МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**

**«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

**КУРСОВА РОБОТА**з дисципліни «Алгоритмізація та програмування»

Виконала студентка 1 курсу, групи КН-723

Вікторія Рослав

Перевірила Олена Нікуліна

Харків 2024

**Зміст**

[**Вступ** 3](#_Toc168052431)

[**Огляд літератури за темою дослідження. Постанова задачі дослідження** 5](#_Toc168052432)

[**1.** **Визначення** 5](#_Toc168052433)

[**2.** **Правила гри** 5](#_Toc168052434)

[**3.** **Користь та недоліки змійки** 6](#_Toc168052435)

[**4.** **Аналіз аналогів** 7](#_Toc168052436)

[**5.** **Постановка задачі дослідження** 12](#_Toc168052437)

[**6.** **Графічні засоби Visual Studio** 12](#_Toc168052438)

[**Розробка та тестування роботи** 14](#_Toc168052439)

[**1.** **Опис розробленої програми** 14](#_Toc168052440)

[**2.** **Тестування програми та аналіз** 16](#_Toc168052441)

[**3.** **Рекомендації** 18](#_Toc168052442)

[**Висновок** 18](#_Toc168052443)

[**Список джерел інформації** 20](#_Toc168052444)

[**Додатки** 22](#_Toc168052445)

# **Вступ**

Тема цієї роботи – розробка віртуальної гри «Змійка» на мові програмування С# з використанням додатку Visual Studio.

Мета цієї роботи – розробка візуального додатку в інтегрованому програмному середовищі для демонстрації роботи автоматного програмування [2, c.10]. Також метою курсової роботи є отримання практичних навичок в розробці додатку з використанням високорівневих методів програмування в середовищі Microsoft Visual Studio, мова програмування С#.

Ігри, особливо комп'ютерні та відеоігри, стали невід'ємною частиною сучасного суспільства. Вони не тільки пропонують розваги, але й мають значні переваги, які позитивно впливають на розвиток і життя людини.

Однією з найважливіших переваг ігор є розвиток когнітивних навичок. Граючи, людина часто стикається із завданнями, які вимагають запам'ятовування деталей, швидкої реакції та концентрації уваги. Це сприяє покращенню пам'яті та здатності швидко приймати рішення. Крім того, багато ігор вимагають розвитку просторового мислення, що корисно для виконання завдань, пов'язаних з орієнтуванням у просторі. Ще однією важливою перевагою є розвиток логічного та критичного мислення. Це звичайно не всі переваги ігор, але вони є основними.

Однією з найбільш популярних та впізнаваних класичних ігор є "Змійка". З моменту своєї появи у 1970-х роках, "Змійка" завоювала серця мільйонів гравців завдяки своїй простоті та захопливому геймплею. У цій роботі ми дослідимо створення гри "Змійка" за допомогою сучасних засобів програмування. Ми розглянемо основні етапи розробки цієї гри.

Крім того, ми розглянемо можливості покращення та модифікації гри для створення більш складних і цікавих версій "Змійки". Ця гра є чудовим прикладом для навчання програмуванню, оскільки вона включає базові концепції, такі як обробка подій, управління графічними об'єктами та розробка алгоритмів. Завдяки своїй модульній структурі, проект гри можна розділити на окремі етапи, кожен з яких має свої завдання та складності. Це дозволяє покроково розробляти гру, поступово нарощуючи її функціональність та складність.

У процесі розробки гри "Змійка" я використовувала популярну мову програмування та інструменти. В нашому випадку за допомогою Windows Forms C#. Це дасть не лише можливість ознайомитися з основами програмування графіки та взаємодії з користувачем за допомогою інтерактиву, а також розвинути навички роботи з сучасними технологіями саме в такій програмі як Visual Studio.

Гра включає в себе керування одним блоком або змієголовою, повертаючи лише ліворуч або праворуч на дев’яносто градусів, доки вам не вдасться з’їсти блок. Коли ви отримуєте блок, Змія вирощує додатковий блок або сегмент тіла. Якщо, точніше, коли змія врізається в край екрана або випадково з’їдає себе, гра закінчена. Чим більше блоків з'їдає змія, тим вище оцінка [1, c.9].

Результатом цієї роботи стане повнофункціональна гра "Змійка", яка не лише відтворює класичний геймплей, але й пропонує нові можливості та вдосконалення. Також розглянемо потенційні напрямки для подальшого розвитку гри, такі як додавання нових рівнів, впровадження складніших алгоритмів для штучного інтелекту та створення багатокористувацького режиму.

Таким чином, ця робота сприятиме як особистісному розвитку навичок програмування та проектування, так і збагаченню загального досвіду розробки комп'ютерних ігор, що може бути корисним у подальшій професійній діяльності.

# **Огляд літератури за темою дослідження. Постанова задачі дослідження**

## **Визначення**

Змійка (Snake Game) — це жанр відеоігор, де гравець керує лінією, яка постійно довшає і врешті стає основною перешкодою для самої себе. Концепція виникла 1976 року в аркадній грі для двох гравців Blockade від Gremlin Industries. Простота реалізації привела до появи сотень версій (деякі з яких мають у назві слово «змія» або «черв'як») для багатьох платформ. Аркадна гра Tron 1982 року, заснована на фільмі, включає ігровий процес про змію для одного користувача Light Cycles. Після того, як варіант з'явився на мобільних телефонах Nokia 1998 року, інтерес до цього типу ігор відродився, оскільки він знайшов більшу аудиторію. Тільки для iOS існує кілька сотень таких ігор [3].

## **Правила гри**

На ігровому полі з'являється змійка та її їжа, представлені набором фігур. У процесі гри змійка постійно повзе, гравець не може її зупинити, але може спрямовувати її голову вліво, вправо, вгору та вниз, а хвіст змійки рухається слідом. Завдання гравця полягає в тому, щоб змійка з'їла їжу, уникаючи зіткнень зі своїм хвостом і з межами ігрового поля. Після кожного з'їденого шматка їжі, змійка стає довшою (що ускладнює завдання гравцю), а їжа знову з'являється у випадковому місці ігрового поля.

Примітка: У деяких версіях гра не має меж ігрового поля, заповзаючи за екран, змійка з'являється з протилежного боку екрана. Є версії гри з додатковими перешкодами та навіть тривимірна версія гри [4].

В нашому випадку буде використовуватися концепція, коли користувач буде стикатися не лише зі своїх хвостом, а й з межами ігрового поля. Я вважаю цю ідею більш логічною.

## **Користь та недоліки змійки**

Давайте розглянемо основні переваги ігор:

Наприклад, вони привчають слідкувати за деталями, реакцію, розвивають дрібну моторику, покращують пам’ять, змушують думати, розвивають уяву, вчать підходити до вирішення завдань творчо та нестандартно. Граючи, дитина вчиться працювати з корисними програмами, а ще комп’ютерні ігри допоможуть цікаво провести вільний час наодинці [6].

Щодо нашої гри, можемо сказати також, що ця розвага дозволяє відволіктися від буденних проблем та весело провести час, вона має й багато інших переваг:

* Тренує спритність.
* Дає змогу проявити хитрість.
* Вмикає кмітливість та логічне мислення.
* Занурює у пригодницьку атмосферу.
* Має віртуальну карту, де показано позицію вашої змійки відносно інших об’єктів.

Сучасні варіації є різноманітними. Від початкових версій вони відрізняються тим, що рухи змійки та її аналогів не обмежуються прямими лініями, а ще характеризуються якісною різнобарвною графікою. Можна обирати варіант гри відповідно до вашого віку, вподобань, інтересів та настрою [4].

Давайте розглянемо основні недоліки ігор:

* Витрати часу

Деякі настільні ігри можуть забирати багато часу, тож буває складно знайти вільну годину для повноцінної гри. Це може бути недоліком для тих, хто має щільний графік або віддає перевагу швидшим іграм.

* Складнощі з налаштуванням і навчанням

Багато ігор мають складні правила та вимоги до налаштування, які можуть бути нелегкими для нових гравців.

* Обмеження рухливості

Часто для гри гравцям доводиться залишатися нерухомими протягом тривалого часу. Це може бути доволі незручним для тих, хто віддає перевагу активнішим видам діяльності [5].

Я вважаю, що в цій грі немає ніяких недоліків, які б могли бути в будь-якій більш складній грі. Це звичайна гра, де люди можуть розслабитися та потренувати свої навички реакції. Гра витрачає дуже мало часу та в ній немає складнощів з управлінням та налаштуванням.

## **Аналіз аналогів**

Зараз ми проаналізуємо аналоги гри «Змійка», для загального розуміння даної гри, а також щоб подивитися на їх розробку, дизайн, концепцію гри тощо. Також роздивимося, які є недоліки і переваги кожної гри.

Для аналізу подальшого оцінювання у нас є два аналоги:

1. <https://g.co/kgs/Eyauucb>
2. <https://logicplays.com/golovolomki/1005-classic-snake-html5.html>

Перейдемо до першого аналогу:

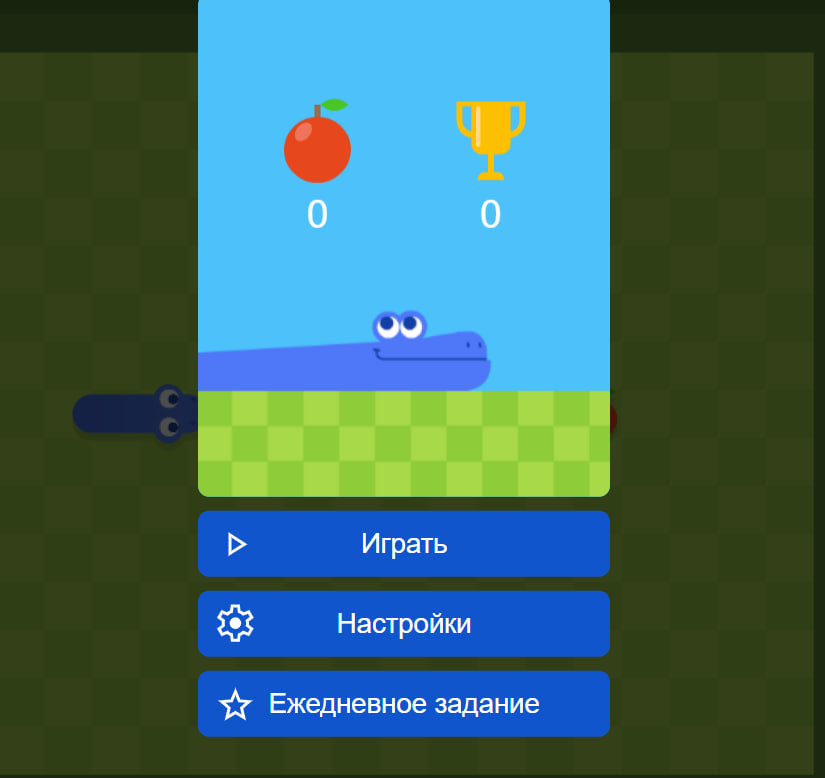
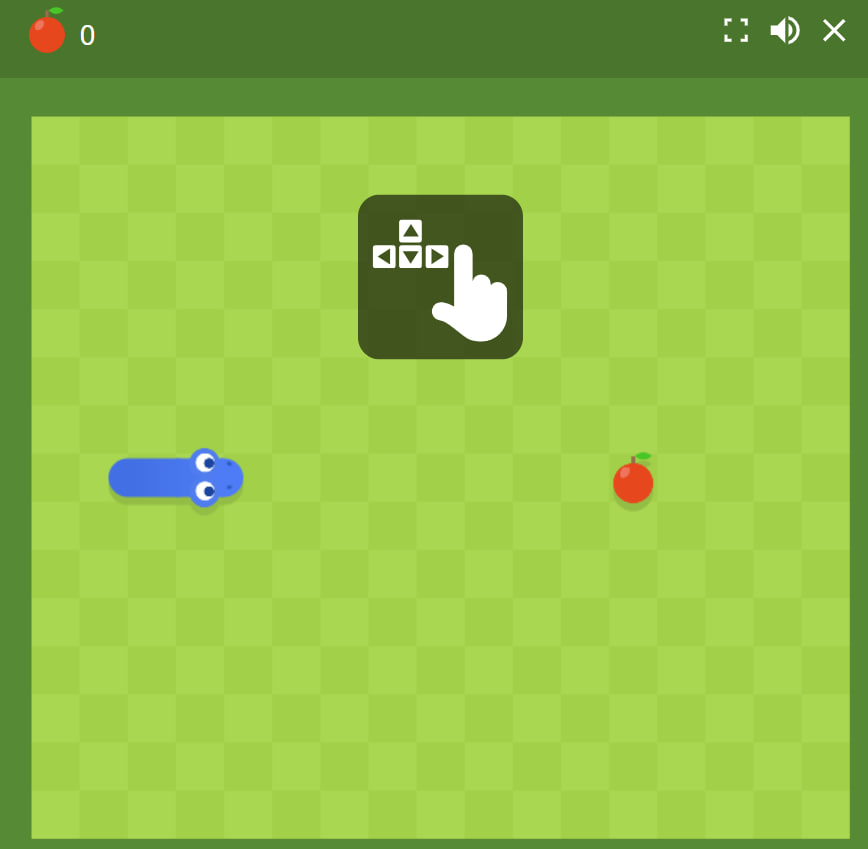
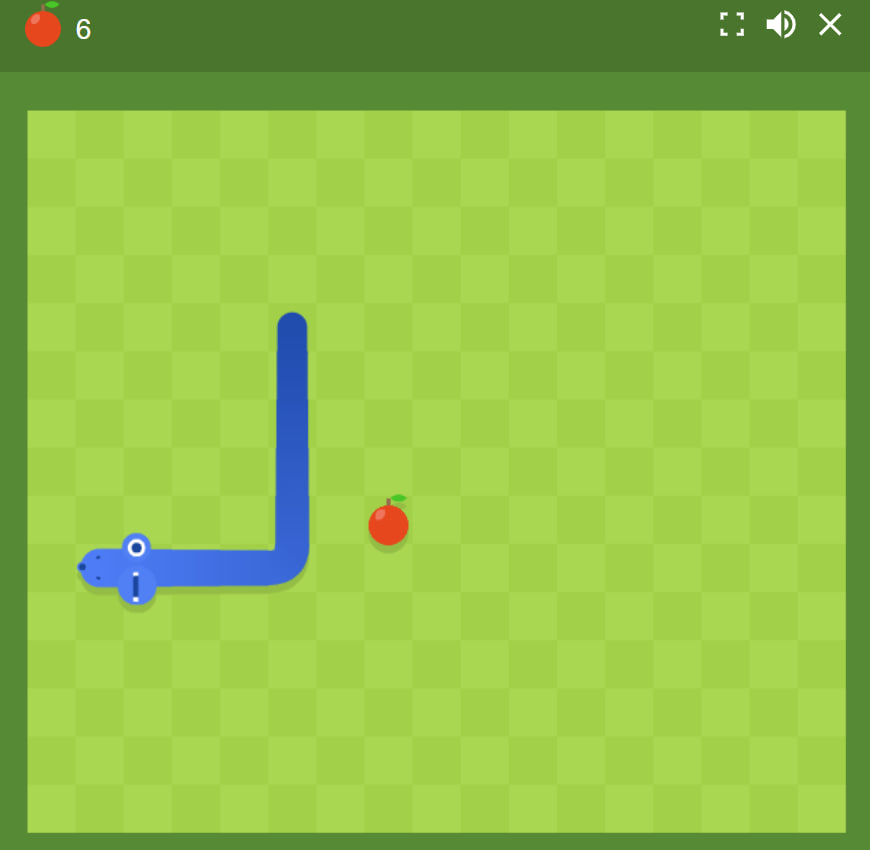
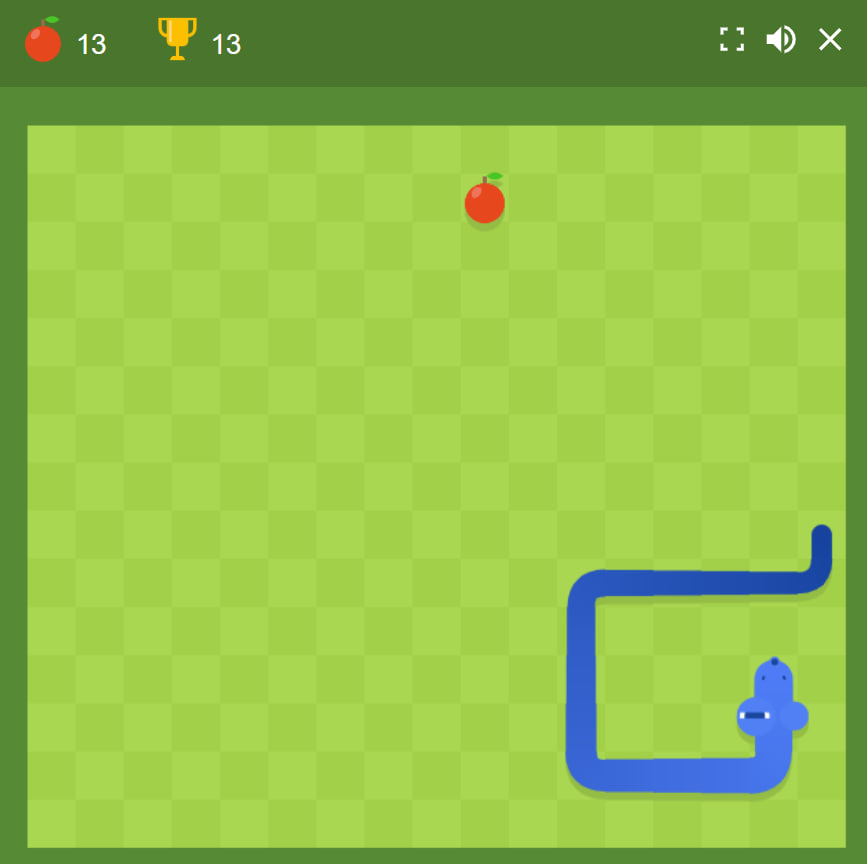
 

Рисунок 1.1 – Скріншоти з першої гри.

Коли починаємо гру у нас показує вікно меню. Після натискання кнопки «грати» починаємо гру не одразу, а після натискання будь-якій кнопки управління (рисунок 1.1).



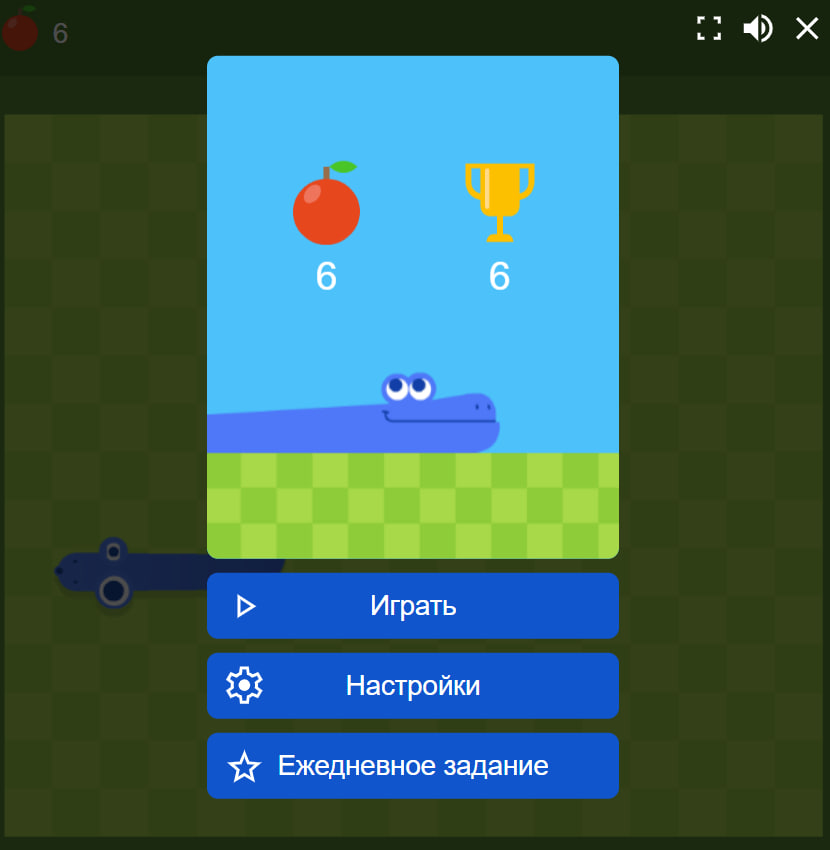


Рисунок 1.2 – Скріншоти з першої гри.

На даному етапі ми бачимо, що користувач закінчує гру коли змійка або натикається на саму себе, або коли змійка натикається на поля гри. Після закінчення гри з’являється віконце з меню. Якщо налаштування у нашій грі стоять за замовчуванням, тобто ми нічого в ній не міняли, то у ньому зображено скільки у користувач зібрав яблук та рекордне значення зібраних яблук (рисунок 1.2).

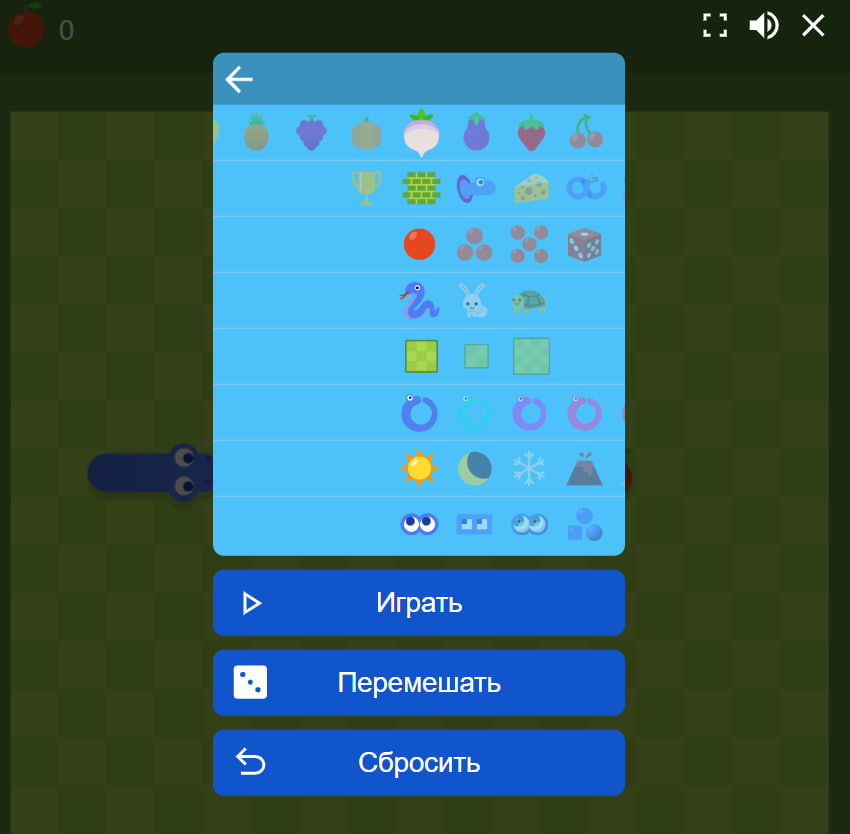


Рисунок 1.3 – Скріншоти з першої гри.

Також ми можемо зайти в налаштування гри та подивитися, що ми можемо поміняти в грі (рисунок 1.3).

Я обрала такі налаштування та отримала гру з перешкодами, вони будуть з’являтися кожного разу, коли я з’їм вишню (рисунок 1.4):

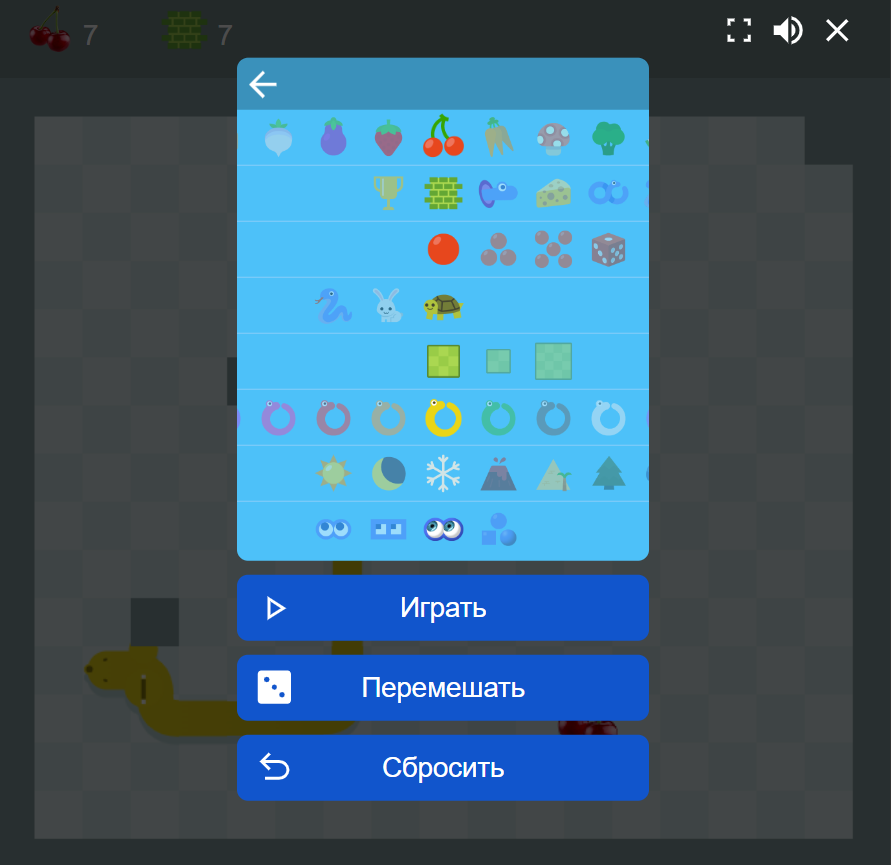


Рисунок 1.4 – Скріншоти з першої гри.

Переваги:

* Гарний інтерфейс.
* Можливість обрати налаштування, де є багато функцій для гри.
* Наявність меню.
* Чудова працездатність.
* Наявність перешкод для гри.

Недоліків не виявлено.

Перейдемо до другого аналогу:

Класична змійка, така яка була на звичайних кнопкових телефонах, хто ще застав ті часи. Управління з комп'ютера стрілочками, на планшеті трохи складніше, але розберетеся. Постарайся виростити змію максимального розміру, таку, щоб вона вже не влазила в жодний піксель на екрані. Це звичайно складно і займає багато часу. Але грати щоб згадати як це була кумедно у далеких 2000-х, це коштує. Проста графіка, прості правила, а що ще потрібне. Чим простіше – тим краще [7].

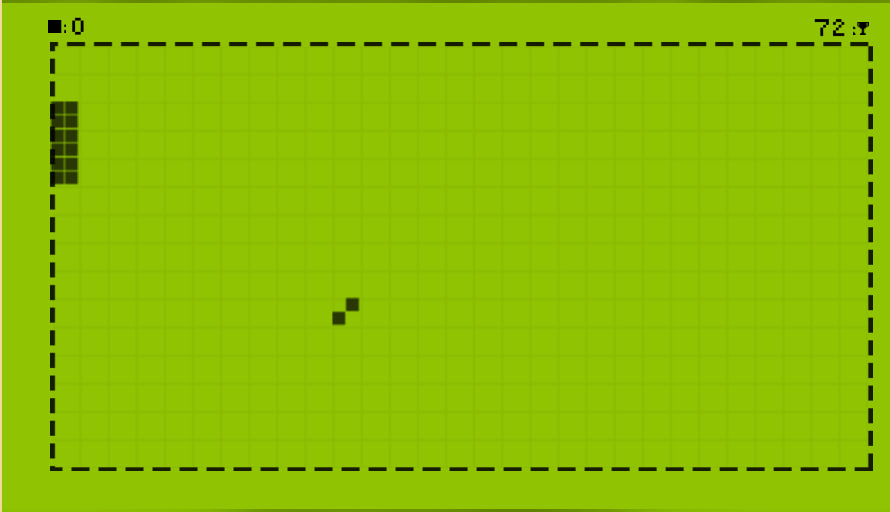
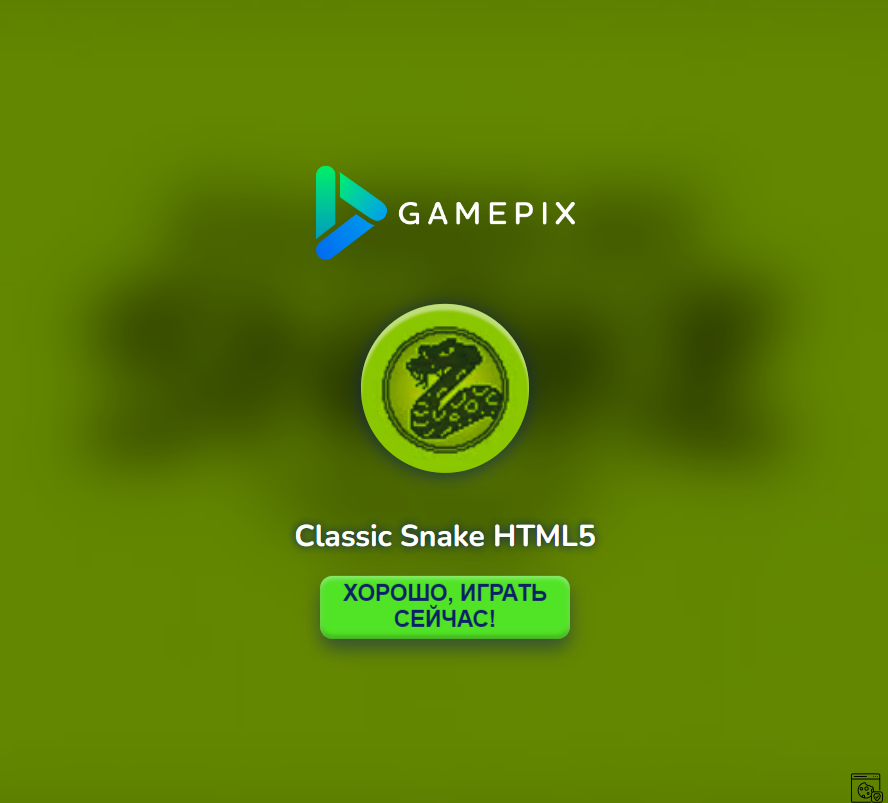


Рисунок 1.5 – Скріншоти з другої гри.

В даній грі ми бачимо, що у нас немає меню. Після натискання кнопки «добре, грати зараз!» починається гра. Змійка автоматично починає пересуватися без натискання кнопки управління, як це було в минулій грі (рисунок 1.5).



Рисунок 1.6 – Скріншоти з другої гри.

На даному скріншоті бачимо, що змійка може пересуватися повз границі гри та гра продовжується (рисунок 1.6).



Рисунок 1.7 – Скріншоти з другої гри.

Якщо на натикаємося на хвіст нашої змії гра закінчується (рисунок 1.7).

Переваги:

* Гарний інтерфейс.
* Чудова працездатність.

Недоліки:

* Немає можливості обрати налаштування (хоча це і дуже стара гра, але порівнюючи з попередньою грою можемо занести це до мінусу).
* Немає меню.
* Немає перешкод.

## **Постановка задачі дослідження**

Окрім розуміння поведінки потенційних споживачів та інформації про них, варто досліджувати та аналізувати конкурентів. Для цього здійснюється аналіз аналогів, виділяються сильні та слабкі сторони. Після цього створюються висновки для подальшого застосування їх у розробці.

Темою данної робити є розробка віртуальної гри «Змійка», тому варто проаналізувати аналоги таких додатків, виявити усі переваги та недоліки, щоб на основі висновків розробити ідеальну гру.

Для аналізу та подальшого дослідження було обрано дві аналоги.

## **Графічні засоби Visual Studio**

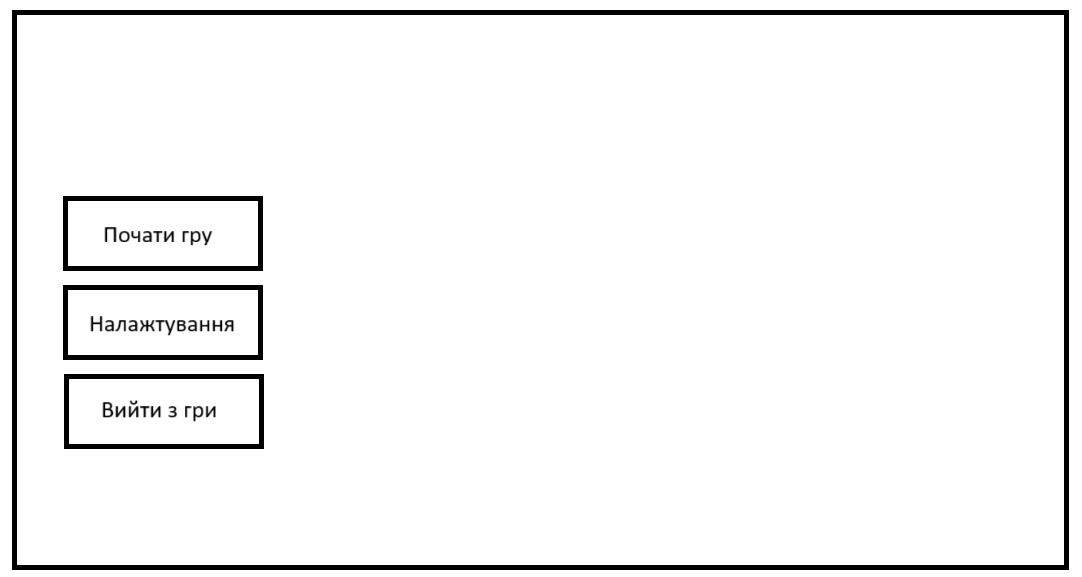


Рисунок 1.8 – Перше схематичне представлення інтерфейсу.

Інтерфейс гри буде спочатку містити робочу обсласть, яка складається з 3 кнопок (рисунок 1.8).

При натискані на першу, гра починається.

При натисканні на другу, відриваються налаштування (рисунок 1.9). В налаштуванням можна налаштувати гру за допомогою кнопок «змінити колір змійки», «змінити колір яблука», «змінити колір фону» та кнопка «Назад».

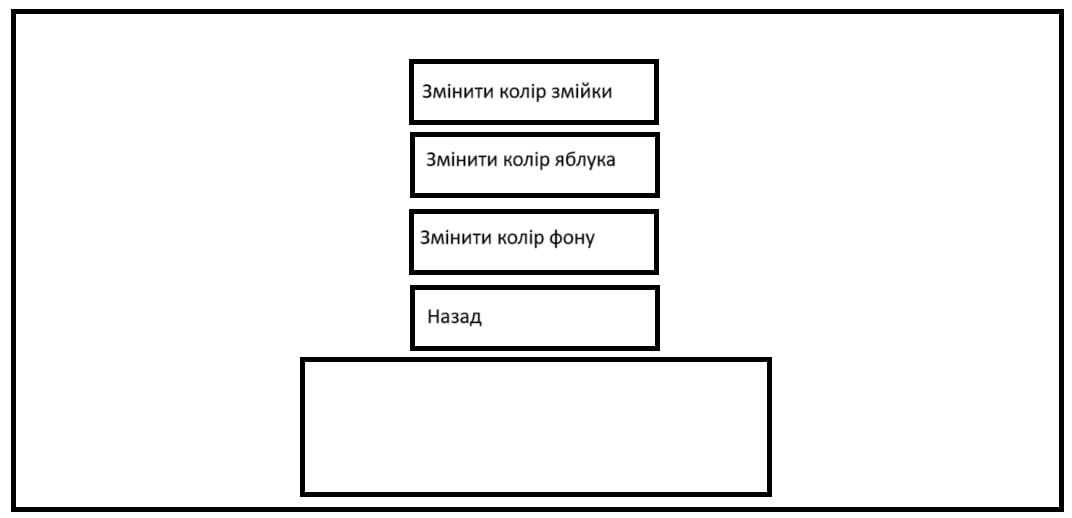


Рисунок 1.9 – Друге схематичне представлення інтерфейсу.

Гра закінчується, коли користувач стикається з своїм хвостом та з границею гри.

# **Розробка та тестування роботи**

## **Опис розробленої програми**

Гра "Змійка" належить до жанру аркадних ігор. Вона є класичним прикладом простої аркадної гри, яка включає основні елементи таких ігор: швидкий темп, просте керування, ігровий процес, орієнтований на реакцію та спритність гравця [8].

Завдання гравця полягає в керуванні змійкою, яка рухається по екрану, збираючи об'єкти (часто представлені як їжа), при цьому уникаючи зіткнення зі стінами або власним тілом, яке збільшується в довжину після кожного зібраного об'єкта.

1. Програма потрібно зробити в Windows forms C#.
2. Інтерфейс повинен складатися з двох кнопок. Обов’язкова кнопка «Почати гру» та додаткова кнопка «Налаштування», у якій будуть ще додаткові параметри.
3. Створити функцію *score\_num* у якій висвітлюється під час гри рахунок.
4. Створити функцію *button2\_Click* у якій написано, які кнопки видно після натискання кнопки «Налаштування».
5. Створити функцію *button3\_Click* у якій налаштовується випадково колір для змійки. Всього записано кольорів 15.
6. Створити функцію *button4\_Click* у якій налаштовується випадково колір для яблука. Всього записано кольорів 15.
7. Створити функцію *button5\_Click* у якій налаштовується випадково колір для фону. Всього записано кольорів 15.
8. Створити функцію *button6\_Click* у якій викликається функція *in\_el()*.
9. Створити функцію *button7\_Click* за допомогою якої можна вийти з гри.
10. Створити функцію *button1\_Click* у якій викликається функція *start\_game()*.
11. Створити функцію *start\_game*, яка інізіалізується початок гри та запуст нової гри.
12. Створити функцію *el\_apple*, яка відповідає за створення нового яблука в грі.
13. Створити функцію *update*, яка відповідає за постійний рух *run\_snake*; перевіряє та оброблює дії, такі як поїдання яблука та зіткнення за допомогою *collision*. Та й в цілому відображає таймер для гри, який при кожному кліку таймера змінюється.
14. Створити функцію *run\_snake*, у якій відбувається рух змійки.
15. Створити функцію *collision*, у якій відбувається перевірка на зіткнення змійки за двома способами. Перший – зіткнення з своїм хвостом. Другий – зіткнення з границею.
16. Створити функцію *game\_over*, що відповідає за кінець гри.
17. Створити функцію *OnPaint*, відповідає за малювання змійки та яблука на формі. Вона викликається кожного разу, коли форма потребує перерисовки (наприклад, при виклику методу *Invalidate*).
18. Створити функцію *ProcessCmdKey*, у якому відбувається управління гри.
19. Створити функцію *hide\_el*, яка демонструє після початку гри лише рахунок.
20. Створити функцію *in\_el*, яка демонтрує після кінця гри дві кнопки початку гри та налаштування.

## **Тестування програми та аналіз**



Рисунок 2.1 – Запуск гри.

Після запуску гри відкривається віконце у якому з’являється дві кнопки «Почати гру» та «Налаштування», а також додаткова «Вийти з гри»(рисунок 2.1).

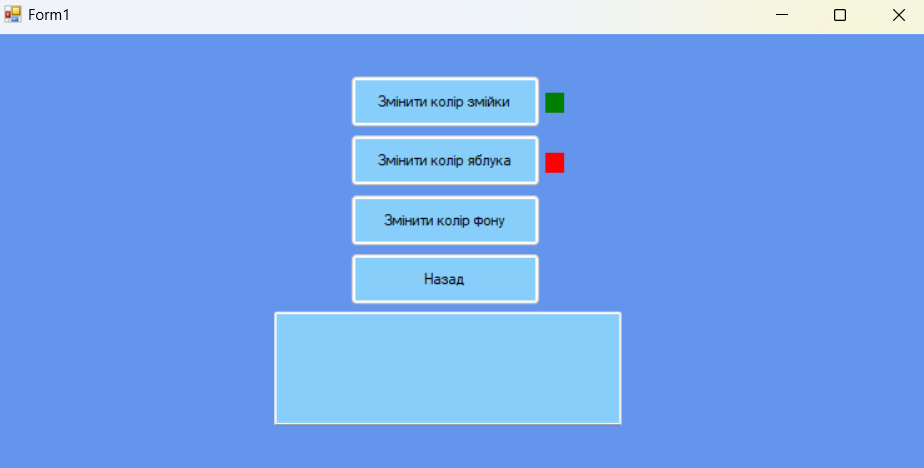


Рисунок 2.2 – Налаштування гри.

В налаштуваннях можна змінити колір змійки, змінити колір фону та змінити колір яблука. Також є кнопка «Назад», який повертає головний екран та текстовий файл у якому виводиться останнє обрану зміну в грі (рисунок 2.2). Наприклад, якщо обрати колір фону, буде виводити назву кольору (рисунок 2.3).



Рисунок 2.3 – Налаштування гри.

Після кнопки «Почати гру» гра починається. Також видно, що яблуко з’являється випадковим чином. Також бачимо, що є рахунок, який буде додаватися на один, коли змійка з’їсть яблуко (рисунок 2.4).

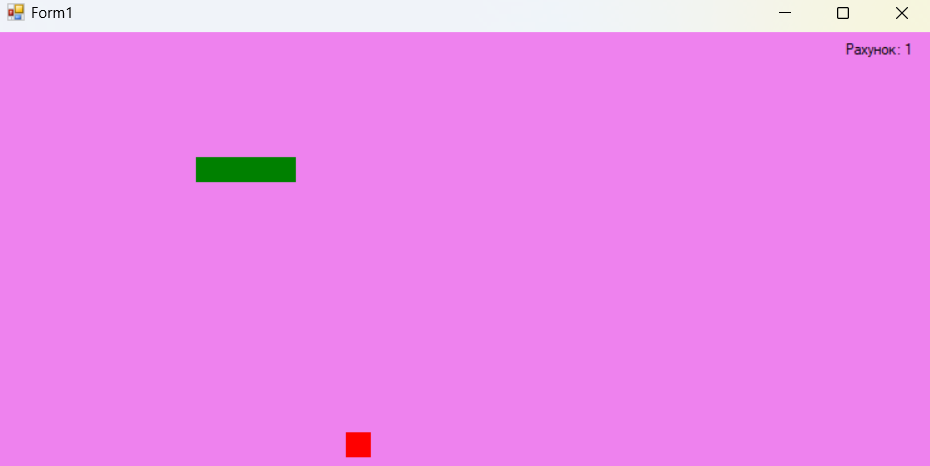


Рисунок 2.4 – Початок гри.

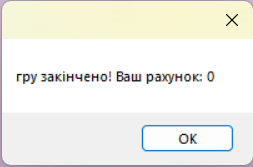


Рисунок 2.5 – Кінець гри.

Після закінчення гри з’являється віконце, що гру закінчено, а також буде висвітлюватись рахунок (рисунок 2.5).

## **Рекомендації**

Перед початком гри зробіть наступні дії.

1. Прочитати інструкцію перед початком гри.
2. Прочитати правила гри, щоб не виникло запитань.
3. Якщо хочете змінити інтерфейс гри проробіть наступні дії:

* Зайти в налаштування гри.
* Обрати налаштування інтерфейсу, який вам до вподоби.
* Почати гру.

# **Висновок**

В роботі було встановлено актуальність розробки в галузі комп’ютерних ігор, зокрема настільної гри «Змійка» для розвитку логічного мислення, уяви та пам’яті, сутність та функції комп’ютерної гри, основні її можливості, а також технології які будуть використовуватись при розробці гри.

В роботі було розроблено гру Змійка з інтерфейсом за допомогою Windows Forms C#. Було розроблено рекомендації з використання розробленної гри.

# **Список джерел інформації**

1. Bachelor of Technology in Computer Science and Engineering. Mrs. P Honey Diana Assistant Professor, 2020 – 25 c.: [Електронний ресурс] – режим доступу: <https://ru.scribd.com/document/540327030/Snake-Game-1-SRS-Documentation-pdf>
2. НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ “КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО”. Кузьмич Роман Олегович, 2022 – 87 с.: [Електронний ресурс] – режим доступу: <https://ela.kpi.ua/server/api/core/bitstreams/19d1672c-daf3-49c5-b3e0-3e01e055a944/content>
3. Змійка: матеріал з Вікіпедії – вільної енциклопедії: [Електронний ресурс] – режим доступу: <https://uk.wikipedia.org/wiki/%D0%97%D0%BC%D1%96%D0%B9%D0%BA%D0%B0_(%D0%B3%D1%80%D0%B0)>
4. <https://lesson.iarduino.ru/page/snake_arduino/#:~:text=%D0%9F%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D0%B0%20%D0%B8%D0%B3%D1%80%D1%8B%20%C2%AB%D0%97%D0%BC%D0%B5%D0%B9%D0%BA%D0%B0%C2%BB%3A&text=%D0%92%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B5%20%D0%B8%D0%B3%D1%80%D1%8B%20%D0%B7%D0%BC%D0%B5%D0%B9%D0%BA%D0%B0%20%D0%BF%D0%BE%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%BD%D0%BE,%D0%B8%20%D1%81%20%D0%B3%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0%D0%BC%D0%B8%20%D0%B8%D0%B3%D1%80%D0%BE%D0%B2%D0%BE%D0%B3%D0%BE%20%D0%BF%D0%BE%D0%BB%D1%8F>
5. <https://life.obozrevatel.com/ukr/section-zhizn/news-nastolnyie-igryi-kakie-u-nih-plyusyi-i-minusyi-24-01-2024.html>
6. <https://debaty.sumy.ua/news/yak-vplivayut-komp-yuterni-igri-na-organizm-ditini-korist-ta-shkoda-vid-onlajn-igor#:~:text=%D0%9D%D0%B0%D0%BF%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%2C%20%D0%B2%D0%BE%D0%BD%D0%B8%20%D0%BF%D1%80%D0%B8%D0%B2%D1%87%D0%B0%D1%8E%D1%82%D1%8C%20%D1%81%D0%BB%D1%96%D0%B4%D0%BA%D1%83%D0%B2%D0%B0%D1%82%D0%B8%20%D0%B7%D0%B0,%D0%B2%D0%B8%D1%80%D1%96%D1%88%D0%B5%D0%BD%D0%BD%D1%8F%20%D0%B7%D0%B0%D0%B2%D0%B4%D0%B0%D0%BD%D1%8C%20%D1%82%D0%B2%D0%BE%D1%80%D1%87%D0%BE%20%D1%82%D0%B0%20%D0%BD%D0%B5%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%BE>.
7. Перший аналог гри: [Електронний ресурс] – режим доступу: <https://g.co/kgs/Eyauucb>
8. Другий аналог гри: [Електронний ресурс] – режим доступу:

<https://logicplays.com/golovolomki/1005-classic-snake-html5.html>

1. Жанри відеоігор: матеріал з Вікіпедії — вільної енциклопедії: [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/%D0%96%D0%B0%D0%BD%D1%80%D0%B8_%D0%B2%D1%96%D0%B4%D0%B5%D0%BE%D1%96%D0%B3%D0%BE%D1%80>

# **Додатки**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

namespace Змейка

{

public partial class Form1 : Form

{

private List<Point> snake = new List<Point>();

private Point apple;

private int dir = 0;

private int w = 20;

private int h = 20;

private int score = 0;

private bool start\_g = false;

private Timer timer;

private Random r;

private Color snake\_c = Color.Green;

private Color apple\_c = Color.Red;

private Color form\_c = Color.CornflowerBlue;

public Form1()

{

InitializeComponent();

}

private void button2\_Click(object sender, EventArgs e)

{

button1.Visible = false;

button2.Visible = false;

label1.Visible = false;

pictureBox1.Visible = true;

pictureBox2.Visible = true;

button3.Visible = true;

button4.Visible = true;

button5.Visible = true;

button6.Visible = true;

button7.Visible = true;

textBox1.Visible = true;

}

private void button3\_Click(object sender, EventArgs e)

{

Color[] color = { Color.Green, Color.Lime, Color.SeaGreen, Color.Aquamarine, Color.MediumSeaGreen, Color.Teal, Color.DarkCyan, Color.DarkSlateGray, Color.White, Color.Black, Color.LightGreen, Color.PaleGreen, Color.DarkGreen, Color.OliveDrab, Color.GreenYellow };

Random ra = new Random();

int i = ra.Next(0, color.Length);

Color col = color[i];

pictureBox1.BackColor = col;

snake\_c = col;

textBox1.Text = "Колір для змійки: " + col.Name;

}

private void button4\_Click(object sender, EventArgs e)

{

Color[] color = { Color.Red, Color.Salmon, Color.Coral, Color.Chocolate, Color.LightSalmon, Color.SandyBrown, Color.DarkRed, Color.Firebrick, Color.IndianRed, Color.Brown, Color.Black, Color.White, Color.OrangeRed, Color.SaddleBrown, Color.LightCoral };

Random ra = new Random();

int i = ra.Next(0, color.Length);

Color col = color[i];

pictureBox2.BackColor = col;

apple\_c = col;

textBox1.Text = "Колір для яблука: " + col.Name;

}

private void button5\_Click(object sender, EventArgs e)

{

Color[] color = { Color.CornflowerBlue, Color.SlateBlue, Color.MediumPurple, Color.Violet, Color.LightPink, Color.Thistle, Color.Indigo, Color.PowderBlue, Color.AliceBlue, Color.Moccasin, Color.LemonChiffon, Color.Gold, Color.Silver, Color.DeepSkyBlue, Color.SkyBlue };

Random ra = new Random();

int i = ra.Next(0, color.Length);

Color col = color[i];

this.BackColor = col;

form\_c = col;

textBox1.Text = "Колір для фону: " + col.Name;

}

private void button6\_Click(object sender, EventArgs e)

{

in\_el();

}

private void in\_el()

{

button1.Visible = true;

button2.Visible = true;

label1.Visible = false;

pictureBox1.Visible = false;

pictureBox2.Visible = false;

button3.Visible = false;

button4.Visible = false;

button5.Visible = false;

button6.Visible = false;

button7.Visible = true;

textBox1.Visible = false;

}

private void button7\_Click(object sender, EventArgs e)

{

Close();

}

private void button1\_Click(object sender, EventArgs e)

{

start\_game();

}

private void hide\_el()

{

button1.Visible = false;

button2.Visible = false;

label1.Visible = true;

pictureBox1.Visible = false;

pictureBox2.Visible = false;

button3.Visible = false;

button4.Visible = false;

button5.Visible = false;

button6.Visible = false;

button7.Visible = false;

textBox1.Visible = false;

}

private void start\_game()

{

snake.Clear();

snake.Add(new Point(5, 5));

snake.Add(new Point(4, 5));

snake.Add(new Point(3, 5));

score = 0;

score\_num();

dir = 0;

start\_g = true;

r = new Random();

timer = new Timer();

timer.Interval = 100;

timer.Tick += update;

timer.Start();

el\_apple();

hide\_el();

}

private void el\_apple()

{

apple = new Point(r.Next(0, this.ClientSize.Width / w), r.Next(0, this.ClientSize.Height / h));

score\_num();

}

private void score\_num()

{

label1.Text = "Рахунок: " + score.ToString();

}

private void update(object sender, EventArgs e)

{

run\_snake();

collision();

this.Invalidate();

}

private void run\_snake()

{

for (int i = snake.Count - 1; i > 0; i--)

{

snake[i] = snake[i - 1];

}

switch (dir)

{

case 0:

snake[0] = new Point(snake[0].X + 1, snake[0].Y);

break;

case 1:

snake[0] = new Point(snake[0].X, snake[0].Y + 1);

break;

case 2:

snake[0] = new Point(snake[0].X - 1, snake[0].Y);

break;

case 3:

snake[0] = new Point(snake[0].X, snake[0].Y - 1);

break;

}

}

private void collision()

{

if (snake[0] == apple)

{

snake.Add(snake[snake.Count - 1]);

score++;

el\_apple();

}

if (snake[0].X < 0 || snake[0].X >= this.ClientSize.Width / w ||

snake[0].Y < 0 || snake[0].Y >= this.ClientSize.Height / h)

{

game\_over();

}

for (int i = 1; i < snake.Count; i++)

{

if (snake[0] == snake[i])

{

game\_over();

}

}

score\_num();

}

private void game\_over()

{

timer.Stop();

start\_g = false;

MessageBox.Show($"Гру закінчено! Ваш рахунок: {score}");

snake.Clear();

apple = Point.Empty;

this.Invalidate();

in\_el();

score = 0;

score\_num();

}

protected override void OnPaint(PaintEventArgs e)

{

Graphics g = e.Graphics;

if (start\_g)

{

foreach (var point in snake)

{

g.FillRectangle(new SolidBrush(snake\_c), new Rectangle(point.X \* w, point.Y \* h, w, h));

}

g.FillRectangle(new SolidBrush(apple\_c), new Rectangle(apple.X \* w, apple.Y \* h, w, h));

}

base.OnPaint(e);

}

protected override bool ProcessCmdKey(ref Message msg, Keys keyData)

{

switch (keyData)

{

case Keys.D:

if (dir != 2) dir = 0;

break;

case Keys.S:

if (dir != 3) dir = 1;

break;

case Keys.A:

if (dir != 0) dir = 2;

break;

case Keys.W:

if (dir != 1) dir = 3;

break;

}

return base.ProcessCmdKey(ref msg, keyData);

}

}

}