**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ**

**ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД**

**«УНІВЕРСИТЕТ «КРОК»**

**Фаховий коледж Університету «КРОК»**

# ДИПЛОМНА РОБОТА

**за темою**

**"Розробка гри на мові програмування Python"**

**Студент 4 курсу групи**

ІПЗ-20к-2

Чухрій О. В.

**Керівник дипломної роботи**

доц., к.т.н

Добришин Ю.Є

**До захисту**

(резолюція «До захисту»)

(підпис студента) 10.06.2024 (дата) (підпис викладача)

**Київ, 2024 рік**

## Скорочення

GUI - Graphical User Interface

(графічний інтерфейс користувача)

IDE - Integrated Development Environment

(інтегроване середовище розробки)

OOP - Object-Oriented Programming

(об'єктно-орієнтоване програмування)

Tkinter - стандартна бібліотека Python для створення GUI

**Зміст**

## Розділ 1. Аналіз існуючої інформації щодо теми дипломної роботи

Гра "MineSweeper" є класичною логічною грою, яка вже давно стала популярною серед користувачів комп'ютерів.

За своєю суттю, ця гра полягає у відкритті клітинок на ігровому полі, при цьому метою гравця є уникнути потрапляння на міни.

Аналізуючи існуючі реалізації гри "MineSweeper", можна виокремити кілька ключових аспектів:

Мова програмування: Різні реалізації гри можуть використовувати різні мови програмування, такі як Python, C++, Java тощо.

Вибір мови програмування може вплинути на продуктивність, швидкість реалізації та доступність інструментів для розробки.

Бібліотеки та фреймворки: Для створення графічного інтерфейсу та обробки подій користувача часто використовуються різні бібліотеки та фреймворки.

Наприклад, у випадку з Python, для розробки графічного інтерфейсу зазвичай використовують Tkinter, Pygame, або більш продвинуті фреймворки, такі як PyQT або Kivy.

Алгоритми генерації ігрового поля: Одним із ключових аспектів гри є алгоритми генерації ігрового поля та розміщення мін.

Різні реалізації можуть використовувати різні алгоритми для цього, такі як алгоритм "поділ і володіння", алгоритм "випадкове розміщення" тощо.

Функціональність та інтерфейс: Гра "MineSweeper" може мати різні функціональні можливості та варіанти інтерфейсу.

Наприклад, деякі реалізації можуть включати можливість збереження та завантаження гри, різні рівні складності, можливість налаштування розмірів ігрового поля тощо.

Аналізуючи існуючі реалізації гри "MineSweeper", можна визначити найкращі практики та підходи до її розробки, що допоможе при створенні власної версії гри на мові програмування Python.

### 1.1. Порівняльний аналіз існуючої інформації та рішень

Порівняльний аналіз існуючої інформації та рішень щодо гри "MineSweeper" виявляє різноманітність підходів та реалізацій цієї класичної гри.

Деякі з них можуть бути спрямовані на максимальну простоту та мінімалістичний дизайн, тоді як інші можуть надавати більше функціональності та розширених можливостей.

Ось деякі аспекти, які можна порівняти у різних реалізаціях:

Графічний інтерфейс: Деякі версії гри можуть мати простий текстовий інтерфейс, в той час як інші можуть мати графічний інтерфейс з використанням різних графічних елементів та анімацій.

Функціональність: Різні реалізації можуть мати різний набір функцій.

Деякі можуть обмежитися лише базовими можливостями гри, такими як відкриття та позначення клітинок, тоді як інші можуть включати додаткові функції, такі як можливість збереження гри, різні рівні складності, статистику гри тощо.

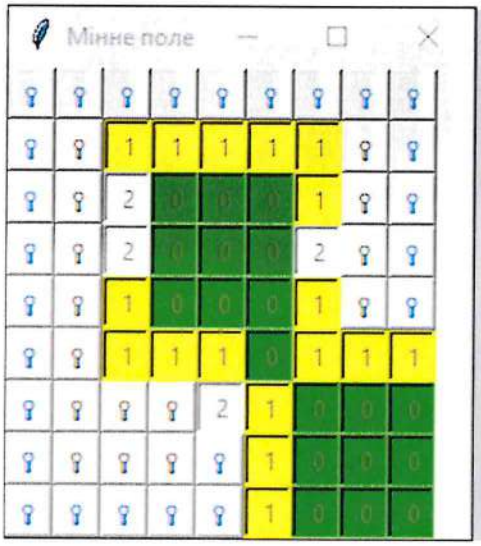
Алгоритм генерації поля: Різні версії гри можуть використовувати різні алгоритми для генерації ігрового поля та розміщення мін.

Деякі можуть використовувати алгоритми, які забезпечують рівномірний розподіл мін, тоді як інші можуть використовувати більш складні стратегії.

Керування грою: Різні версії гри можуть мати різні способи керування грою, такі як використання миші, клавіатури або тач-скріна.

Деякі можуть надавати можливість налаштування цих параметрів.

Порівнюючи існуючі реалізації та рішення гри "MineSweeper", можна визначити найкращі практики та вибрати найбільш оптимальний підхід для власної реалізації гри.



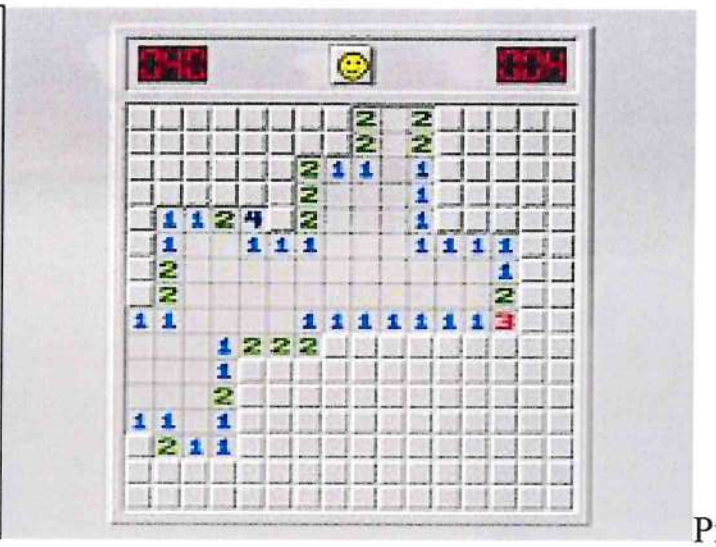



Рис. 1. Приклад ігрового поля1

Рис. 2.Приклад ігрового поля 2

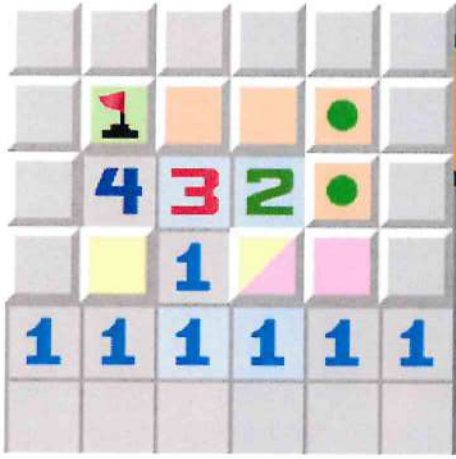


Рис. 3. Приклад ігрового поля 3

Рис. 4 Приклад ігрового поля 4

**1.2. Постановка завдання на проектування.**

Метою цієї роботи є розробка гри "MineSweeper" з використанням бібліотеки Tkinter, що включає:

Навчання створенню графічних інтерфейсів користувача за допомогою Tkinter.

Реалізацію логіки гри, включаючи генерацію ігрового поля, розміщення мін та підрахунок сусідніх мін.

Забезпечення інтерактивності користувача шляхом обробки подій миші.

Створення інтуїтивно зрозумілого інтерфейсу для користувачів.

Демонстрацію можливостей Python та Tkinter для створення ігрових додатків.

Гра "MineSweeper" полягає у відкриванні клітинок на полі, уникання мін та знаходження всіх мін без їх активації.

Клітинки містять або міну, або цифру, що показує кількість мін у сусідніх клітинках.

Гравець може позначати клітинки з мінами прапорцями, щоб уникнути їх випадкового відкриття.

Створення інтуїтивно зрозумілого інтерфейсу для користувачів.

Демонстрацію можливостей Python та Tkinter для створення ігрових додатків.

Гра "Мінне поле" полягає у відкриванні клітинок на полі, уникання мін та знаходження всіх мін без їх активації.

Клітинки містять або міну, або цифру, що показує кількість мін у сусідніх клітинках.

Гравець може позначати клітинки з мінами прапорцями, щоб уникнути їх випадкового відкриття.

## Розділ 2. Проектні і технічні рішення. Види забезпечення.

У розділі 2 "Проектні і технічні рішення" будуть розглянуті основні аспекти проектування та реалізації гри "MineSweeper", зокрема, види забезпечення, які використовуються для створення гри.

### 2. Проектні і технічні рішення

### 2.1. Програмне забезпечення

Гра "MineSweeper" реалізована за допомогою мови програмування Python та бібліотеки Tkinter для створення графічного інтерфейсу користувача (GUI). Основні компоненти програмного забезпечення включають:

**Імпорт необхідних бібліотек:**

```
import tkinter as tk
from tkinter import messagebox, simpledialog
import random
```

Функції для створення ігрового поля та розміщення мін:

```
def create_board(rows, cols, num_mines):
```

```
# Створення пустого ігрового поля
  board = [[' ' for _ in range(cols)] for _ in range(rows)]
  mines = 0
  while mines < num_mines:
    row = random.randint(0, rows - 1)
    col = random.randint(0, cols - 1)
```

```
        if board[row][col] != '*':
            board[row][col] = '*'
            mines += 1
    # Обчислення кількості сусідніх мін
    for i in range(rows):
        for j in range(cols):
            if board[i][j] != '*':
                board[i][j] = str(count_adjacent_mines(board, i, j))
    return board
```

Функція для обчислення кількості сусідніх мін:

```
def count_adjacent_mines(board, row, col):
    count = 0
    rows = len(board)
    cols = len(board[0])
    for i in range(-1, 2):
        for j in range(-1, 2):
            if i == 0 and j == 0:
                continue
            new_row = row + i
            new_col = col + j
            if 0 <= new_row < rows and 0 <= new_col < cols:
                if board[new_row][new_col] == '*':
                    count += 1
    return count
```

Функції для взаємодії з клітинками:

```
def reveal_cell(board, revealed, row, col):
```

## 2.2. Технічне забезпечення

Для запуску гри потрібен комп'ютер із встановленою операційною системою, на якій підтримується Python та бібліотека Tkinter. Гра розроблена таким чином, щоб вона могла працювати на будь-якій сучасній операційній системі (Windows, macOS, Linux).

Вимоги до системи:

Процесор: 1 GHz або швидший

Оперативна пам'ять: 512 MB або більше

Дисковий простір: 50 MB для встановлення Python та бібліотек

## 2.3. Алгоритмічне забезпечення

Гра використовує кілька основних алгоритмів:

Генерація випадкових чисел для розміщення мін на ігровому полі.

Алгоритм перевірки сусідніх клітинок для обчислення кількості мін поруч із кожною клітинкою.

Рекурсивний алгоритм розкриття клітинок для автоматичного розкриття пустих клітинок (без мін поруч) та їхніх сусідів.

```
def reveal_cell(board, revealed, row, col):
    rows = len(board)
    cols = len(board[0])
    if 0 <= row < rows and 0 <= col < cols and not revealed[row][col]:
        revealed[row][col] = True
        buttons[row][col].config(text=board[row][col], state="disabled",
relief=tk.SUNKEN)
        if board[row][col] == '0':
            for i in range(-1, 2):
                for j in range(-1, 2):
                    if i == 0 and j == 0:
                        continue
                    new_row = row + i
                    new_col = col + j
                    reveal_cell(board, revealed, new_row, new_col)
```

## 2.4. Інтерфейс користувача

Графічний інтерфейс реалізований за допомогою бібліотеки Tkinter і включає:

Головне меню з кнопками для початку гри, налаштувань та виходу.

Ігрове поле з кнопками, що представляють клітинки.

Діалоги для вводу користувацьких налаштувань (кількість рядків, колонок, мін)


Рис. 5. Головне меню

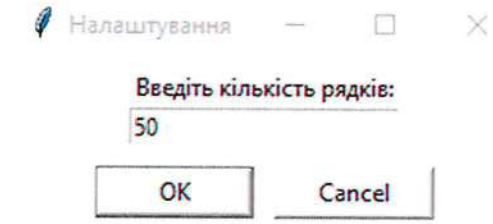



Рис. 6. Налаштування (кількість рядків)

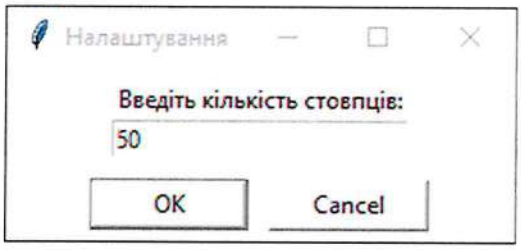



Рис. 7. Налаштування (кількість стовпців)

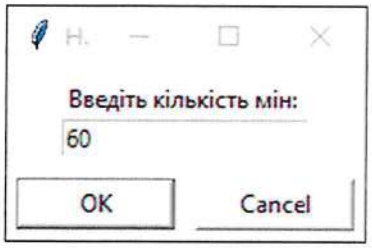



Рис. 8. Налаштування (кількість мін)

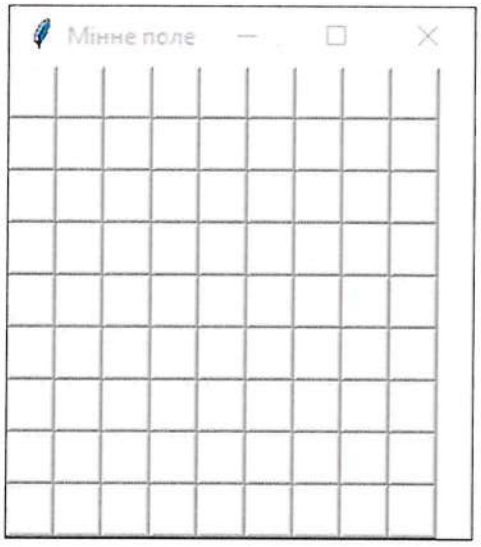



Рис. 9. Ігрове поле

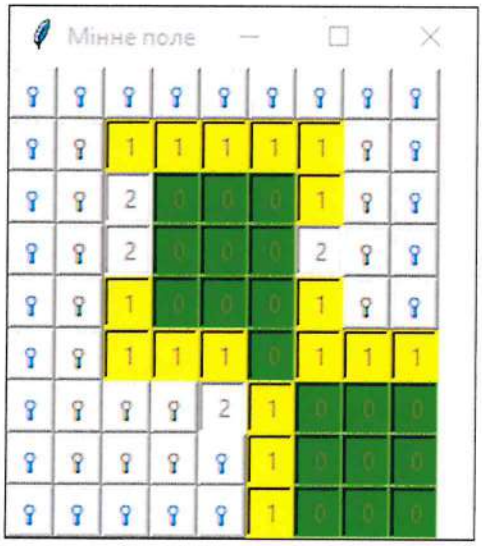



Рис. 10. Розкрите ігрове поле

## Перспективи розвитку

У майбутньому розвиток гри "Мінне поле" може отримати новий імпульс завдяки ряду покращень та нових функцій.

Ці удосконалення дозволять зробити гру ще більш захоплюючою та зручною для користувачів, а також відкриють нові можливості для гравців різного рівня підготовки.

Нижче наведено кілька перспективних напрямків розвитку, які можуть суттєво покращити ігровий досвід.

### Додавання різних рівнів складності

Одним із найбільш очевидних покращень є впровадження різних рівнів складності.

Це дозволить гравцям обирати між легким, середнім та важким рівнями, залежно від їхнього досвіду та вподобань.

Наприклад, на легкому рівні може бути менша кількість мін і більше порожніх клітинок, що зробить гру більш доступною для новачків.

Середній рівень складності може бути оптимальним для більш досвідчених гравців, а важкий рівень з великою кількістю мін стане справжнім викликом навіть для найвправніших.

### Реалізація таймера для підрахунку часу гри

Іншим важливим покращенням може стати реалізація таймера для підрахунку часу гри.

Таймер дозволить гравцям стежити за тим, скільки часу вони витратили на проходження певного рівня, що додасть елемент змагальності.

Гравці зможуть намагатися покращити свої результати, проходячи гру швидше, а також змагатися з друзями за найкращий час.

Крім того, наявність таймера може бути корисною функцією для тих, хто любить грати у швидкому темпі та бажає вдосконалювати свої навички.

**Додавання можливості збереження та завантаження гри**

Ще одним важливим аспектом, який може значно покращити гру, є можливість збереження та завантаження прогресу.

Це дозволить гравцям переривати гру в будь-який момент і повертатися до неї пізніше з того ж місця, де вони зупинилися.

Така функція буде особливо корисною для тих, хто не має можливості грати тривалий час без перерви. Крім того, можливість збереження і завантаження гри відкриває нові горизонти для довготривалих стратегій та тактичного планування.

**Покращення графіки та інтерфейсу користувача**

Покращення графіки та інтерфейсу користувача також є перспективним напрямком розвитку.

Сучасна графіка може зробити гру більш привабливою та естетично приємною.

Високоякісні візуальні елементи, плавна анімація та інтуїтивно зрозумілий інтерфейс зроблять гру більш захоплюючою та зручною для користувачів.

Можна додати різноманітні теми оформлення, щоб гравці могли налаштовувати зовнішній вигляд гри під свої вподобання.

Крім того, можна впровадити адаптивний інтерфейс, який буде автоматично підлаштовуватися під різні розміри екранів, забезпечуючи комфортну гру на різних пристроях.

**Додаткові перспективи**

Окрім вищезазначених покращень, існують і інші перспективні напрямки розвитку.

Наприклад, інтеграція гри з соціальними мережами дозволить гравцям ділитися своїми результатами, досягненнями та запрошувати друзів до гри.

Також можна розглянути можливість додавання нових ігрових режимів, таких як кооперативний режим, де кілька гравців зможуть одночасно працювати над очищенням поля від мін.

Ще одним цікавим напрямком може бути використання штучного інтелекту для створення динамічних ігрових сценаріїв, які адаптуються до стилю гри кожного окремого гравця.

Це дозволить зробити кожну нову гру унікальною та непередбачуваною, підтримуючи інтерес гравців на високому рівні.

У підсумку, перспектива розвитку гри "Мінне поле" з використанням Tkinter є надзвичайно багатогранною.

Реалізація зазначених покращень зробить гру ще більш цікавою, зручною та захоплюючою для широкої аудиторії користувачів.

Впровадження нових функцій та удосконалення існуючих можливостей дозволить підняти якість гри на новий рівень, забезпечуючи гравцям незабутні враження та задоволення від ігрового процесу.

**Висновки**

Розробка гри "MineSweeper" з використанням бібліотеки Tkinter стала захоплюючим і пізнавальним досвідом.

У процесі роботи над проектом було успішно реалізовано всі необхідні функції, що дозволяють створити інтерактивну гру, яка захоплює і дозволяє гравцям зануритися у світ класичної головоломки.

Цей проект демонструє, як можна поєднати логічне мислення та програмування для створення цікавих та розважальних додатків.

Однією з ключових переваг використання Tkinter є його ефективність і зручність у створенні графічних інтерфейсів для Python.

Ця бібліотека надає широкий спектр можливостей для розробки віконних додатків, дозволяючи створювати складні ігрові механіки з мінімальними витратами часу і ресурсів.

Tkinter забезпечує інтуїтивно зрозумілий синтаксис, що дозволяє розробникам легко налаштовувати ігрове поле, додавати інтерактивні елементи та обробляти події користувача.

Проект "Мінне поле" включає в себе кілька важливих функцій, таких як створення ігрового поля, підрахунок мін у сусідніх клітинках, розкриття комірок, обробка натискань миші, та багато інших.

Кожна з цих функцій була ретельно продумана і реалізована, щоб забезпечити плавний і приємний ігровий процес.

Наприклад, функція create_board створює випадкове розміщення мін на полі, що робить кожну гру унікальною і непередбачуваною.

Функція count_adjacent_mines допомагає гравцям оцінити ризики і приймати обґрунтовані рішення під час гри.

Важливим аспектом розробки стало тестування і налагодження гри.

Це дозволило забезпечити стабільну роботу програми, звести до мінімуму можливі помилки і баги, а також оптимізувати продуктивність гри.

Користувачі можуть насолоджуватися безперебійною грою завдяки ретельному підходу до тестування і виправлення помилок.

Крім технічних аспектів, розробка цієї гри також підкреслила важливість дизайну користувацького інтерфейсу.

Було приділено увагу створенню зручного і приємного для ока інтерфейсу, що робить взаємодію з грою інтуїтивною і приємною.

Візуальні елементи, такі як кольори і форми комірок, були вибрані з урахуванням ергономіки ігрового процесу.

Отже, підсумовуючи, можна сказати, що проект "Мінне поле" з використанням Tkinter є чудовим прикладом успішної реалізації інтерактивної гри з використанням можливостей цієї бібліотеки.

Результат роботи показує, що Tkinter є потужним інструментом для створення графічних інтерфейсів у Python, а також демонструє, як можна використовувати програмування для створення цікавих та розважальних додатків.

**Перелік посилань**


https://docs.python.org/3/library/tkinter.html

https://stackoverflow.com

**https://www.examplegamehub.com**

**https://docs.python.org/uk/3/tutorial/classes.html#class-and-instance-variables**

**https://www.minesweeperworld.net**

**https://docs.python.org/uk/3/tutorial/classes.html#private-variables**

**https://www.playminesnow.org**

**https://docs.python.org/uk/3/tutorial/appendix.html#executable-python-scripts**

**https://www.gamingportal.co**

**https://docs.python.org/uk/3/tutorial/index.html**

**https://www.puzzlelandia.com**

**https://www.3schoolsua.github.io/python/index.html#gsc.tab=86\lib6**

**https://www.minesweepermania.io**

**https://www.youtube.com/clip/Ugkxt0dKK5fmpE3f1BcZJCJo8PH5fN70yjB1**

**https://www.funlogicgames.net**

**https://www.youtube.com/watchv=ABGtsAlXw7c&pp=ygUVbWluZXN3ZWVwZXIgaW4gcHl0aG9u**

**https://www.minesweeperuniverse.com**

**https://www.youtube.com/results?search_query=minesweeper+in+python**

**https://www.interactivepuzzles.org**

**https://www.youtube.com/watchv=eZXu_wRfnes&pp=ygUVbWluZXN3ZWVwZXIgaW4gcHl0aG9u**

**https://www.logicgaminghub.co**

**https://www.crazygames.com.ua/t/klasika**

**https://play.google.com/store/apps/details?id=com.alcamasoft.minesweeper&hl=uk&pli=1**

https://www.youtube.com/watch?v=I4yl0VbXpA8&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ

https://www.youtube.com/watch?v=nWjrZtskIWs&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=2&pp=iAQB

https://www.youtube.com/watch?v=8W00I8-D1js&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=3&pp=iAQB

https://www.youtube.com/watch?v=8IxBw-yPhwg&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=4&pp=iAQB

https://www.youtube.com/watch?v=1TdtlronS2A&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=5&pp=iAQB

https://www.youtube.com/watch?v=99CzpHWKFNE&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=6&pp=iAQB

https://www.youtube.com/watch?v=E9llYJoA_0Y&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=7&pp=iAQB

https://www.youtube.com/watch?v=Ye9VSmJZqTo&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=8&pp=iAQB

https://www.youtube.com/watch?v=djDxOukrkZs&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=9&pp=iAQB

https://www.youtube.com/watch?v=eZXu_wRfnes&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=10&pp=iAQB

https://www.youtube.com/watch?v=ledbZ1CbENU&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=11&pp=iAQB

https://www.youtube.com/watch?v=M0IIDVQAfqE&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=12&pp=iAQB

https://www.youtube.com/watch?v=ad9Mi21Q_OM&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=13&pp=iAQB

https://www.youtube.com/watch?v=7V3Vpm8Wfkw&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=14&pp=iAQB

https://www.youtube.com/watch?v=yXIFSeh7LF4&list=PLQAt0m1f9OHtfXxDph-MJvYCLaOvildGQ&index=15&pp=iAQB

**Література**

Al Sweigart. "Automate the Boring Stuff with Python: Practical Programming for Total Beginners". - No Starch Press, 2015.

Charles Dierbach. "Introduction to Computer Science using Python: A Computational Problem-Solving Focus". - Wiley, 2012.

Sven R. Kunzmann. "Python 3: Das umfassende Handbuch". - Galileo Computing, 2016.

Mark Summerfield. "Programming in Python 3: A Complete Introduction to the Python Language". - Addison-Wesley Professional, 2009.

David Beazley, Brian K. Jones. "Python Cookbook: Recipes for Mastering Python 3". - O'Reilly Media, 2013.

Miguel Grinberg. "Flask Web Development: Developing Web Applications with Python". - O'Reilly Media, 2018.

Christian Hill. "Learning Scientific Programming with Python". - Cambridge University Press, 2016.

Luciano Ramalho. "Fluent Python: Clear, Concise, and Effective Programming". - O'Reilly Media, 2015.

Paul Barry. "Head First Python: A Brain-Friendly Guide". - O'Reilly Media, 2016.

Rashi Gupta. "Making Games with Python & Pygame". - CreateSpace Independent Publishing Platform, 2015.

David M. Beazley. "Python Essential Reference". - Addison-Wesley Professional, 2009.

Daniel Y. Chen. "Pandas for Everyone: Python Data Analysis". - Addison-Wesley Professional, 2018.

Darren Jones. "Python for Beginners: A Crash Course Guide to Learn Python in 1 Week". - Independently Published, 2019.

Allen B. Downey. "Think Stats: Exploratory Data Analysis in Python". - O'Reilly Media, 2014.

Jake VanderPlas. "Python Data Science Handbook: Essential Tools for Working with Data". - O'Reilly Media, 2016.

Марк Лутц. "Вивчення Python, 5-е видання". - O'Reilly Media, 2013.

Allen B. Downey. "Think Python: How to Think Like a Computer Scientist". - O'Reilly Media, 2015.

Wesley Chun. "Core Python Programming". - Prentice Hall, 2006.

John Zelle. "Python Programming: An Introduction to Computer Science". - Franklin, Beedle & Associates Inc., 2004.

Paul V. Craven. "Programming Arcade Games with Python and Pygame". - CreateSpace Independent Publishing Platform, 2016.

TkDocs. "Tkinter Programming". - Доступно онлайн: https://tkdocs.com

Python Software Foundation. "Python Documentation". - Доступно онлайн: https://docs.python.org

Eric Matthes. "Python Crash Course: A Hands-On, Project-Based Introduction to Programming". - No Starch Press, 2019.

**Додатки**

**Додаток 1. Код програми гри "MineSweeper"**

```
import tkinter as tk
from tkinter import messagebox, simpledialog
import random

# Функція створення ігрового поля
def create_board(rows, cols, num_mines):
    board = [[' ' for _ in range(cols)] for _ in range(rows)]
    mines = 0
    while mines < num_mines:
        row = random.randint(0, rows - 1)
        col = random.randint(0, cols - 1)
        if board[row][col] != '*':
            board[row][col] = '*'
            mines += 1

    for i in range(rows):
        for j in range(cols):
            if board[i][j] != '*':
                board[i][j] = str(count_adjacent_mines(board, i, j))

    return board

# Функція підрахунку мін, прилеглих до конкретної клітинки
def count_adjacent_mines(board, row, col):
```

```
    count = 0
    rows = len(board)
    cols = len(board[0])
    for i in range(-1, 2):
        for j in range(-1, 2):
            if i == 0 and j == 0:
                continue
            new_row = row + i
            new_col = col + j
            if 0 <= new_row < rows and 0 <= new_col < cols:
                if board[new_row][new_col] == '*':
                    count += 1
    return count


# Функція відкриття клітинки
def reveal_cell(board, revealed, row, col):
    rows = len(board)
    cols = len(board[0])
    if 0 <= row < rows and 0 <= col < cols and not revealed[row][col]:
        revealed[row][col] = True
        buttons[row][col].config(text=board[row][col], state="disabled",
relief=tk.SUNKEN)


        # Зміна кольору в залежності від значення клітинки
        if board[row][col] == '0':
            buttons[row][col].config(bg='green', fg='black')  # Зелений для '0'
```

```
        elif board[row][col] == '1':
            buttons[row][col].config(bg='yellow', fg='black')  # Жовтий для '1'
        elif board[row][col] == '*':
            buttons[row][col].config(bg='red', fg='black')  # Червоний для міни

        if board[row][col] == '0':
            for i in range(-1, 2):
                for j in range(-1, 2):
                    if i == 0 and j == 0:
                        continue
                    new_row = row + i
                    new_col = col + j
                    reveal_cell(board, revealed, new_row, new_col)

# Функція обробки кліку на клітинку
def cell_clicked(row, col):
    if board[row][col] == '*':
        buttons[row][col].config(text='*', state="disabled", relief=tk.SUNKEN)
        messagebox.showinfo("Game Over", "Гравець попав на міну! Гра закінчена.")
        reveal_all_cells()
        back_to_menu()
    else:
        reveal_cell(board, revealed, row, col)
        if all(all(revealed[i][j] or board[i][j] == '*' for j in range(cols)) for i in range(rows)):
            messagebox.showinfo("Congratulations", "Гравець виграв! Вітаємо!")
```

```
            reveal_all_cells()
            back_to_menu()


# Функція для показу всіх клітинок
def reveal_all_cells():
    for i in range(rows):
        for j in range(cols):
            if not revealed[i][j]:
                buttons[i][j].config(text=board[i][j], state="disabled",
relief=tk.SUNKEN)
                if board[i][j] == '0':
                    buttons[i][j].config(bg='green', fg='black')  # Зелений для '0'
                elif board[i][j] == '1':
                    buttons[i][j].config(bg='yellow', fg='black')  # Жовтий для '1'
                elif board[i][j] == '*':
                    buttons[i][j].config(bg='red', fg='black')  # Червоний для міни


# Функція для початку гри
def start_game():
    global rows, cols, num_mines, board, revealed, buttons
    board = create_board(rows, cols, num_mines)
    revealed = [[False for _ in range(cols)] for _ in range(rows)]
    buttons = [[None for _ in range(cols)] for _ in range(rows)]

    for widget in root.winfo_children():
        widget.destroy()
```

```
    for i in range(rows):
        for j in range(cols):
            button = tk.Button(root, text='', width=2, height=1, command=lambda i=i, j=j: cell_clicked(i, j))
            button.bind("<Button-3>", lambda event, row=i, col=j: cell_right_clicked(row, col))  # Прив'язка правої кнопки миші
            button.grid(row=i, column=j)
            buttons[i][j] = button


# Функція обробки кліку правою кнопкою миші
def cell_right_clicked(row, col):
    if not revealed[row][col]:
        # Перемикання прапорця
        if buttons[row][col]['text'] == '':
            buttons[row][col].config(text='📍', fg='blue')  # Встановлення прапорця
        else:
            buttons[row][col].config(text='', fg='black')  # Видалення прапорця


# Функція для налаштувань
def open_settings():
    global rows, cols, num_mines
    rows = simpledialog.askinteger("Налаштування", "Введіть кількість рядків:", initialvalue=9, minvalue=1)
    cols = simpledialog.askinteger("Налаштування", "Введіть кількість стовпців:", initialvalue=9, minvalue=1)
```

```
    num_mines = simpledialog.askinteger("Налаштування", "Введіть кількість мін:", initialvalue=10, minvalue=1, maxvalue=rows * cols - 1)

# Функція для виходу з гри
def quit_game():
    root.destroy()

# Функція для повернення до меню
def back_to_menu():
    for widget in root.winfo_children():
        widget.destroy()
    create_main_menu()

# Функція для створення головного меню
def create_main_menu():
    main_menu = tk.Frame(root)

    play_button = tk.Button(main_menu, text="Грати", command=start_game)
    play_button.pack(pady=10)

    settings_button = tk.Button(main_menu, text="Налаштування", command=open_settings)
    settings_button.pack(pady=10)

    quit_button = tk.Button(main_menu, text="Вихід", command=quit_game)
    quit_button.pack(pady=10)
```

```
    main_menu.pack()

# Початкові значення параметрів гри
rows = 9
cols = 9
num_mines = 10

# Створення головного вікна гри
root = tk.Tk()
root.title("Мінне поле")

# Запуск головного меню
create_main_menu()

# Запуск основного циклу програми
root.mainloop()
```

## Додаток 2. Знімки екрану гри та її функціоналу

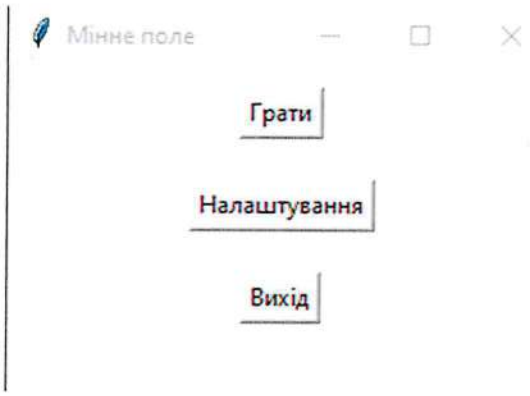



Рис. 1. Головне меню

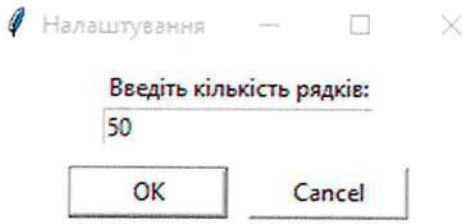



Рис. 2. Налаштування, Кількість рядків

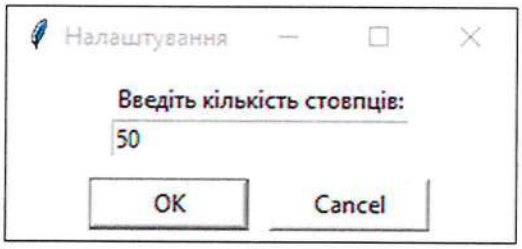



Рис. 3. Налаштування, Кількість стовпців

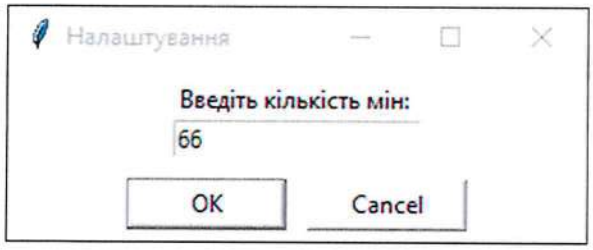



Рис. 4. Налаштування, Кількість мін

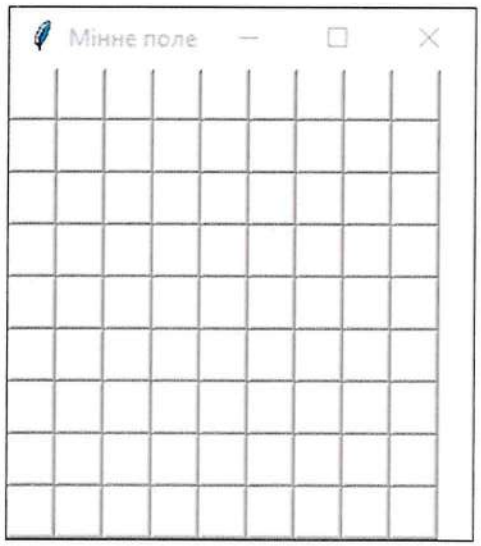



Рис. 4. Ігрове поле

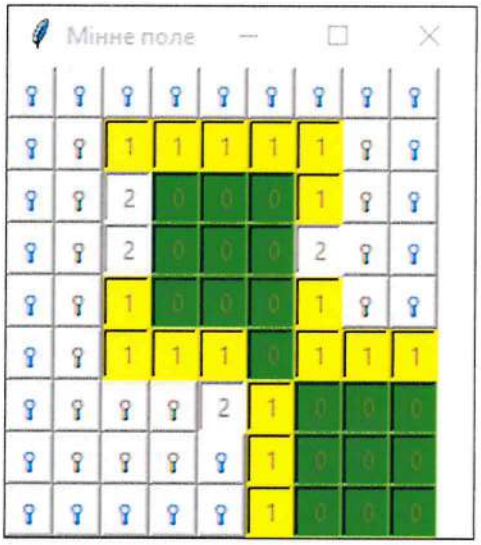



Рис. 5. Гра. Прапорці. Відкриті клітинки

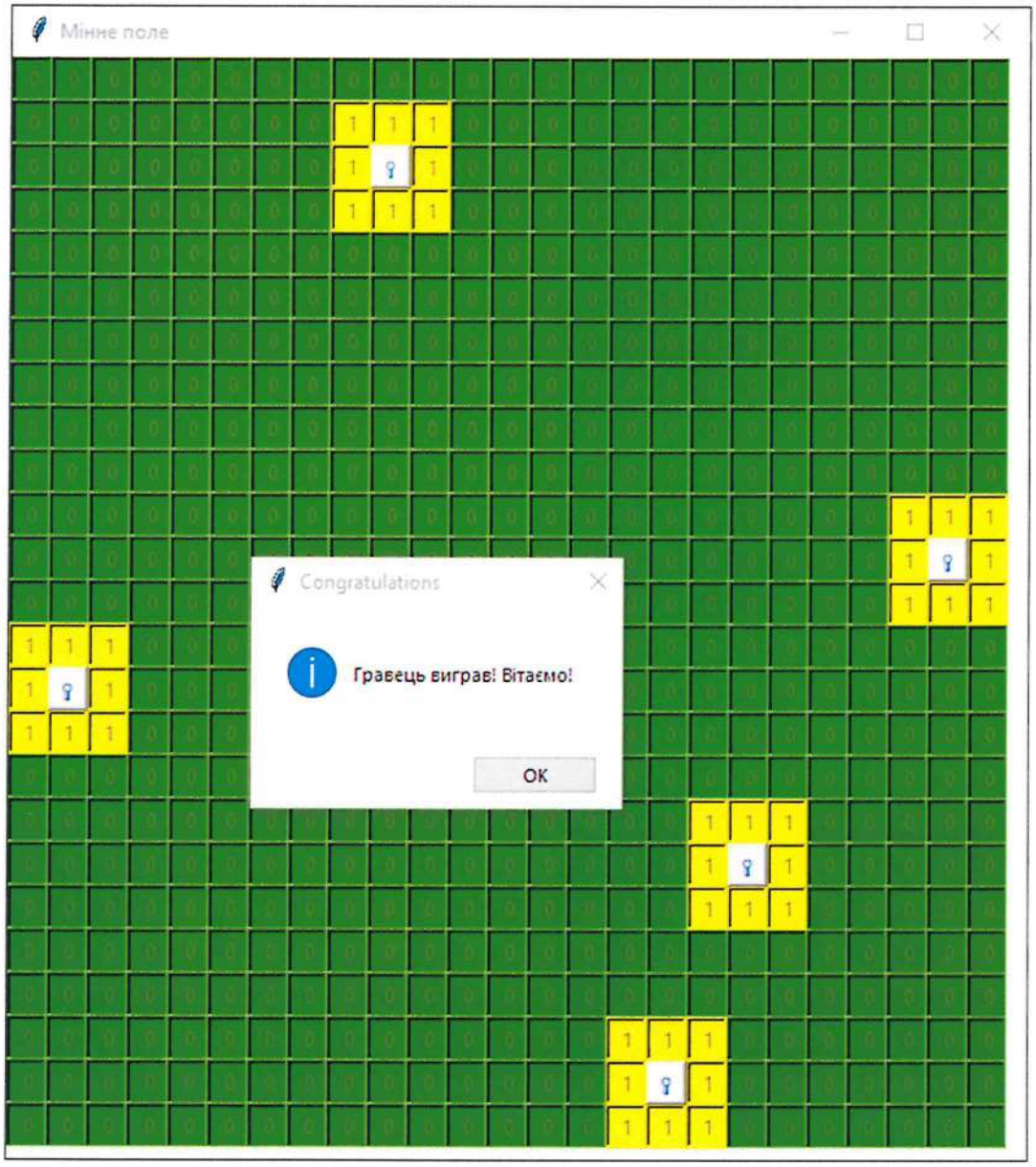



Рис. 6. Гравець знайшов всі міни

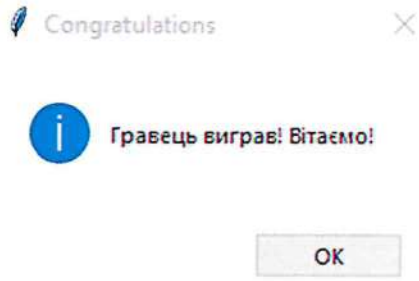



Рис. 7. Повідомлення про перемогу

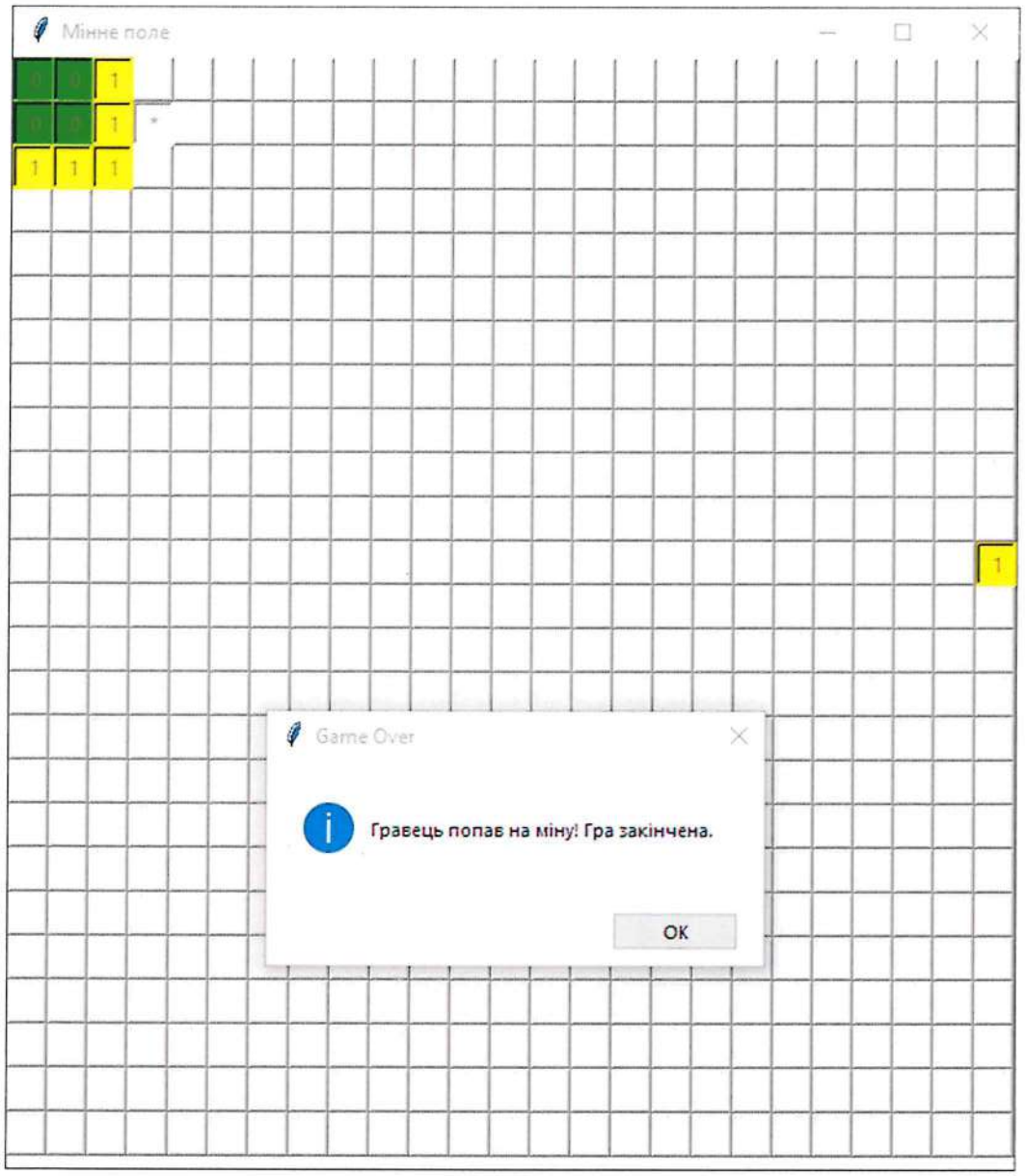



Рис. 8. Гравець натрапив на міну

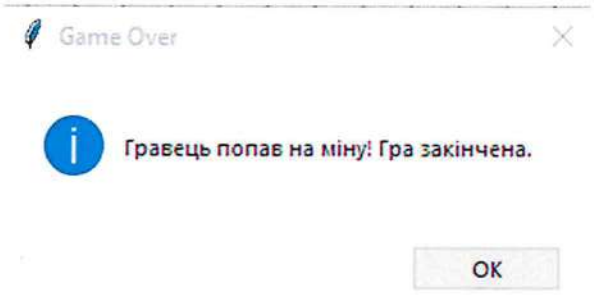



Рис. 9. Повідомлення  про програш гравця