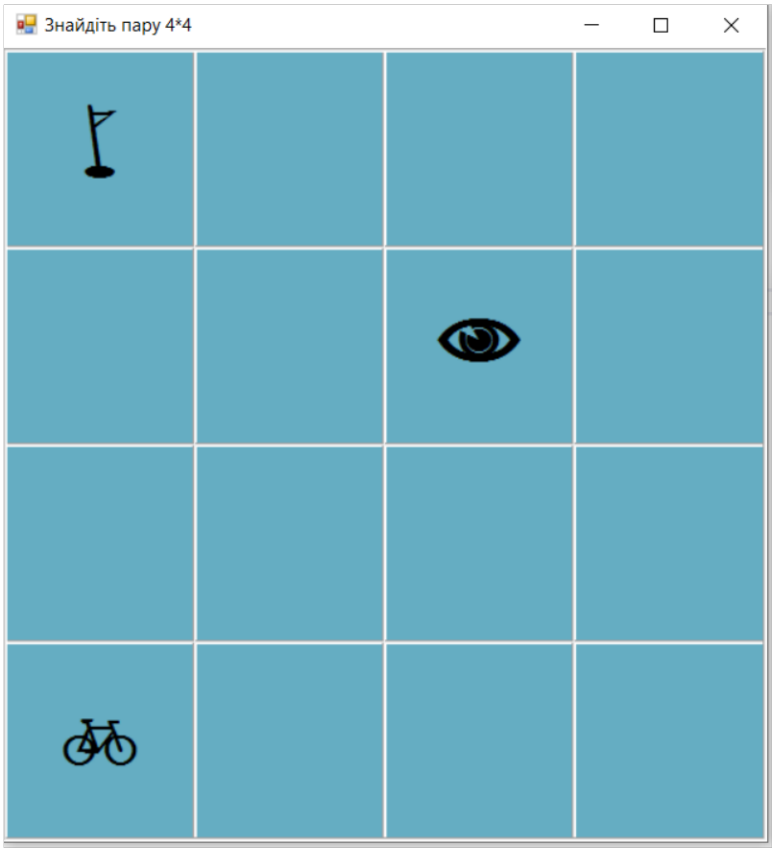


Рисунок 3.17 – Код додавання посилань на мітки клітин

### 3.2.6 Додавання посилань на мітки клітин

Код на рисунку 3.18 оголошує дві змінні-посилання. Перша змінна firstClicked вказує на першу мітку сітки. Друга змінна secondClicked вказує на другу мітку. На початку цим змінним присвоюється значення Null. З урахуванням появи цих змінних, частково модернізуємо обробник події Click label1_Click( object sender, EventArgs e) (рис. 3.19).

```
public partial class Form1 : Form
{
    Label firstClicked = null;
    Label secondClicked = null;
}
```

Рисунок 3.18 – Код додавання посилань на мітки клітин

```
private void label1_Click(object sender, EventArgs e)
{
    Label clickedLabel = sender as Label;

    if (clickedLabel != null)
    {
        if (clickedLabel.ForeColor == Color.Black)
            return;
        if (firstClicked == null)
        {
            firstClicked = clickedLabel;
            firstClicked.ForeColor = Color.Black;
            return;
        }
    }
}
```

Рисунок 3.19 – Модернізований код обробника подій label1_Click

### 3.2.7 Додавання таймера

Перетягуємо з Панелі інструментів на форму Form2 компонент таймер і створюємо об’єкт timer1 і програмуємо обробник події tick таймера (рис. 3.20). Надаємо його властивості Interval значення 750 мс. Таймер запускається в момент клацання по першому значку. Коли протягом 750 мс ми не знайшли пару, таймер вимикається, а обидва значки зникнуть з екрана.

```
private void timer1_Tick(object sender, EventArgs e)
{   timer1.Stop();
    firstClicked.ForeColor = firstClicked.BackColor;
    secondClicked.ForeColor = secondClicked.BackColor;
    firstClicked = null;
    secondClicked = null;
}
```

Рисунок 3.20 – Код додавання таймера

У зв’язку з появою таймера, відповідним чином модернізуємо обробник події Click label1_Click( object sender, EventArgs e) (рис. 3.21)

```
private void label1_Click(object sender, EventArgs e)
{
    if (timer1.Enabled == true)
        return;
    Label clickedLabel = sender as Label;
    if (clickedLabel != null)
    {
        if (clickedLabel.ForeColor == Color.Black)
            return;
        if (firstClicked == null)
        {
            firstClicked = clickedLabel;
            firstClicked.ForeColor = Color.Black;
            return;
        }
        secondClicked = clickedLabel;
        secondClicked.ForeColor = Color.Black;
        timer1.Start();
    }
}
```

Рисунок 3.21 – Код обробника подій label1_Click з урахуванням появи таймера

3.2.8 Відміна зникнення пари значків

Якщо під час гри гравець знайшов пару, тобто другий значок виявився таким самим , як перший, а час не перевищує 750 мс, то зображення обох значків треба залишити на екрані. Цього досягнемо за рахунок змін у коді обробника подій label1_Click, наведених на рисунку 3.22.

```
secondClicked = clickedLabel;
secondClicked.ForeColor = Color.Black;
if (firstClicked.Text == secondClicked.Text)
{
    firstClicked = null;
    secondClicked = null;
    return
}
timer1.Start();
```

Рисунок 3.22 – Код відміни щезання пари значків

3.2.9 Перевірка факту виграшу чи програшу

Якщо під час гри гравець знайшов пари усім значкам, то гравець виграв. Перевірка цього факту здійснюється функцією CheckForWinner(), код якої наведено на рисунку 3.23.

```
private void CheckForWinner()
{
    foreach (Control control in tableLayoutPanel1.Controls)
    {
        Label iconLabel = control as Label;
        if (iconLabel != null)
        {
            if (iconLabel.ForeColor == iconLabel.BackColor)
                return;
        }
    }
    MessageBox.Show("Вітаємо! Ви знайшли всі пари!");
    this.Hide();
    Form3 f3 = new Form3();
    f3.Show();
}
```

Рисунок 3.23 – Код перевірки факту виграшу чи програшу

Остаточний варіант коду обробника події Click клацання по мітці label1_Click(), з урахуванням змін, наведених на рис 3.22, має вигляд, зображений на рисунку 3.24.

Запустивши гру, бачимо, що вона функціонує цілком коректно. Про це свідчать і проміжні результати гри, зображені на рисунку 3.25.

Таким чином, нами розроблена цікава і корисна гра «Знайдіть пару 4*4», використовуючи яку можна тренувати увагу. Така програма може бути корисна для дітей дошкільного віку, школярів та інших категорій населення. Як і більшість відомих ігор, цю гру можна зробити багатоступеневою за принципом – від простішого до складнішого. Тому на наступній формі нами розроблена складніша версія гри «Знайдіть пару 4*5» на сітці, що складається з чотирьох стрічок та п’яти стовпчиків.

```
private void label1_Click(object sender, EventArgs e)
{
   if (timer1.Enabled == true)
       return;
   Label clickedLabel = sender as Label;
   if (clickedLabel != null)
   {
      if (clickedLabel.ForeColor == Color.Black)
          return;
      if (firstClicked == null)
      {
          firstClicked = clickedLabel;
          firstClicked.ForeColor = Color.Black;
          return;
      }
      secondClicked = clickedLabel;
      secondClicked.ForeColor = Color.Black;
      CheckForWinner();
      if (firstClicked.Text == secondClicked.Text)
      {
         firstClicked = null;
         secondClicked = null;
         return;
      }
      timer1.Start();
    }
}
```

Рисунок 3.24 – Остаточний вигляд обробника події label1_Click

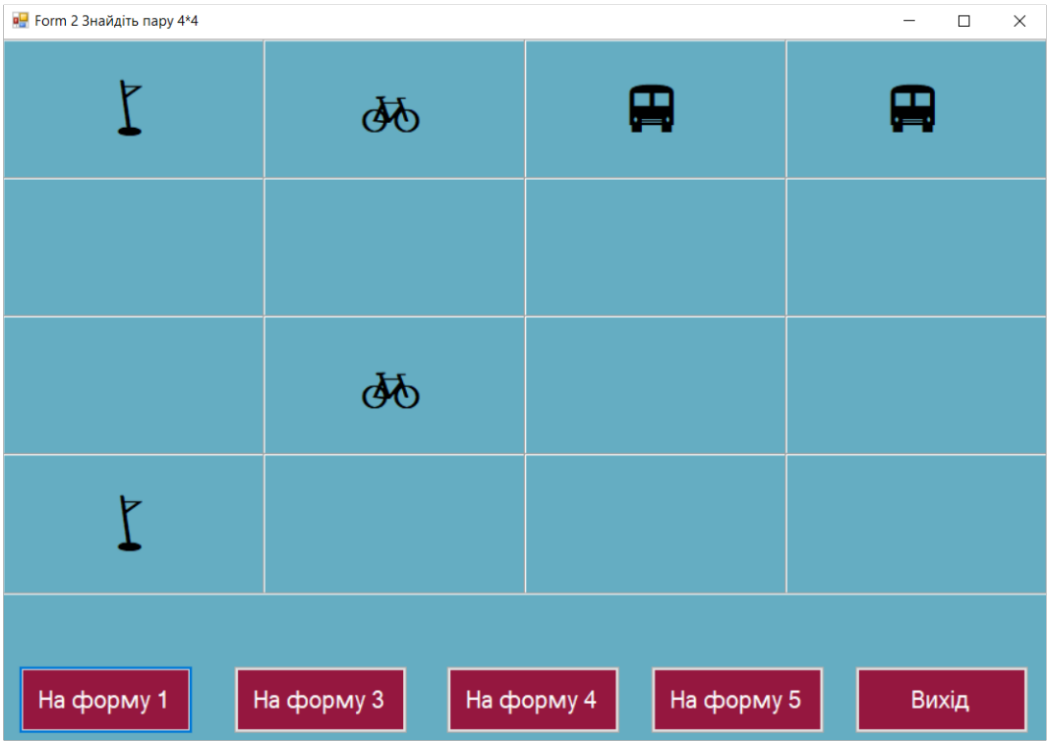


Рисунок 3.25 – Фрагмент гри «Знайдіть пару 4*4»

### 3.3 Додаток-гра «Знайдіть пару 4*5» для тренування уваги

Додаток-гра «Знайдіть пару 4*5» (рис.3.26) є складнішою за гру «Знайдіть пару 4*4», оскільки її макет містить 20 клітин, а не 16. Запам’ятати, у якій з цих клітин відображувався раніше той чи інший значок, складніше. Але саме це нам і потрібно, адже наше завдання – зробити гру складнішою.

Додаток-гра «Знайдіть пару 4*5» розроблена за тією ж технологією, що й попередня гра, тому немає сенсу описувати цей процес ще раз. Натомість у додатку А наведено код гри, який враховує ті незначні відмінності, які з’являються під час ускладнення гри. Зокрема, список символів icons, який створюється інструкцією List <string> icons = new List<string(), містить вже не 16 літер, як раніше, а 20, що відповідає числу клітин сітки. До попереднього списку символів додані нові символи ‘’, ‘’, ‘’, ‘’.

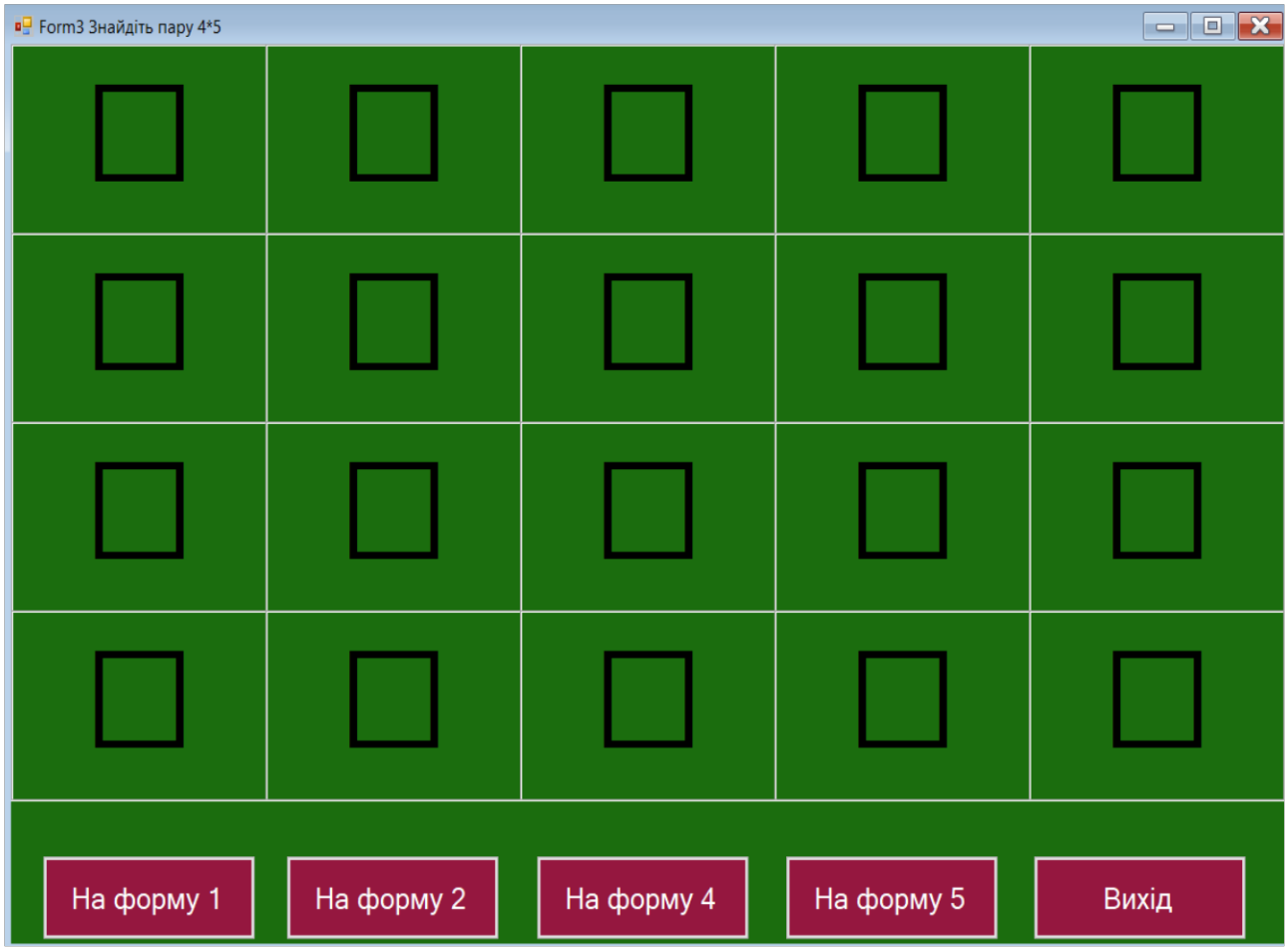


Рисунок 3.26 – Вигляд форми Form гри «Знайдіть пару 4*5»

### 3.4 Додаток-гра «Математичний тест» для перевірки знань з арифметики

Призначення додатку-гри «Математичний тест» – перевірити, чи правильно виконує гравець арифметичні дії (додавання, віднімання, множення, ділення). Вигляд форми Form4 гри «Математичний тест» наведено – на рисунку 3.27, склад елементів гри – у таблиці 3.5, а код гри – на рисунках 3.28 – 3.30.


Рисунок 3.27 – Вигляд форми Form4 гри «Математичний тест»

Ідея гри полягає у тому, що кнопкою Почати тестування гравець запускає гру і одночасно запускає таймер, значення властивості Інтервал якого встановлюємо рівним 1000 мс. Комп'ютер генерує випадкові значення операндів арифметичних операцій. Гравець має ввести правильні відповіді у вікна відповідних елементів форми. Якщо всі відповіді вірні, гравець виграв. Якщо хоча б одна відповідь хибна або час виконання арифметичних дій перевищив 1000 мс, гравець програв.

Таблиця 3.5 – Склад елементів гри «Математичний тест»

| № з/п | Ім’я елемента або групи елементів | Категорія елементів | Призначення елементів |
|---|---|---|---|
| 1 | button1 – button4 | Button | Перехід на інші форми проекту |
| 2 | startButton1 | Button | Запуск гри на виконання |
| 3 | label1 | Label | Відображення надпису «Залишилося часу» |
| 4 | timeLabel | Label | Відображення значення часу, щоо залишився |
| 5 | label3 | Label | Відображення знака «+» |
| 6 | label8 | Label | Відображення знака «–» |
| 7 | Label12 | Label | Відображення знака «*» |
| 7 | label16 | Label | Відображення знака «/» |
| 8 | label5, label6, label10, label4 | Label | Відображення знака «=» |
| 9 | plusLeftLabel | Label | Відображення значення лівого операнда операції додавання |
| 10 | minusLeftLabel | Label | Відображення значення лівого операнда операції віднімання |
| 11 | timesLeftLabel | Label | Відображення значення лівого операнда операції множення |
| 12 | dividedLeftLabel | Label | Відображення значення лівого операнда операції ділення |
| 13 | plusRightLabel | Label | Відображення значення правого операнда операції додавання |
| 14 | minusRightLabel | Label | Відображення значення правого операнда операції віднімання |
| 15 | timesRightLabel | Label | Відображення значення правого операнда операції множення |
| 16 | dividedRightLabel | Label | Відображення значення правого операнда операції ділення |
| 17 | sum | NumericUpDown | Результат операції додавання |
| 18 | diffrence | NumericUpDown | Результат операції віднімання |
| 19 | product | NumericUpDown | Результат операції множення |
| 20 | quotient | NumericUpDown | Результат операції ділення |
| 21 | timer1 | Timer | Таймер |

```
{
    // Генерування випадкового цілого числа
    Random randomizer = new Random();

    // Оголошення двох змінних – опрандів операції додавання
    int addend1;
    int addend2;

    // Оголошення двох змінних – опрандів операції віднімання
    int minuend;
    int subtrahend;

    // Оголошення двох змінних – опрандів операції множення
    int multiplicand;
    int multiplier;

    // Оголошення двох змінних – опрандів операції ділення
    int dividend;
    int divisor;

    // Оголошення змінної для значення часу, що залишився
    int timeLeft;

    /// Запуск гри на виконання
    public void StartTheQuiz()
    {
        // Генерування та збереження значень опрандів операції додавання
        addend1 = randomizer.Next(51);
        addend2 = randomizer.Next(51);

        // Конвертація значень опрандів операції додавання у стрічковий тип
        plusLeftLabel.Text = addend1.ToString();
        plusRightLabel.Text = addend2.ToString();

        // Присвоєння сумі початкового значення 0.
        sum.Value = 0;
       // Реалізація дій стосовно операції віднімання.
       minuend = randomizer.Next(1, 101);
       subtrahend = randomizer.Next(1, minuend);
       minusLeftLabel.Text = minuend.ToString();
       minusRightLabel.Text = subtrahend.ToString();
       difference.Value = 0;
       // Реалізація дій стосовно операції множення.
       multiplicand = randomizer.Next(2, 11);
       multiplier = randomizer.Next(2, 11);
       timesLeftLabel.Text = multiplicand.ToString();
       timesRightLabel.Text = multiplier.ToString();
       product.Value = 0;
       // Реалізація дій стосовно операції ділення.
       divisor = randomizer.Next(2, 11);
       int temporaryQuotient = randomizer.Next(2, 11);
       dividend = divisor * temporaryQuotient;
       dividedLeftLabel.Text = dividend.ToString();
       dividedRightLabel.Text = divisor.ToString();
       quotient.Value = 0;
```

Рисунок 3.28 – Код гри «Математичний тест»

```
        // Запуск таймера.
        timeLeft = 60;
        timeLabel.Text = "60 секунд";
        timer1.Start();
    }
    // Повертає значення True якщо відповідь правильна, інакше false
    private bool CheckTheAnswer()
    {
        if ((addend1 + addend2 == sum.Value)
        && (minuend - subtrahend == difference.Value)
        && (multiplicand * multiplier == product.Value)
        && (dividend / divisor == quotient.Value))
            return true;
        else
            return false;
    }
    public Form4()
    {
        InitializeComponent();
    }
    private void button1_Click(object sender, EventArgs e)
    {
        this.Hide();
        Form1 f1 = new Form1();
        f1.Show();
    }
    private void button4_Click(object sender, EventArgs e)
    {
        this.Hide();
        Form3 f3 = new Form3();
        f3.Show();
    }
    private void button3_Click(object sender, EventArgs e)
    {
        this.Hide();
        Form2 f2 = new Form2();
        f2.Show();
    }

    private void button5_Click(object sender, EventArgs e)
    {
        this.Hide();
        Form5 f5 = new Form5();
        f5.Show();
    }
    private void button2_Click(object sender, EventArgs e)
    {
        Application.Exit();
    }
```

Рисунок 3.29 – Код гри «Математичний тест»

```
    private void timer1_Tick(object sender, EventArgs e)
    {
        if (CheckTheAnswer())
        {
            // Якщо гравець правильно виконав усі дії
            timer1.Stop();
            MessageBox.Show("Вітаємо! Ви вчасно дали правильні відповіді!
        Переходимо до наступної форми!");
            startButton.Enabled = true;

            this.Hide();
            Form5 f5 = new Form5();
            f5.Show();
        }
        else if (timeLeft > 0)
        {
            // Якщо час гри ще не закінчився
            timeLeft = timeLeft - 1;
            timeLabel.Text = timeLeft + " секунд";
        }
        else
        {
            // Якщо час гри закінчився.
            timer1.Stop();
            timeLabel.Text = "Час закінчився!";
            MessageBox.Show("Жаль. Ви не впоралися вчасно!");
            sum.Value = addend1 + addend2;
            difference.Value = minuend - subtrahend;
            product.Value = multiplicand * multiplier;
            quotient.Value = dividend / divisor;
            startButton.Enabled = true;

        }
    }
     private void startButton_Click(object sender, EventArgs e)
     {
         StartTheQuiz();
         startButton.Enabled = false;
     }
}
```

Рисунок 3.30 – Код гри «Математичний тест»

З наведеного коду видно, що функція startButton_Click(object sender, EventArgs e) запускає гру на виконання, функція StartTheQuiz() виконує всі арифметичні операції, а функція CheckTheAnswer() перевіряє правильність виконання цих операцій. Результати тестування додатку-гри засвідчили, що вона працює коректно.

### 3.5 Тест на базі компонента RadioButton

Призначення тесту, розробленого на базі компоненти Radiobutton – вибрати правильну відповідь із кількох запропонованих, за умови, що лише одна відповідь правильна. Вигляд форми Form6 тесту наведено – на рисунку 3.31, склад елементів гри – у таблиці 3.6, а код гри – на рисунках 3.32-3.33.

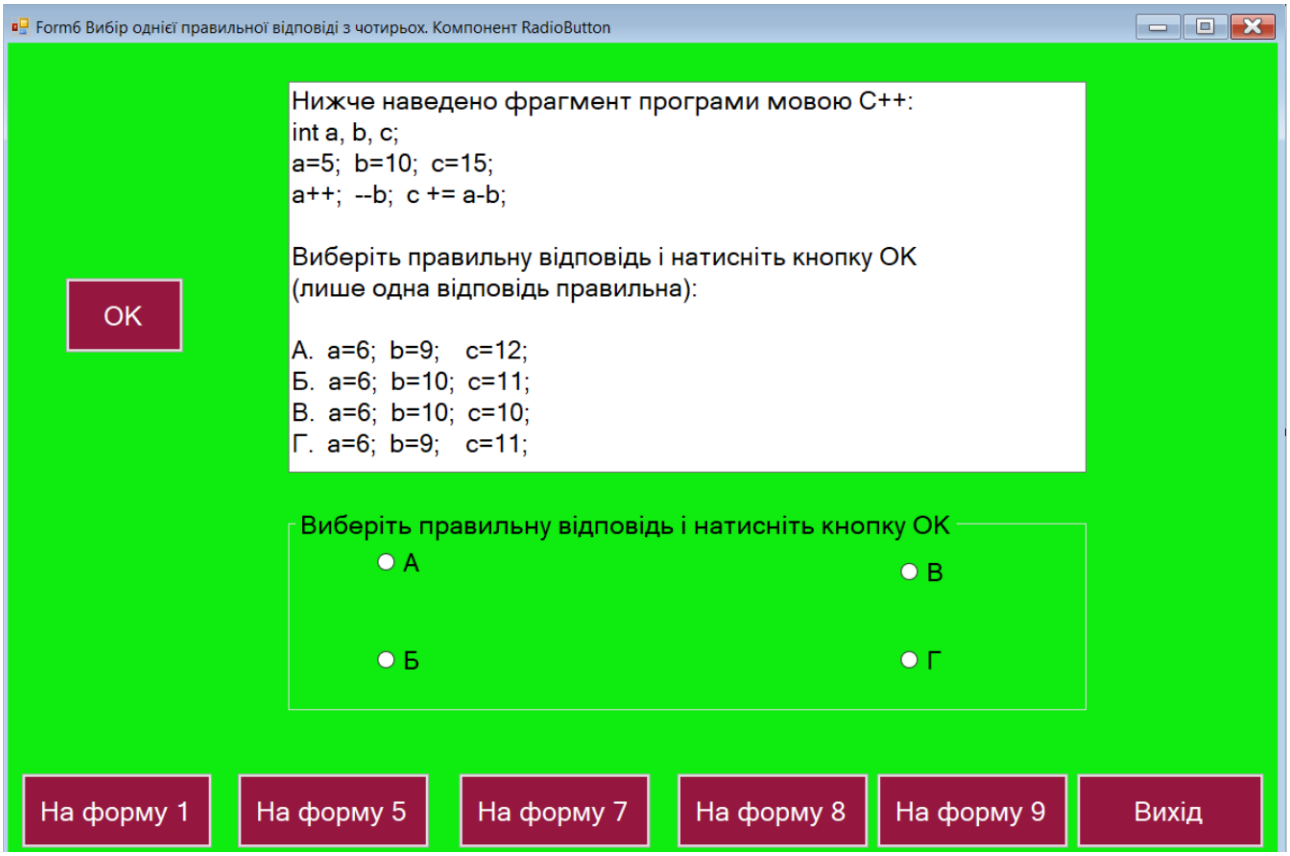


Рисунок 3.31 – Вигляд форми Form6 тесту на базі компонента RadioButton

Таблиця 3.6 – Склад елементів тесту на базі компоненти RadioButton

| № з/п | Ім'я елемента або групи елементів | Категорія елементів | Призначення елементів |
|---|---|---|---|
| 1 | button1 – button6 | Button | Перехід на інші форми проекту |
| 2 | button7 | Button | Підтвердження вибору (OK) |
| 3 | radioButton1 radiobutton4 | RadioButton | Вибір однієї правильної відповіді |
| 4 | groupBox1 | GroupBox | Оформлення групи елементіва |

```
public Form6()
{
    InitializeComponent();
}
private void button1_Click(object sender, EventArgs e)
{
    this.Hide();
    Form1 f1 = new Form1();
    f1.Show();
}
private void button3_Click(object sender, EventArgs e)
{
    this.Hide();
    Form5 f5 = new Form5();
    f5.Show();
}
private void button4_Click(object sender, EventArgs e)
{
    this.Hide();
    Form7 f7 = new Form7();
    f7.Show();
}
private void button5_Click(object sender, EventArgs e)
{
    this.Hide();
    Form8 f8 = new Form8();
    f8.Show();
}
private void button6_Click(object sender, EventArgs e)
{
    this.Hide();
    Form9 f9 = new Form9();
    f9.Show();
}
```

```
private void button7_Click(object sender, EventArgs e)
{
    if (radioButton1.Checked)
    {
        MessageBox.Show("Ви відповіли правильно!", "Тест",
                        MessageBoxButtons.OK, MessageBoxIcon.Information);
       radioButton1.Enabled = false; radioButton2.Enabled = false;
          radioButton3.Enabled = false; radioButton4.Enabled = false;
    }
    if (radioButton2.Checked)
    {
        MessageBox.Show("Ви відповіли неправильно! Переходимо до
          наступного питання!", "Тест", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
        radioButton1.Enabled = false; radioButton2.Enabled = false;
         radioButton3.Enabled = false; radioButton4.Enabled = false;
    }
```

Рисунок 3.31 – Код тесту на базі компонента RadioButton

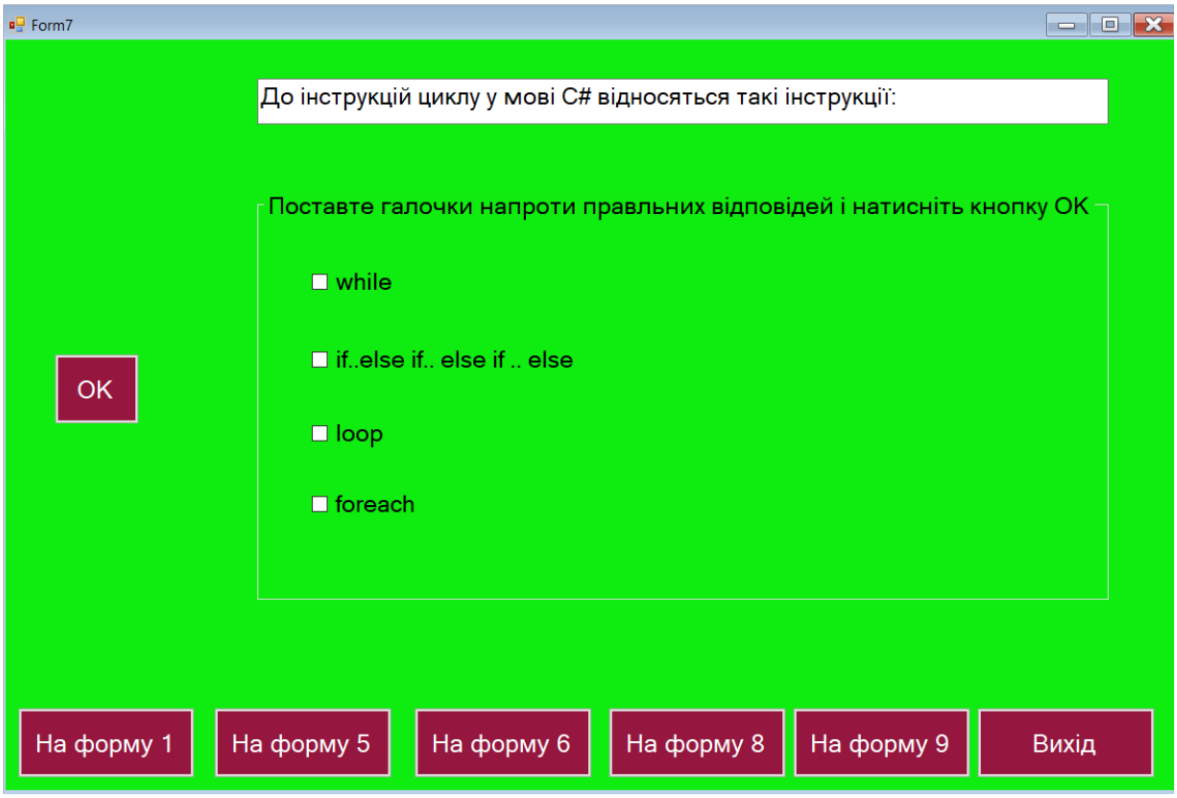


Рисунок 3.33 – Вигляд форми Form7 тесту на базі компонента CheckBox

Таблиця 3.7 – Склад елементів тесту на базі компонента CheckBox

| № з/п | Ім'я елемента або групи елементів | Категорія елементів | Призначення елементів |
|---|---|---|---|
| 1 | button1 – button6 | Button | Перехід на інші форми проекту |
| 2 | button7 | Button | Підтвердження вибору (OK) |
| 3 | checkBox1 – checkBox4 | CheckBox | Вибір правильної відповіді (або кількох правильних відповідей) |
| 4 | GroupBox 1 | GroupBox | Оформлення групи елементіва |
| 5 | textBox1 | TextBox | Пояснювальний текст (умова завдання)) |

### 3.7 Тест на базі компонента ListBox

Призначення тесту, розробленого на базі компонент ListBox – для кожного елемента першої множини вибрати відповідний елемент у другій

множині (інакше кажучи, знайти правильні пари, формуючи їх з елементів двох множин). Вигляд форми Form8 тесту наведено – на рисунку 3.34, склад елементів гри – у таблиці 3.8, а код гри – у додатку В.

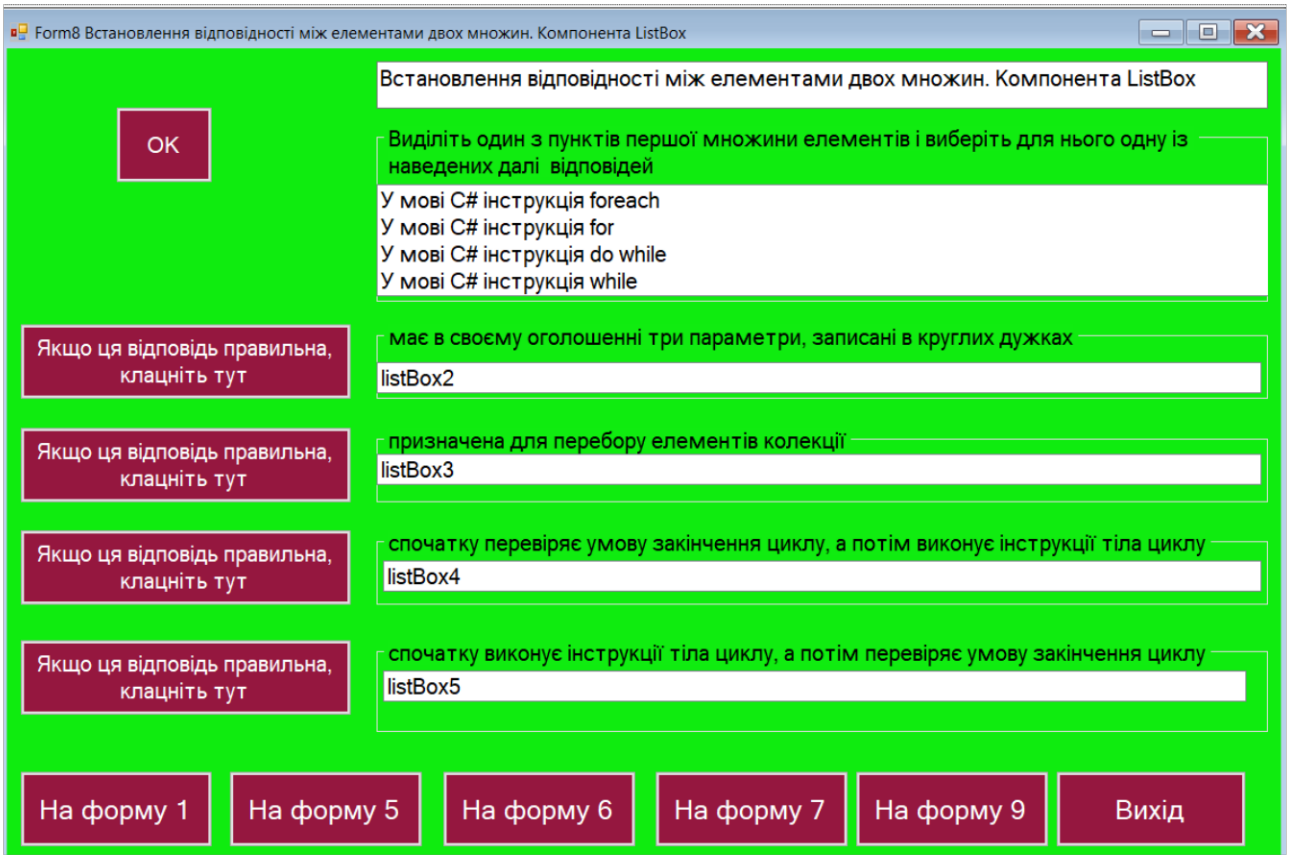


Рисунок 3.34 – Вигляд форми Form8 тесту на базі компонента ListBox

Таблиця 3.8 – Склад елементів тесту на базі компонента ListBox

| № з/п | Ім'я елемента або групи елементів | Категорія елементів | Призначення елементів |
|---|---|---|---|
| 1 | button1 – button6 | Button | Перехід на інші форми проекту |
| 2 | button7 | Button | Підтвердження вибору (OK) |
| 3 | listBox1 – listBox5 | ListBox | Розміщення елементів множин |
| 4 | groupBox1 – groupBox 5 | GroupBox | Оформлення груп елементіва |
| 5 | textBox1 | TextBox | Пояснювальний текст (умова завдання)) |
| 6 | button8 – button11 | Button | Підтвердження вибору |

### 3.8 Тест на базі компонента ComboBox

Призначення тесту, розробленого на базі компонента ComboBox – вибрати для кожного питання екзаменаційного білета правильну відповідь зі списку запропонованих. Вигляд форми Form9 тесту наведено – на рисунку 3.35, склад елементів гри – у таблиці 3.9, а код гри – у додатку Г.

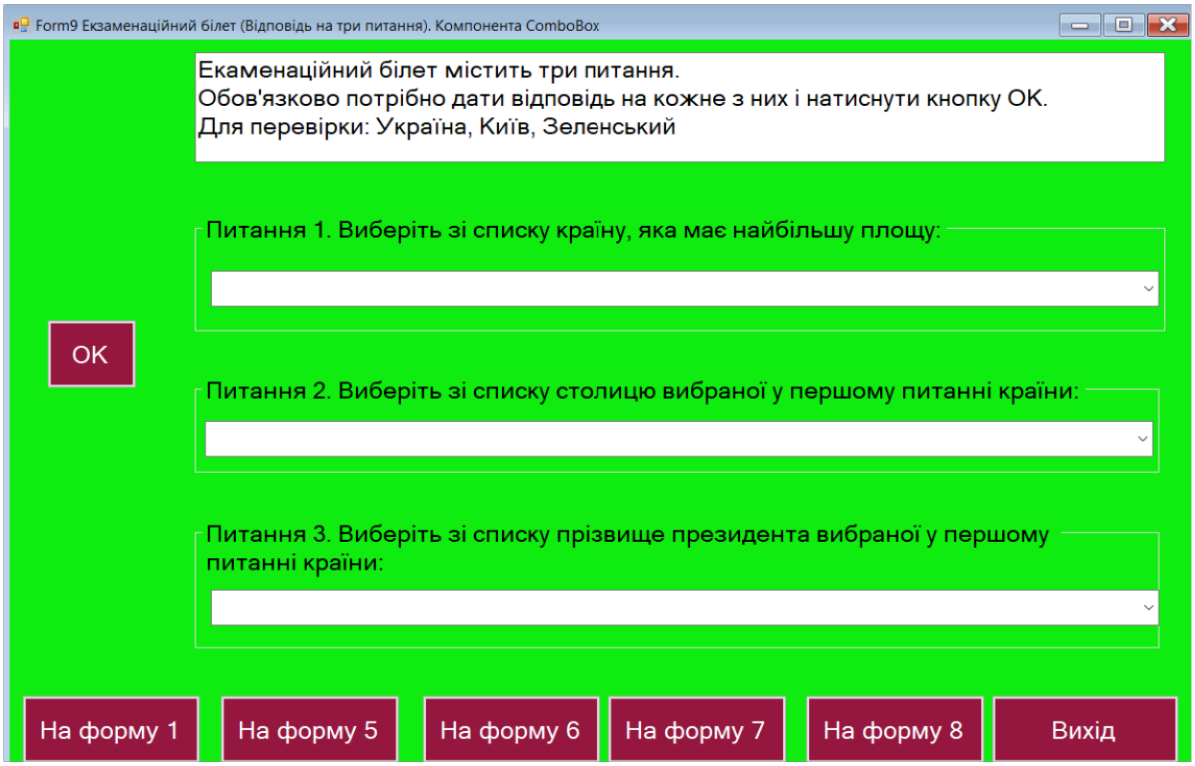


Рисунок 3.35 – Вигляд форми Form9 тесту на базі компонента ComboBox

Таблиця 3.9 – Склад елементів тесту на базі компонента ComboBox

| № з/п | Ім'я елемента або групи елементів | Категорія елементів | Призначення елементів |
|---|---|---|---|
| 1 | button1 – button6 | Button | Перехід на інші форми проекту |
| 2 | button7 | Button | Підтвердження вибору (OK) |
| 3 | comboBox1 – comboBox3 | ComboBox | Розміщення списків з відповідями |
| 4 | groupBox1 – groupBox 3 | GroupBox | Оформлення груп елементіва |
| 5 | textBox1 | TextBox | Пояснювальний текст (умова завдання)) |

# ВИСНОВКИ

Проведені у кваліфікаційній роботі бакалавра дослідження та розроблений у ній програмний проект підтвердили актуальність розв'язання сформульованих у роботі завдань. Встановлено, що тренування уваги та перевірка знань за допомогою програмних засобів є на теперішній час актуальним завданням.

У кваліфікаційній роботі бакалавра:

1. Виконано огляд програмних засобів для розробки ігор та тестів для тренування уваги та перевірки знань. Дано детальну характеристику інструментів для розробки програмного комплексу засобами C# WinForms.
2. Розроблено засобами C# WinForms проект у складі семи програм.
3. Розроблено програму-гру «Знайдіть пару» на сітці розмірами 4*4.
4. Розроблено програму-гру «Знайдіть пару» на сітці розмірами 4*5.
5. Розроблено програму-гру «Математичний тест» для перевірки правильності виконання арифметичних операцій.
6. Розроблено програму-тест на базі компонентів RadioButton для вибору однієї правильної відповіді з кількох.
7. Розроблено програму-тест на базі компонентів CheckBox для вибору більше однієї правильної відповіді з кількох.
8. Розроблено програму-тест на базі компонентів ListBox для встановлення відповідності між двома множинами тверджень.
9. Розроблено програму-тест на базі компонентів ComboBox для вибору правильних відповідей екзаменаційного білета.

Усі програми належним чином розроблені, відлагоджені та протестовані, в результаті чого було підтверджено факт їх коректної роботи.

Розроблений у кваліфікаційній роботі проект (комплекс програм) може бути використаний для тренування уваги дітей та перевірки знань студентів у навчальному процесі.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Пех П.А. Програмування: Конспект лекцій для здобувачів першого (бакалаврського) рівня освітньо-професійної програми «Комп'ютерна інженерія» галузі знань 12 «Інформаційні технології» спеціальності 123 «Комп'ютерна інженерія» денної та заочної форм навчання. Луцьк: ЛНТУ, 2020. 128 с.

2. OhNxuB. Рейтинг мов програмування-2024: з якими мовами працюють та що планують вчити айтівці різних напрямків. URL: https://dev.ua/news/reitynh-mov-1708330900 (дата звернення: 30.03.2024).

3. Що таке .NET? – Про Програмування. URL: https://abitap.com/1-1/ (дата звернення: 30.03.2024).

4. Getting Started. Language selection | European Commission. URL: https://ec.europa.eu/programmes/erasmus-plus/project-result-content/28f2f10e-27c9-4223-af95-19a4445a41a1/prohramy_2018_Osnovy_kompiuternykh_ihor.pdf (дата звернення: 30.03.2024).

5. Clickteam – Home. A Quick Intro to Physics in ClickTeam Fusion 2.5. URL: http://download.clickteam.com/tutorials/en/quick–physics.pdf (дата звернення: 30.03.2024).

6. Крег Стіл Берат. Геймер на 100%. Переходь у режим профі. Київ: Ранок, 2021. 96 с.

7. Джонатан Геннесі, Джек МакҐовен. Історія відеоігор в коміксах. Київ: Yakaboo Publishing, 2020. 192 с.

8. Шраєр Джейсон, Кров, піт і пікселі. Тріумфальні та бурхливі історії по той бік створення відеоігор. Київ: BookChef, 2020. 336 с.

9. Річард Вільямс. Анімація: Посібник з виживання. Київ: ArtHuss, 2019. 392 с.

10. Epic Games. FORTNITE Official. Як малювати. Київ: Artbooks, 2020. 102 с.

11. Раф Костер. Теорія розваг для ігрового дизайну. Київ: ArtHuss, 2023. 288 с.
12. Джейн Макгонігал. Реальність під питанням. Чому ігри роблять нас кращими і як вони можуть змінити світ. Київ: BookChef, 2019. 384 с.
13. John Staats. The WOW Diary: a jornal of computer games. Київ: BookChef, 2023. 336 с.
14. Террі Вулф. Кодзіма – геній. Історія розробника, перевернув індустрію відеоігор. Київ: Форс, 2019. 600 с.
15. Девід Шефф. Game over! Як Nintendo завоював світ. Київ: Біле яблуко, 2023. 384 с.
16. Девід Кушнер. Повелителі DOOM. Як два хлопця створили культовий шутер і розгойдали індустрію відеоігор. Київ: Форс, 2020. 496 с.
17. Трістан Донаван. Грай! Історія відеоігор. Київ: ArtHuss, 2020. 630 с.
18. Система електронного навчання інституту інформатики. Режим доступу. URL: http://www.moodle.ii.npu.edu.ua (дата звернення: 30.03.2024).
19. MoodleDocs. URL: http://docs.moodle.org (дата звернення: 30.03.2024).
20. Персональний сайт Франчука В.М. URL: http://www.vfranchuk.npu.edu.ua (дата звернення: 30.03.2024).
21. USAID. Проект «Справедливе правосуддя». Практичний посібник для розробників тестових завдань. URL: https://newjustice.org.ua/wp-content/uploads/2018/05/Manual_for_test_writers.pdf (дата звернення: 30.03.2024).
22. 7 сервісів для створення навчальних тестів та завдань онлайн. Букі | Buki – ваш репетитор з будь-якого предмету. Репетитори України. URL: https://buki.com.ua/news/7-servisiv-dlya-stvorennya-navchalnykh-testiv-ta-zavdan-onlayn (дата звернення: 30.03.2024).
23. Genesis. URL: https://www.gen.tech/post/yak-obrati-knigu-po-c-10-rekomendacij-dlya-rozrobnikiv-riznih-rivniv (дата звернення: 30.03.2024).
24. Ендрю Троелсен Філіп Джепікс. Мова програмування C#7 і платформи .NET і .NET Core. Київ: Діалектика, 2020. 1000 с.

25. Ерік Фрімен, Елізабет Робсон, Берт Бейтс, Кеті Сієрра. Head First. Патерни проєктування. Харків: Фабула, 2020. 672 с.
26. Роберт Мартін. Чистий код. Харків: Фабула, 2019. 416 с.
27. Ріхтер Джефрі. CLR via C#. Програмування на платформі Microsoft .NET Framework 4.5 на мові C#. Київ: Print2Print, 2020. 896 с.
28. Джон Скіт. C# in Depth. Київ: Manning, 2019. 528 с.
29. Джон Вліссідес Еріх Гамма. Патерни об'єктно-орієнтованого проектування. Київ: Print2Print, 2020. 448 с.
30. Eric Evans. Domain-Driven Design: Tackling Complexity in the Heart of Software. London : PublishingHouse, 2020. 560 с.

# ДОДАТКИ

# Додаток А

## Код гри «Знайдіть пару 4*5»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DiplomaWork
{
    public partial class Form3 : Form
    {
        Label firstClicked = null;
        Label secondClicked = null;

        Random random = new Random();

        List<string> icons = new List<string>()
        {
        "!", "!", "N", "N", ",", ",", "k", "k", "q", "q",
        "b", "b", "v", "v", "w", "w", "z", "z", "p", "p"
        };

        private void AssignIconsToSquares()
        {
            foreach (Control control in tableLayoutPanel1.Controls)
            {
                Label iconLabel = control as Label;
                if (iconLabel != null)
                {
                    int randomNumber = random.Next(icons.Count);
                    iconLabel.Text = icons[randomNumber];
                    iconLabel.ForeColor = iconLabel.BackColor;
                    icons.RemoveAt(randomNumber);
                }
            }
        }

        public Form3()
        {
            InitializeComponent();
            AssignIconsToSquares();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form1 f1 = new Form1();
            f1.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form2 f2 = new Form2();
            f2.Show();
        }
```

```
private void button4_Click(object sender, EventArgs e)
{
    this.Hide();
    Form4 f4 = new Form4();
    f4.Show();
}

private void button5_Click(object sender, EventArgs e)
{
    this.Hide();
    Form5 f5 = new Form5();
    f5.Show();
}

private void button2_Click(object sender, EventArgs e)
{
    Application.Exit();
}

private void label1_Click(object sender, EventArgs e)
{
    if (timer1.Enabled == true)
        return;

    Label clickedLabel = sender as Label;

    if (clickedLabel != null)
    {
        if (clickedLabel.ForeColor == Color.Black)
            return;

        if (firstClicked == null)
        {
            firstClicked = clickedLabel;
            firstClicked.ForeColor = Color.Black;
            return;
        }

        secondClicked = clickedLabel;
        secondClicked.ForeColor = Color.Black;

        CheckForWinner();

        if (firstClicked.Text == secondClicked.Text)
        {
            firstClicked = null;
            secondClicked = null;
            return;
        }

        timer1.Start();
    }
}

private void timer1_Tick(object sender, EventArgs e)
{
    timer1.Stop();

    firstClicked.ForeColor = firstClicked.BackColor;
    secondClicked.ForeColor = secondClicked.BackColor;

    firstClicked = null;
    secondClicked = null;
}
```

```
        private void CheckForWinner()
        {
            foreach (Control control in tableLayoutPanel1.Controls)
            {
                Label iconLabel = control as Label;

                if (iconLabel != null)
                {
                    if (iconLabel.ForeColor == iconLabel.BackColor)
                        return;
                }
            }

            MessageBox.Show("Вітаємо! Ви знайшли всі пари! Переходимо до наступної
форми!");

            this.Hide();
            Form4 f4 = new Form4();
            f4.Show();
        }

    }
}
```

# Додаток Б

## Код програми-тесту на базі компонентf CheckBox

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement.Button;

namespace DiplomaWork
{
    public partial class Form7 : Form
    {
        public Form7()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form1 f1 = new Form1();
            f1.Show();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form6 f6 = new Form6();
            f6.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form5 f5 = new Form5();
            f5.Show();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form8 f8 = new Form8();
            f8.Show();
        }

        private void button6_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form9 f9 = new Form9();
            f9.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
```

```
        private void button7_Click(object sender, EventArgs e)
        {
            if ((!checkBox1.Checked) && (!checkBox2.Checked) && (!checkBox3.Checked)
&& (!checkBox4.Checked))
                MessageBox.Show("Потрібно обов'язково дати відповідь!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            else
             if ((checkBox1.Checked) && (!checkBox2.Checked) && (!checkBox3.Checked)
&& (checkBox4.Checked))
            {
                MessageBox.Show("Ви відповіли правильно!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                MessageBox.Show("Переходимо до наступного питання - натисніть кнопку
На форму 8!", "Тест", MessageBoxButtons.OK, MessageBoxIcon.Information);
                checkBox1.Enabled = false; checkBox2.Enabled = false;
checkBox3.Enabled = false; checkBox4.Enabled = false;
            }
            else
            {
                MessageBox.Show("Ви відповіли неправильно!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                MessageBox.Show("Переходимо до наступного питання - натисніть кнопку
На форму 8!", "Тест", MessageBoxButtons.OK, MessageBoxIcon.Information);
                checkBox1.Enabled = false; checkBox2.Enabled = false;
checkBox3.Enabled = false; checkBox4.Enabled = false;
            }

        }
    }
}
```

# Додаток В

## Код програми-тесту на базі компонентf ListBox

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement.Button;

namespace DiplomaWork
{
    public partial class Form7 : Form
    {
        public Form7()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form1 f1 = new Form1();
            f1.Show();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form6 f6 = new Form6();
            f6.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form5 f5 = new Form5();
            f5.Show();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form8 f8 = new Form8();
            f8.Show();
        }

        private void button6_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form9 f9 = new Form9();
            f9.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
```

```
        private void button7_Click(object sender, EventArgs e)
        {
            if ((!checkBox1.Checked) && (!checkBox2.Checked) && (!checkBox3.Checked)
&& (!checkBox4.Checked))
                MessageBox.Show("Потрібно обов'язково дати відповідь!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            else
             if ((checkBox1.Checked) && (!checkBox2.Checked) && (!checkBox3.Checked)
&& (checkBox4.Checked))
            {
                MessageBox.Show("Ви відповіли правильно!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                MessageBox.Show("Переходимо до наступного питання – натисніть кнопку
На форму 8!", "Тест", MessageBoxButtons.OK, MessageBoxIcon.Information);
                checkBox1.Enabled = false; checkBox2.Enabled = false;
checkBox3.Enabled = false; checkBox4.Enabled = false;
            }
            else
            {
                MessageBox.Show("Ви відповіли неправильно!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                MessageBox.Show("Переходимо до наступного питання – натисніть кнопку
На форму 8!", "Тест", MessageBoxButtons.OK, MessageBoxIcon.Information);
                checkBox1.Enabled = false; checkBox2.Enabled = false;
checkBox3.Enabled = false; checkBox4.Enabled = false;
            }

        }
    }
}
```

# Додаток Г

## Код програми-тесту на базі компонентf ComboBox

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement.Button;

namespace DiplomaWork
{
    public partial class Form7 : Form
    {
        public Form7()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form1 f1 = new Form1();
            f1.Show();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form6 f6 = new Form6();
            f6.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form5 f5 = new Form5();
            f5.Show();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form8 f8 = new Form8();
            f8.Show();
        }

        private void button6_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form9 f9 = new Form9();
            f9.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
```

```
        private void button7_Click(object sender, EventArgs e)
        {
            if ((!checkBox1.Checked) && (!checkBox2.Checked) && (!checkBox3.Checked)
&& (!checkBox4.Checked))
                MessageBox.Show("Потрібно обов'язково дати відповідь!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            else
             if ((checkBox1.Checked) && (!checkBox2.Checked) && (!checkBox3.Checked)
&& (checkBox4.Checked))
            {
                MessageBox.Show("Ви відповіли правильно!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                MessageBox.Show("Переходимо до наступного питання – натисніть кнопку
На форму 8!", "Тест", MessageBoxButtons.OK, MessageBoxIcon.Information);
                checkBox1.Enabled = false; checkBox2.Enabled = false;
checkBox3.Enabled = false; checkBox4.Enabled = false;
            }
            else
            {
                MessageBox.Show("Ви відповіли неправильно!", "Тест",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                MessageBox.Show("Переходимо до наступного питання – натисніть кнопку
На форму 8!", "Тест", MessageBoxButtons.OK, MessageBoxIcon.Information);
                checkBox1.Enabled = false; checkBox2.Enabled = false;
checkBox3.Enabled = false; checkBox4.Enabled = false;
            }

        }
    }
}
```