НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ім. Ігоря Сікорського»
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

КУРСОВА РОБОТА

з дисципліни

Програмування та алгоритмічні мови

на тему: *ігрова програма «Сапер»*

Студента 1 курсу групи КА-11
Галузь знань 12 Інформаційні технології

Сороки Дмитра Володимировича

Керівник старший викладач кафедри
ММСА_Назарчук Ірина Василівна

Національна оцінка _____
Кількість балів: _____Оцінка: ECTS _____

Члени комісії _____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)
_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)
_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали

Київ – 2022 рік

НТУУ „КПІ ім.І.Сікорського” ІПСА
(назва вищого закладу освіти)

Кафедра	***математичних методів системного аналізу***

Дисципліна	***Програмування та алгоритмічні мови***

Галузь знань	***12 Інформаційні технології***

Курс	***перший***	Група	***КА—11***	Семестр	***другий***

ЗАВДАННЯ
на курсовий проект(роботу) студента

Сорока Дмитро Володимирович
(прізвище, ім’я, по батькові)

1. Тема проекту(роботи)	Ігрова програма «Сапер»

2. Строк здачі студентом закінченого проекту(роботи)	***20.06.2022 р.***

3. Вихідні дані до проекту(роботи)	Створення ігрової програми «Сапер» за класичними
ігровими правилами в алфавітно-цифровому режимі консолі вікна з використанням	
керуючих клавіш для управління грою та генеруванням випадкових чисел.	

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)	
1. Постановка задачі.	
2. Метод розв’язку задачі	
3. Загальна блок-схема алгоритму та опис алгоритму	
4. Опис програмного продукту.	
5. Результати роботи.	
6. Висновки.	
7. Список літератури.	
Додаток А. Текст програми.	

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)	
1. Загальна блок-схема алгоритму.	
2. Ілюстрації роботи програми.	

6. Дата видачі завдання	11.02.2022

КАЛЕНДАРНИЙ ПЛАН

№/п	Назва етапів курсового проекту (роботи)	Строк виконання етапів роботи	Примітка
1.	Вибір теми курсової роботи. Опрацювання відповідної літератури. Оформлення листа Завдання.	21.02.2022	
2.	Аналіз постановки задачі.	04.04.2022	
3.	Вибір та дослідження методів, вибір відповідних структур даних, розробка алгоритму. Перше узгодження з керівником.	11.04.2022	
4.	Проектування інтерфейсу.	11.04.2022	
5.	Друге узгодження з керівником.	18.04.2022	
6.	Програмна реалізація.	28.04.2022	
7.	Демонстрація першого варіанту. Третє узгодження з керівником.	05.05.2022	
8.	Заключне тестування програми.	19.05.2022	
9.	Аналіз результатів. Оформлення звіту.	26.05.2022	
10.	Захист та демонстрація курсової роботи.	16.06.2022 – «A» 22.06.2022 –«B,C» 27.06.2022 – «D,E»	

Студент		
	(підпис)	

Керівник		
	(підпис)	(прізвище, ім’я, по батькові)

(дата)	

Анотація

Представлена курсова робота є реалізацією консольної ігрової програми «Сапер». Гра-головоломка була виконана за допомогою псевдографічних елементів на основі класичної гри «Сапер».

При написанні курсової роботи використовувалась мова програмування C++, графічне оформлення було створено на основі графічної бібліотеки «windows.h». Структура програми є багатовимірною.

Продукт курсової роботи має розважальний характер гри. Ця гра є однією з найперших комп'ютерних ігор, але попри це і досі залишається досить популярною.

Annotation

The submitted course work is an implementation of the console game program «Minesweeper». Puzzle game was created with using of the pseudographic elements on basis of the classic game «Minesweeper».

During writing course work, the programming language C++ was used, the graphic design was created with using of the graphic library «windows.h». The structure of the program is multidimensional.

The course work product has entertaining aim. This game is one of the first computer games, but it still remains popular.

Зміст

ВСТУП

У наш час одними з найнеобхідніших навичок, які потрібні сучасній людині є логічне мислення і вміння швидко приймати рішення. На мою думку, одними із найкращих способів розвитку цих здібностей є ігри-головоломки. Саме тому темою курсової роботи було обрано досить поширену і популярну ігрову програму «Сапер».

Мета курсової роботи полягає розробленні та реалізації закінченого програмного продукту на основі класичних правил гри «Сапер».

Задля досягнення поставленої мети, було виконано декілька завдань, а саме: на основі аналізу літературних джерел визначено основні риси майбутньої розробки, оцінення існуючих методів та алгоритмів реалізації програми, складено структурний алгоритм роботи майбутньої програми та реалізовано його у вигляді програмного продукту з використанням процедурного програмування.

При виконанні роботи було використане таке програмне забезпечення: середовище розробки Microsoft Visual Studio 2022; операційна система Windows 10; веб-браузер Google Chrome для роботи з веб-сайтами; текстовий редактор Microsoft Word 2013для підготовки та оформлення курсової роботи.

Робота складається зі вступу, трьох розділів, висновків, додатку та списку використаних джерел.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ

1.1. Огляд існуючих підходів до розв'язання поставленої задачі

Гра «Сапер» полягає в тому, що гравець має відкрити все поле не потрапивши на міну. Гравець відкриває клітинки намагаючись не потрапити на «замінovану» клітинку. Якщо під відкритою клітинкою немає міни то з'являється число, яке вказує на кількість мін навколо (рисунок 1.1). Для того щоб відмітити місце, де на вашу думку знаходиться міна потрібно поставити прапорець (рисунок 1.2).

Це означає, що, по-перше, потрібно створити генератор випадкових чисел для розстановки мін і алгоритм для підрахунку мін навколо «незамінованих» клітинок. По-друге, потрібно створити алгоритм для пересування по ігровому полю. По-третє, після відкриття всіх клітинок і правильної розтановки прапорців гра має закінчуватися.

«Сапер бере свій початок з найперших мейнфреймових ігор 1960-х і 1970-х років. Найдавнішим предком «Сапера» був «Cube» Джерімака Ретліффа. Основний стиль гри став популярним сегментом жанру відеоігор-головоломок у 1980-х роках з такими назвами, як Mined-Out (Quicksilva, 1983), Yomp (Virgin Interactive, 1983) і Cube.[7]

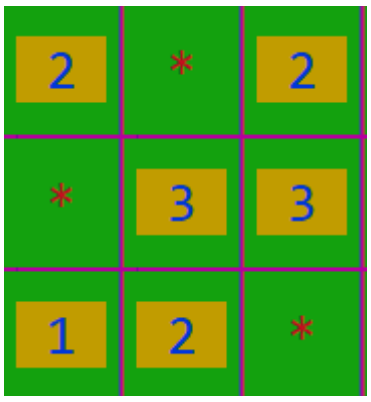

Рисунок 1.1

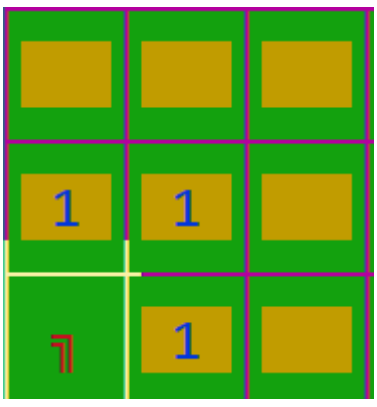

Рисунок 1.2

1.2. Уточнена постановка задачи на розробку програмного забезпечення

Основна задача полягає у створенні ігрової програми «Сапер» з використанням програмного забезпечення Microsoft Visual Studio 2022.

Проаналізуваши вже існуючі підходи до реалізації поставленої задачі, було обраний, на думку автора, найкращий і водночас інтуїтивно зрозумілий варіант гри «Сапер».

За основу береться двовимірний масив розміром 8x9 для легкого рівня, 13x15 для середнього і 17x21 для складного. Для масиву легкого рівня випадково генерується 10 чисел (35 і 75 для середнього та важкого рівнів відповідно) і обчислюється кількість «замінованих» клітинок навколо вільних.

Для перміщення по ігровому полю розроблено алгоритм для використання стрілок-клавіш. Для того щоб відкрити певну клітинку, користувач має натиснути Enter, а для установки прапорця потрібно натиснути Пробіл. Також є можливість покинути гру за допомогою клавіші Esc, після натиснення якої буде відображене меню, яке дає можливість покинути гру або залишитися в ній.

Окрім цього в головному меню гри передбачена можливість ознайомлення з правилами (Help), перегляд інформації про розробника гри (About) та виходу з гри.

РОЗДІЛ 2
РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Метод розв’язку задачі

Дана програма написана на мові програмування C++ з використанням бібліотеки «windows.h» для створення псевдографіки, а також допоміжних бібліотек «iostream», «cwchar», «conio.h», «time.h», «ctime».

Для зручного користування на початку гри розташоване головне меню у якому є 4 пункти:

1. Play
2. Help
3. About
4. Exit

Функції пунктів Help і About були описані в Розділі 1. Пункт Exit використовється для виходу з програми. Якщо вибрати пункт Play з’явиться нове меню з вибором складності (Easy, Medium Hard). Після вибору складності власне починається сама гра.

Як було зазначено у Розділі 1 основа гри двовимірний масив чисел, який у даній програмі знаходиться у створеній структурі saper, яка складається з полів цілочисельних close та open, в які відповідно записується присутність міни (число 10) або кількість мін навколо певної клітинки.

Для випадкового розашування мін генерується набір чисел, які не повторюються і не більше кількості клітинок поля. Потім визначається положення міни в двовимірному масиві і після цього підраховується кількість мін навколо пустих клітинок і все записується в двовимірний масив структур saper, а саме до поля close. На початку усі поля open дорівнюють нулю.

Після натискання Enter відкривається певна клітинка. Якщо клітинка межує з мінами, то виведеться кількість мін навколо (рисунок 2.1). Якщо навколо клітинки немає мін (close=0), то відкриваються клітинки навколо в усіх напрямках, поки не досягнуть клітинок, які вже межують з мінами

(рисунок 2.2). Якщо гравець потрапляє на міну, то все поле автоматично відкривається і гра завершується програшем.

Для того, щоб помітити клітинки, де, на думку гравця, розташовується міна, необхідно натиснути Пробіл і розмістити прапорець, як було зазначено в Розділі 1.

Гра завершується перемогою, коли гравець відкрив усі «безпечні» клітинки і правильно розставив прапорці.

Крім цього у грі передбачена можливість виходу з гри до її завершення за допомогою клавіші Esc і подальшим вибором пункту Yes.

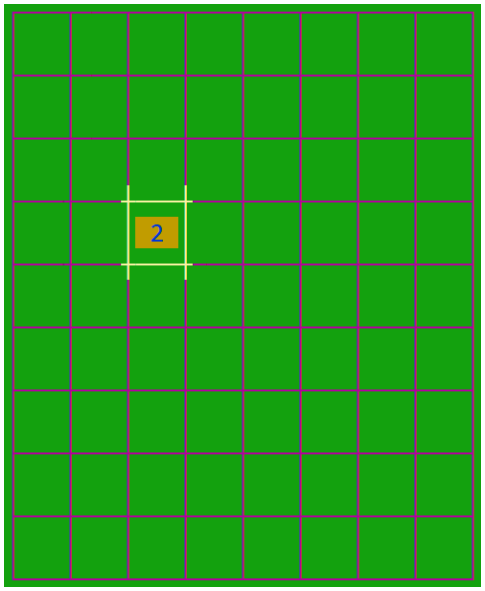

Рисунок 2.1

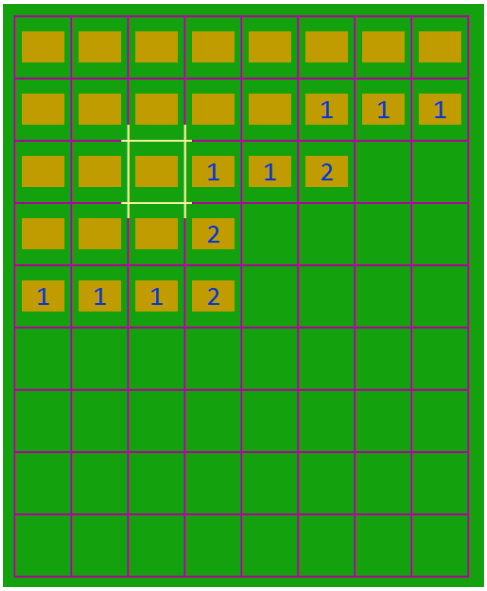

Рисунок 2.2

2.2. Алгоритм розв’язку задачі

Наступна блок-схема ілюструє між взаємозв’язки компонентами програми (рисунок 2.3)

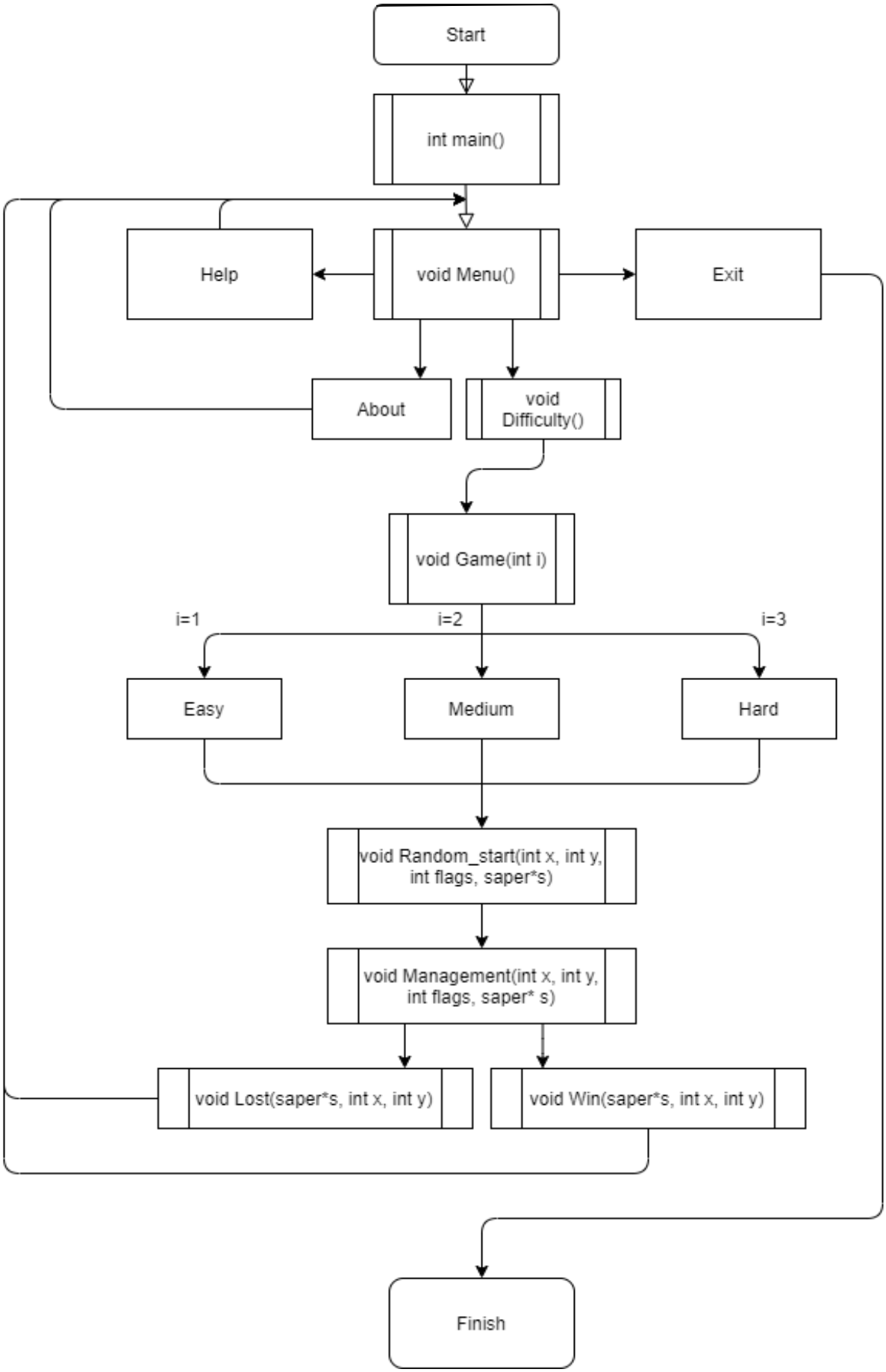

Рисунок 2.3. Структурна блок-схема алгоритму програми

РОЗДІЛ 3
ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ

Програма складається з файлу «saper.cpp», у якому знаходяться всі необхідні для роботи програми дані і фукції.

3.1. Опис головних структур і змінних програми

Таблиця 3.1.Опис головоних структур програми ілюструє усі структури , які використовувались при написанні коду програми.

Таблиця 3.1.Опис головоних структур програми

№	Тип	Опис та призначення
1	HANDLE	Структура дескриптора консольного вікна
2	COORD	Структура з полями X і Y, яка використовується для встановлення координат курсора
3	CONSOLE_SCREEN_BUFFER_INFO	Структура яка містить інформацію про буфер консольного вікна
4	CONSOLE_FONT_INFOEX	Структура яка містить інформацію про шрифт консольного вікна
5	time_t	Тип, який містить інформацію про час
6	saper	Користувацька структура з цілочисельними полями close та open, використовується для збереження інформації в процесі гри

Таблиця 3.2.Опис головних змінних програми ілюструє найважливіші змінні, використовувались при написанні коду програми.

Таблиця 3.2.Опис головних змінних програми

№	Змінна	Тип	Опис та призначення
1	hStdOut	HANDLE	Дескриптор консольного вікна
2	x	const int	Оголошується в функції Game() для кожної складності окремо, визначає кількість стовбчиків ігрового поля
3	y	const int	Оголошується в функції Game() для кожної складності окремо, визначає кількість рядків ігрового поля
4	flags	const int	Оголошується в функції Game() для кожної складності окремо, визначає кількість прапорців і мін
5	s[x][y]	saper	Оголошується в функції Game() для кожної складності окремо (з різною розмірністю), зберігає інформацію в процесі гри
6	t1	time_t	Зберігає час першого натиснення Enter у грі, завдяки чому вираховується час
7	w1	COORD	Використовується у багатьох функціях для встановлення положення курсора
8	cfi1	CONSOLE_FONT_INFOEX	Використовується для зміни шрифту консольного вікна
9	csbuf	CONSOLE_SCREEN_BUFFER_INFO	Використовується для отримання інформації про буфер консольного вікна
10	i1	int	Використовується для руху по ігровому полю в функції Management
11	i2	int	Використовується для руху по ігровому полю в функції Management

3.2. Опис головних функцій програми

У таблиці 3.3.Основні функції програми наведені функції, що використовуються під час роботи програми.

Таблиця 3.3.Основні функції програми

№	Синтаксис	Опис
1	void Menu()	Функція визначає роботу головного меню гри, а також пунктів Help, About, Exit.
2	void Difficulty()	Викликається після вибору пункту Play головного меню, завдяки ній відбувається вибір складності.
3	void ChangeFont(int x, int y, int w)	Використовується для зміни розміру шрифту вікна консолі. Викликається у функціях Menu(), Game(int a). Як параметри вводяться ширина, висота та жирність шрифту.
4	void Printmenu(int attr, COORD w, char* word)	Використовується для виведення пункті меню. Як параметри вводяться атрибут, координати, куди потрібно перемістити курсор і назва пункту
5	void Game(int a)	У цій функції створюються основні змінні необхідні для гри відповідно складності (за яку відповідає параметр int a). Викликається з функції Difficulty() після вибору гравцем складності.
6	void Field(int xmax, int ymax)	Ця функція виводить на екран ігрове поле розмірами xmax на ymax. Викликається з функції Game(int a).

7	void Random_start(int x, int y, int flags, saper*s)	Ця функція використовується для випадкової генерації розташування мін на полі.
8	void Management(int x, int y, int flags, saper* s)	Одна з основних функцій програми. Завдяки ній відбуваються основні процеси управління грою: переміщення по ігровому полю, виведення тамера та кількості прапорців, відкриття полів, встановлення прапорців, вихід з гри. Ця функція викликається з Game(int a), як параметри передаються дані визначені складністю та уже голошений масив структур з «розтавленими мінами»
9	void myclock(time_t t1)	Функція, яка відповідає за таймер. Як параметр передається час першого відкриття клітинки.
10	void Open(int i1, int i2,int x, int y ,saper* s)	Функція, яка відкриває клітинку, яка межує з мінами. Як параметри передається координати поточної клітинки (i1, i2), розміри поля (x, y), та масив структур.
11	void Open0(int i1, int i2, int x, int y,saper* s)	Функція, яка відкриває клітинку, яка не межує з мінами. Вона є рекурсивною, оскільки викликається для сусідніх клітинок, поки ті не будть межувати з мінами. Параметри аналогічні параметрам функції Open.

12	void Lost(saper*s, int x, int y)	Функція, яка викликається після потрапляння на міну. Відкриває все поле включно з мінами і виводить повідомлення про програш. Через 5 секунд викликається функція Menu().
13	void Win(saper* s, int x, int y)	Функція викликається, коли залишилось 0 прапорців. Перевіряє чи правильно стоять усі прапорці і чи відкриті усі клітинки. Якщо все правильно, то виводиться відповідне повідомлення і через 5 секунд викликається функція Menu().

3.3. Опис інтерфейсу

Коли користувач запускає гру, він бачить перед собою саму назву гри та головне меню, як зображено на рисунку 3.1. Гра оформлена анлійською мовою.

Рисунок 3.1. Головне меню

Якщо вибрати пункт Help, то гравець зможе побачити правила гри (рисунок 3.2).

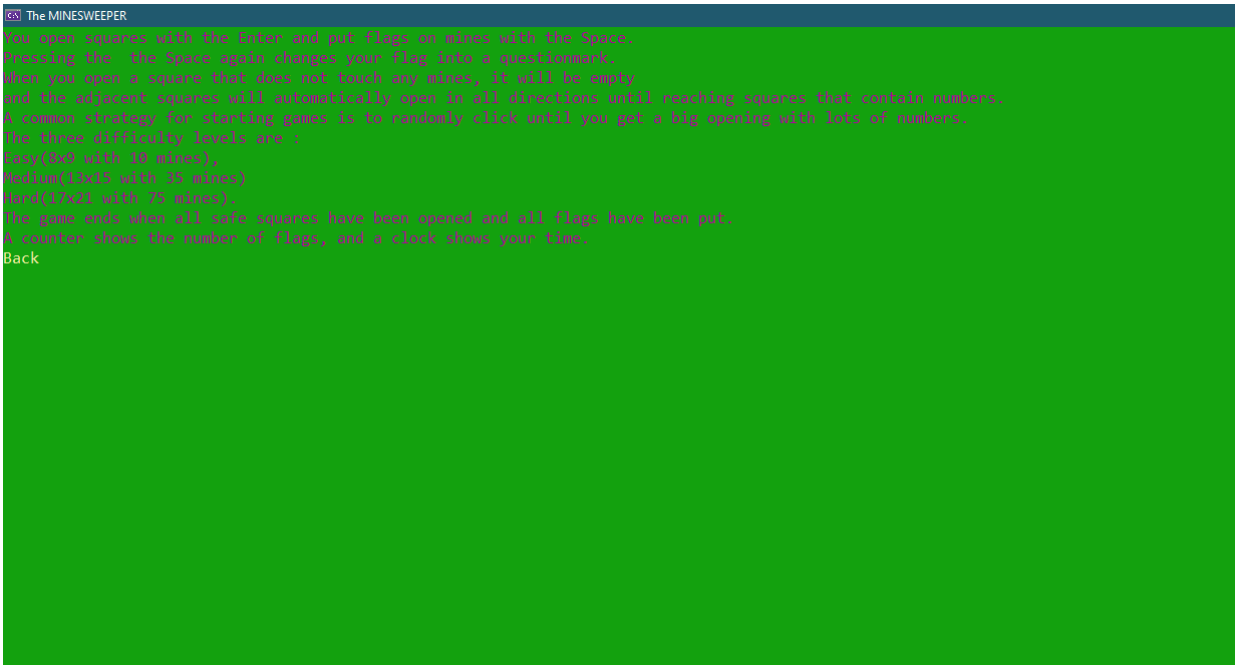

Рисунок 3.2. Пункт Help

Після вибору пункту About, буде показана інформація про автора гри (рисунок 3.3).

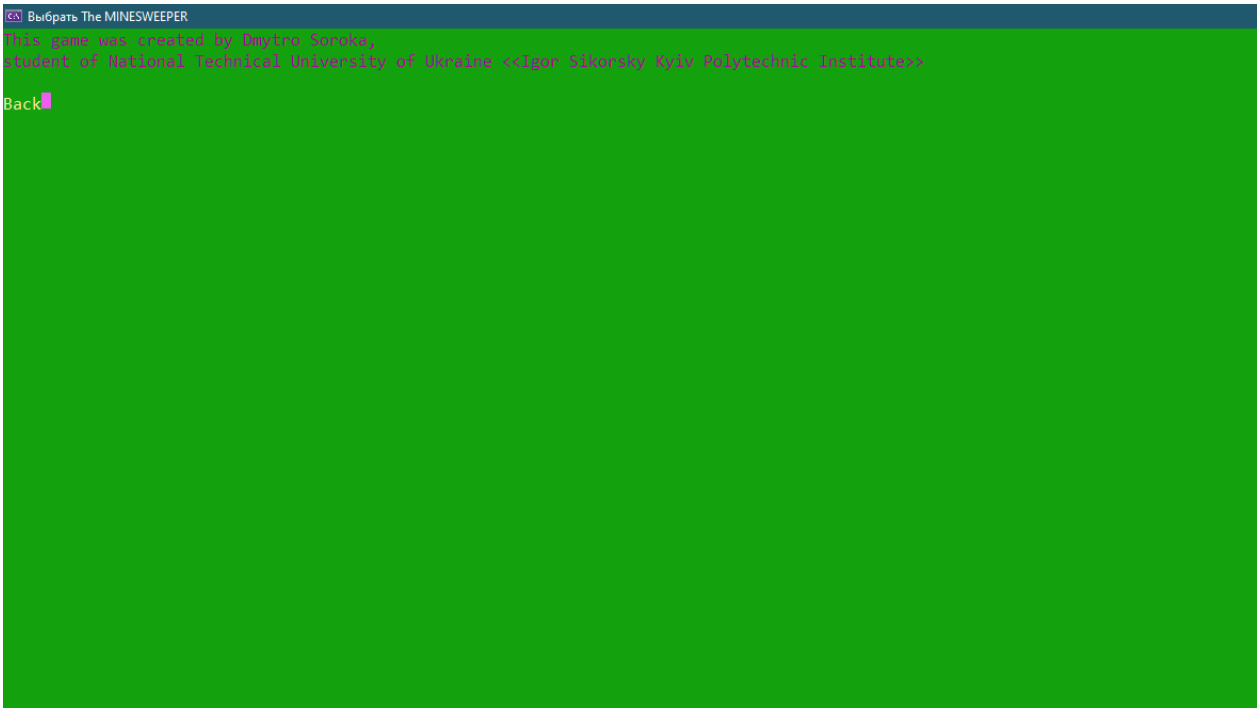

Рисунок 3.3. Пункт About

Після вибору пункту Play на екрані з’явиться меню з виобором складності гри (рисунок 3.4).

Рисунок 3.4. Вибір складності

В залежності від вибору складнасті на екрані з’являться поля різного розміру (Easy - рисунок 3.5 , Medium - рисунок 3.6, Hard - рисунок 3.7)

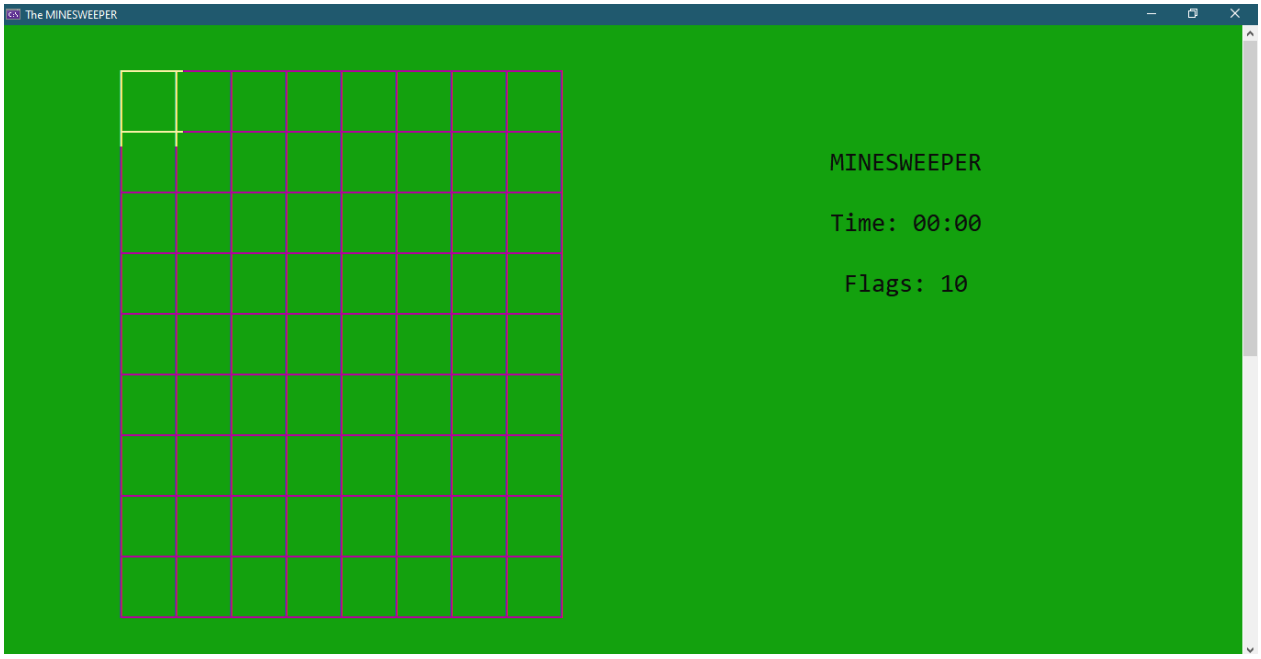

Рисунок 3.5. Складність Easy

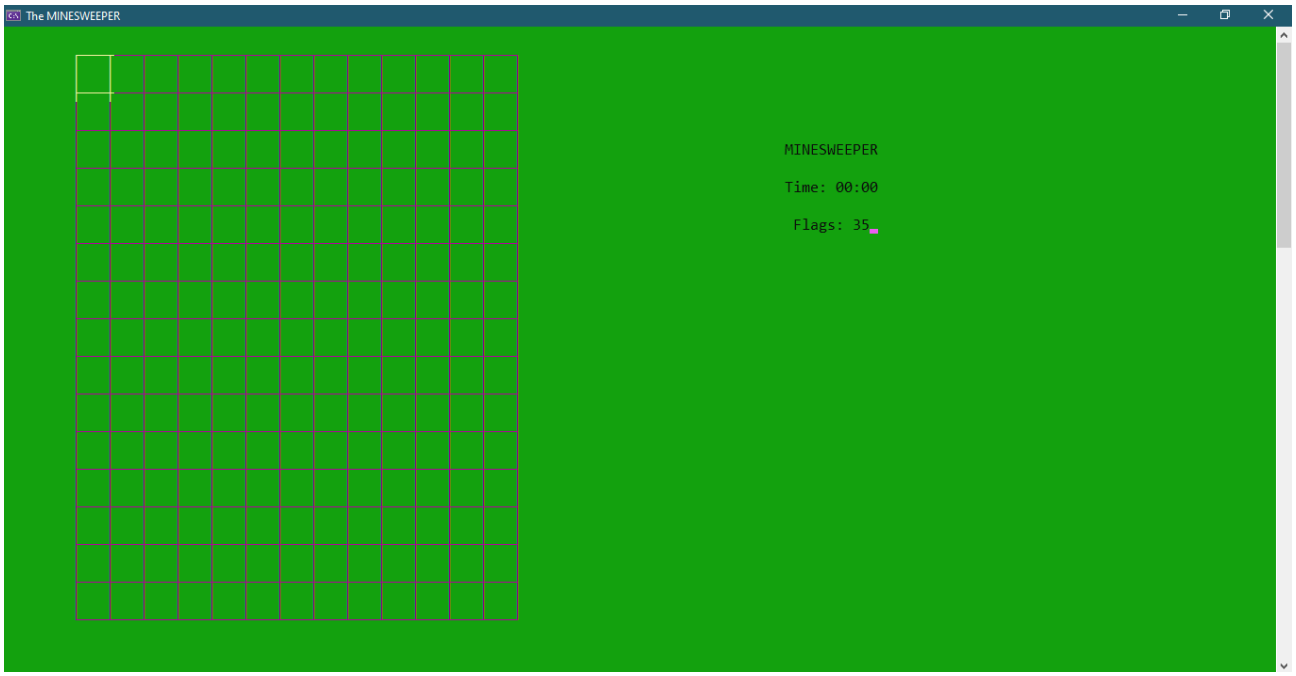

Рисунок 3.6. Складність Medium

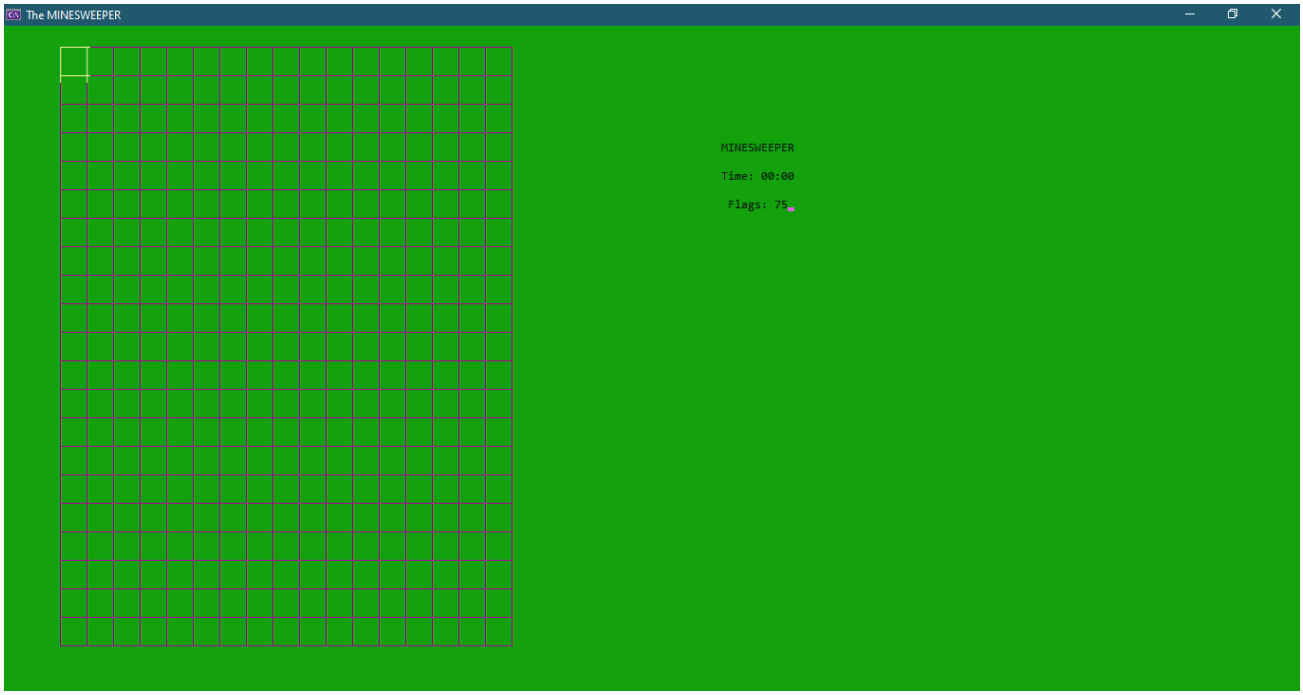

Рисунок 3.5. Складність Hard

Для того щоб почати гру, необхідно натиснути Enter. Ви можете потрапити на пусту клітинку, з числом або з міною. У разі попадання на «замінovану» клітинку, відкриється все поле і виведеться повідомлення про поразку, як показано на рисунку 3.6

Рисунок 3.6. Поразка

Для того щоб поставити прапорець там, де на вашу думку знаходиться міна потрібно натуснити Пробіл. Ось що виведеться на екран(рисунок 3.7).

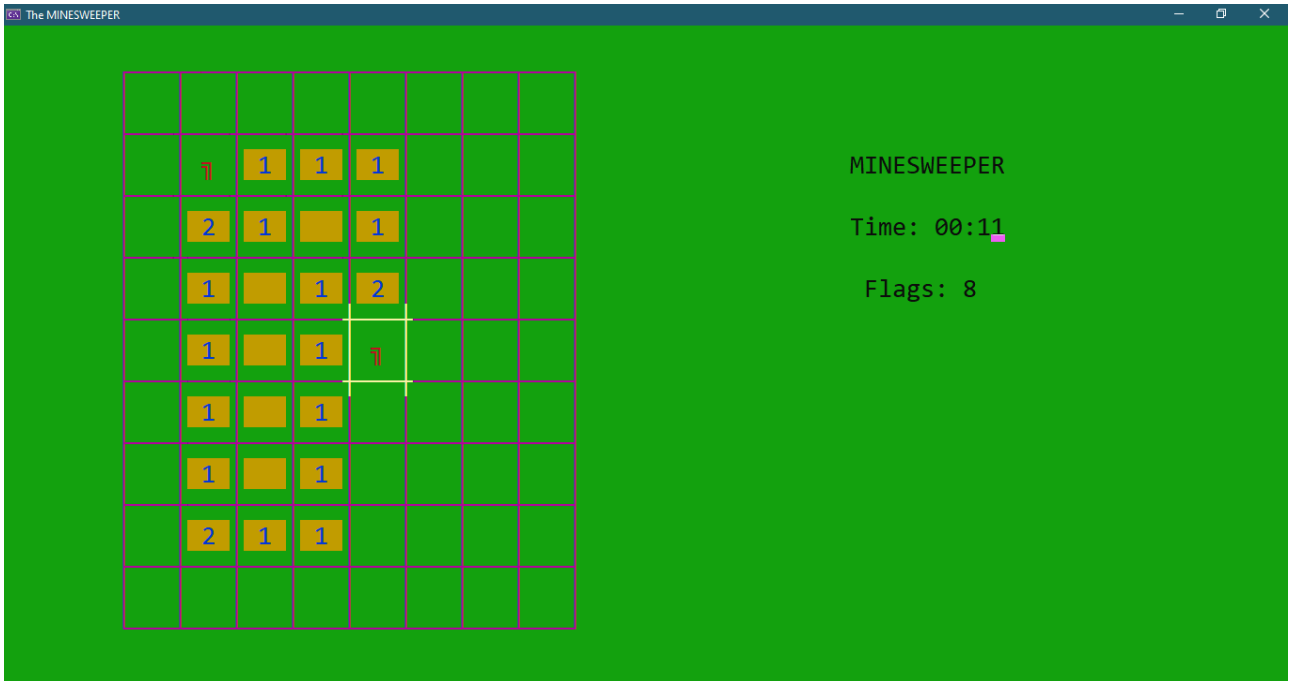

Рисунок 3.7. Прапорці

Для того щоб перемогти, необхідно правильно розставити прапорці і відкрити всі «безпечні» клітинки. Якщо все зроблено правильно, зупиниться таймер і на екран буде виведено повідослення про перемогу (рисунок 3.8).

Рисунок 3.8. Перемога

Також під час гри є можливість вийти з неї за допомогою клавіші Esc. Після її натиснення буде виведене відповідне меню (рисунок 3.9).

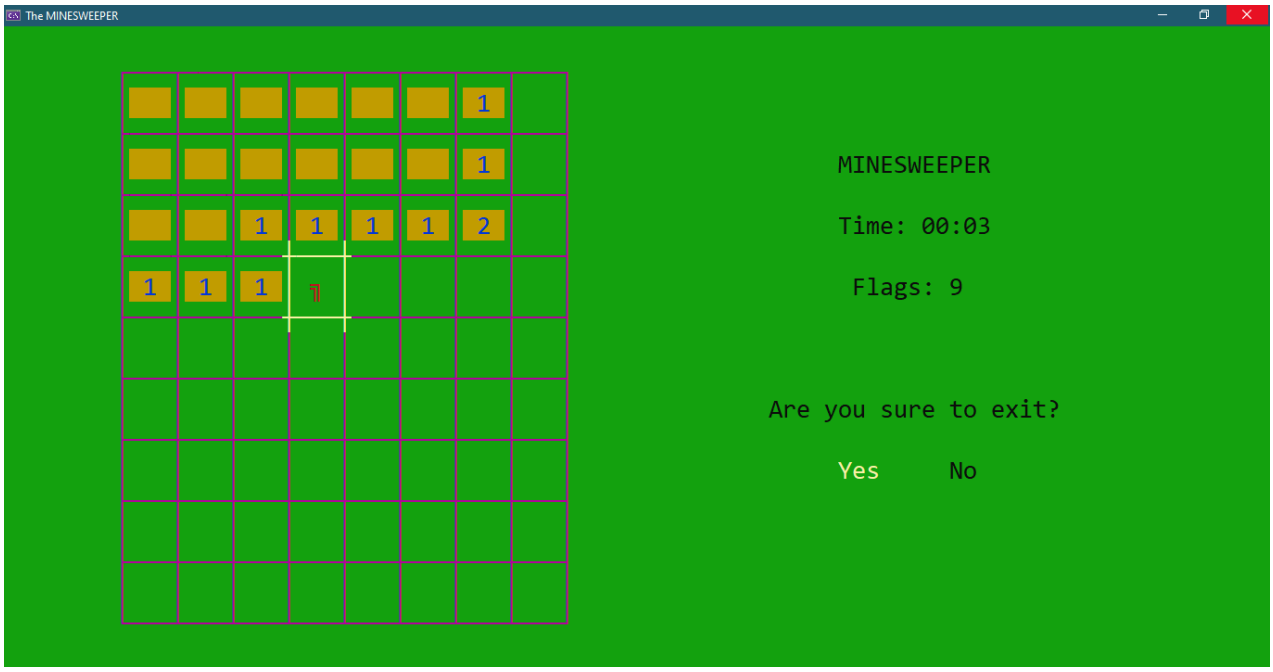

Рисунок 3.9. Клавіша Esc

3.4. Результати роботи програмного продукту

Після перемоги на екрані з’явиться відповідне повідомлення, як показано на рисунку 3.8. Після цього з п’ятисекундною затримкою на екрані з’явиться Головне меню (рисунок 3.1).

ВИСНОВКИ

У ході реалізації теми курсової роботи було розроблено програмний продукт, який у повному обсязі відповідає поставленому технічному завданню. Створена ігрова програма «Сапер» дозволяє з користю та інтересом проводити вільний час, розвиваючи логічні здібності і швидкість прийняття рішень.

Гра була розроблена за допомогою мови C++ з використанням графічної бібліотеки <windows.h> та інших. Автору повністю вдалося відворити класичну гру «Сапер» за стандартними правилами.

Для створення і розробки програми було використано середовище Microsoft Visual Studio 2022.

Перевагами представленої версії гри інтуїтивно зрозуміла і продумана механіка гри, просте і зручне управління, можливість вибору рівня складності.

Одним з головних недоліків гри є, на мою думку, одноманітність гри. В цю гру не можна грати багато часу, оскільки через досить короткий проміжок часу вона починає набридати.

Зрозуміло, що створену програму є куди покращувати. Зокрема можна зробити можливість першого «безпечного» ходу, тобто гравець відкриваючи першу клітинку потрапляє на клітинку навколо якої не знаходяться міни. Також можна модернізувати гру, зробивши клітинки, наприклад, не квадратними, а шестикутними. Більш того, гру взагалі можна зробити трьохвимірною.

В майбутньому автор має план, перш за все, прибрати, якщо такі будуть, недоліки і помилки, які потенційно можуть виявитися під час довгого користування. Також одним з головних пріорітетів покращення гри є розробка вищеозазначеної можливості першого «безпечного» ходу гравцем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Назарчук І.В., Селін О.М., Швачко Г.Г. Програмування та алгоритмічні мови програмування: Методичні вказівки до виконання курсового проекту галузь знань 12 інформаційні технології 122 «Комп'ютерні науки та інформаційні технології» 124 «Системний аналіз». Київ «ІПСА» НТУУ "КПІ" 2017
2. Назарчук І.В. Програмування та алгоритмічні мови 1: Алгоритмізація та основи програмування конспект лекцій. КПІ ім. Ігоря Сікорського, 2020
3. Васильєв О. Програмування С++ в прикладах і задачах. Ліра-К, 2017
4. З. Шпак Програмування мовою С. Львівська політехніка, 2011
5. Г. Шилдт С++ класичне видання. Вільямс, 2019
6. Minesweeper (video game) URL: https://en.wikipedia.org/wiki/Minesweeper_(video_game) (дата звернення: 02.05.2022).
7. How To Play Minesweeper URL: https://minesweepergame.com/strategy/how-to-play-minesweeper.php (дата звернення: 02.05.2022).
8. Операции управления временем URL: https://docs.microsoft.com/ru-ru/cpp/c-runtime-library/time-management?view=msvc-170 (дата звернення: 29.05.2022).
9. Как поменять размер шрифта в консоли c++? URL: https://ru.stackoverflow.com/questions/243596/%D0%9A%D0%B0%D0%BA-%D0%BF%D0%BE%D0%BC%D0%B5%D0%BD%D1%8F%D1%82%D1%8C-%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%80-%D1%88%D1%80%D0%B8%D1%84%D1%82%D0%B0-%D0%B2-%D0%BA%D0%BE%D0%BD%D1%81%D0%BE%D0%BB%D0%B8-%D1%81 (дата звернення: 30.05.2022).

10. ASCII Table URL: https://www.asciitable.com/ (дата звернення: 30.05.2022).
11. Как изменить цвет для окна консоли c++ URL:https://ru.stackoverflow.com/questions/1141821/%D0%9A%D0%B0%D0%BA-%D0%B8%D0%B7%D0%BC%D0%B5%D0%BD%D0%B8%D1%82%D1%8C-%D1%86%D0%B2%D0%B5%D1%82-%D0%B4%D0%BB%D1%8F-%D0%BE%D0%BA%D0%BD%D0%B0-%D0%BA%D0%BE%D0%BD%D1%81%D0%BE%D0%BB%D0%B8-c (дата звернення: 01.06.2022)

ДОДАТОК А

Текст програми

```
#include <iostream>
#include <windows.h>
#include <cwchar>
#include <conio.h>
#include <time.h>
#include <ctime>

using namespace std;

HANDLE hStdOut = GetStdHandle(STD_OUTPUT_HANDLE);

typedef struct {
   int close;
   int open;
}saper;

void Field(int , int);
void Game(int);
void ChangeFont(int , int , int );
void Printmenu(int , COORD , char* );
void Difficulty();
void Menu();
void Random_start(int , int ,const int, saper*);
void Management(int,int, int, saper*);
void Open(int, int, int, int, saper*);
void Open0(int, int, int, int, saper*);
void myclock(time_t);
void Lost(saper*, int, int);
void Win(saper*, int, int);
int main()
{
   SetConsoleTitle(L"The MINESWEEPER ");
   Menu();
}
void Game(int a) {
   int i1=0, i2=0, q=1;
   system("cls");
   SetConsoleTextAttribute(hStdOut, 45);
   switch (a) {
```

```
case 1: {
   ChangeFont(90, 33, 500);
   system("mode con cols=80 lines=21");
   Field(8, 19);
   char a;
   const int x = 8, y = 9, flags = 10;
   COORD w{1, 1};
   saper s[x][y]{};
   saper* b = &s[0][0];
   Random_start (x, y, flags, b);
   i1 = 0, i2 = 0;
   while (i1 < y) {
      i2 = 0;
      while (i2 < x) {
         if (s[i2][i1].close == 10) {
            if (i2 > 0 && i1 > 0&& s[i2 - 1][i1 - 1].close !=10)
               (s[i2-1][i1-1]).close++;
            if (i1 > 0 && s[i2][i1 - 1].close != 10)
               (s[i2][i1 - 1]).close++;
            if (i2 < x - 1 && i1 > 0 && s[i2 + 1][i1 - 1].close != 10)
               (s[i2 + 1][i1 - 1]).close++;
            if (i2 < x - 1 && s[i2 + 1][i1].close != 10)
               (s[i2 + 1][i1]).close++;
            if (i2 <x - 1 && i1 < y - 1 && s[i2 + 1][i1 + 1].close != 10)
               (s[i2+1][i1 + 1]).close++;
            if (i1 < y - 1 && s[i2][i1 + 1].close != 10)
               (s[i2][i1 + 1]).close++;
            if (i2>0 && i1 < y - 1 && s[i2 - 1][i1 + 1].close != 10)
               (s[i2 - 1][i1 + 1]).close++;
            if (i2 > 0 && s[i2 - 1][i1].close != 10)
               (s[i2 - 1][i1]).close++;
         }
         i2++;
      }
      i1++;
   }
   i1 = 0, i2 = 0;
   Management(x, y, flags, b);
   break;

}
case 2: {
```

```
    ChangeFont(57, 20, 500);
    system("mode con cols=120 lines=33");
    Field(13, 31);
    const int x = 13, y = 15, flags = 35;
    saper s[x][y]{};
    saper* b = &s[0][0];
    Random_start(x, y, flags, b);
    i1 = 0, i2 = 0;
    while (i1 < y) {
        i2 = 0;
        while (i2 < x) {
            if (s[i2][i1].close == 10) {
                if (i2 > 0 && i1 > 0 && s[i2 - 1][i1 - 1].close != 10)
                    (s[i2 - 1][i1 - 1]).close++;
                if (i1 > 0 && s[i2][i1 - 1].close != 10)
                    (s[i2][i1 - 1]).close++;
                if (i2 < x - 1 && i1 > 0 && s[i2 + 1][i1 - 1].close != 10)
                    (s[i2 + 1][i1 - 1]).close++;
                if (i2 < x - 1 && s[i2 + 1][i1].close != 10)
                    (s[i2 + 1][i1]).close++;
                if (i2 < x - 1 && i1 < y - 1 && s[i2 + 1][i1 + 1].close != 10)
                    (s[i2 + 1][i1 + 1]).close++;
                if (i1 < y - 1 && s[i2][i1 + 1].close != 10)
                    (s[i2][i1 + 1]).close++;
                if (i2 > 0 && i1 < y - 1 && s[i2 - 1][i1 + 1].close != 10)
                    (s[i2 - 1][i1 + 1]).close++;
                if (i2 > 0 && s[i2 - 1][i1].close != 10)
                    (s[i2 - 1][i1]).close++;
            }
            i2++;
        }
        i1++;
    }
    i1 = 0, i2 = 0;
    Management(x, y, flags, b);
    break;
}
case 3: {
    ChangeFont(41, 15, 500);
    system("mode con cols=136 lines=45");
    Field(17, 43);
    const int x = 17, y = 21, flags = 75;
```

```
        saper s[x][y]{};
        saper* b = &s[0][0];
        Random_start (x, y, flags, b);
        i1 = 0, i2 = 0;
        while (i1 < y) {
          i2 = 0;
          while (i2 < x) {
            if (s[i2][i1].close == 10) {
              if (i2 > 0 && i1 > 0&& s[i2 - 1][i1 - 1].close !=10)
                (s[i2-1][i1-1]).close++;
              if (i1 > 0 && s[i2][i1 - 1].close != 10)
                (s[i2][i1 - 1]).close++;
              if (i2 < x - 1 && i1 > 0 && s[i2 + 1][i1 - 1].close != 10)
                (s[i2 + 1][i1 - 1]).close++;
              if (i2 < x - 1 && s[i2 + 1][i1].close != 10)
                (s[i2 + 1][i1]).close++;
              if (i2 <x - 1 && i1 < y - 1 && s[i2 + 1][i1 + 1].close != 10)
                (s[i2+1][i1 + 1]).close++;
              if (i1 < y - 1 && s[i2][i1 + 1].close != 10)
                (s[i2][i1 + 1]).close++;
              if (i2>0 && i1 < y - 1 && s[i2 - 1][i1 + 1].close != 10)
                (s[i2 - 1][i1 + 1]).close++;
              if (i2 > 0 && s[i2 - 1][i1].close != 10)
                (s[i2 - 1][i1]).close++;
            }
            i2++;
          }
          i1++;
        }
        i1 = 0, i2 = 0;
        Management(x, y, flags, b);
        break;
    }
    }
}
void ChangeFont(int x, int y, int w) {
    CONSOLE_FONT_INFOEX cfi1;
    cfi1.cbSize = sizeof(cfi1);
    GetCurrentConsoleFontEx(hStdOut, TRUE, &cfi1);
    cfi1.dwFontSize.X = x;
    cfi1.dwFontSize.Y = y;
    cfi1.FontWeight = w;
```

```
    SetCurrentConsoleFontEx(hStdOut, FALSE, &cfi1);
};
void Printmenu(int attr, COORD w, char* word) {
    SetConsoleTextAttribute(hStdOut, attr);
    SetConsoleCursorPosition(hStdOut, w);
    cout << word;
};
void Difficulty() {
    int sizex, sizey, i = 0, q = 1;;
    COORD w, w1[4];
    CONSOLE_SCREEN_BUFFER_INFO csbuf;
    char m[4][7] = { "Easy", "Medium","Hard","Back" }, ch;
    GetConsoleScreenBufferInfo(hStdOut, &csbuf);
    sizex = csbuf.srWindow.Right;
    sizey = csbuf.srWindow.Bottom;
    w.X = sizex / 2 - 8;
    w.Y = sizey / 5;
    system("cls");
    SetConsoleTextAttribute(hStdOut, 45);
    SetConsoleCursorPosition(hStdOut, w);
    cout << "CHOOSE DIFFICULTY";
    w1[0].X = sizex / 2 - 2;
    w1[0].Y = 2 + w.Y;
    w1[1].Y = w1[0].Y + 2;
    w1[1].X = sizex / 2 - 3;
    w1[2].X = w1[0].X;
    w1[2].Y = w1[1].Y + 2;
    w1[3].X = w1[0].X;
    w1[3].Y = w1[2].Y + 2;
    Printmenu(46, w1[0], m[0]);
    Printmenu(45, w1[1], m[1]);
    Printmenu(45, w1[2], m[2]);
    Printmenu(45, w1[3], m[3]);
    while (q) {
        if (_kbhit()) {
            ch = _getch();
            switch (ch) {
            case 72: {
                if (i != 0) {
                    Printmenu(45, w1[i], m[i]);
                    Printmenu(46, w1[i - 1], m[i - 1]);
                    i--;
```

```
            }
            else {
               Printmenu(45, w1[i], m[i]);
               Printmenu(46, w1[3], m[3]);
               i = 3;
            }
            break;
         }
         case 80: {
            if (i != 3) {
               Printmenu(45, w1[i], m[i]);
               Printmenu(46, w1[i + 1], m[i + 1]);
               i++;
            }
            else {
               Printmenu(45, w1[i], m[i]);
               Printmenu(46, w1[0], m[0]);
               i = 0;
            }
            break;
         case 13: {
            if (i != 3) Game(i + 1);
            else Menu();
            q = 0;
            break;
         }
         }
         }
         }
      }
};
void Menu() {
   COORD a1, a2, w{}, w1{}, w2{}, w3{}, w4{};
   int sizex, sizey, q = 1, i = 0, q1 = 1;
   char ch;
   char m[4][6] = { "Play","Help", "About", "Exit" };
   SetConsoleTextAttribute(hStdOut, 32);
   system("cls");
   SetConsoleTextAttribute(hStdOut, 45);
   ChangeFont(140, 50, 600);
   system("mode con cols=65 lines=11");
   CONSOLE_SCREEN_BUFFER_INFO csbuf;
```

```
GetConsoleScreenBufferInfo(hStdOut, &csbuf);
sizex = csbuf.srWindow.Right;
sizey = csbuf.srWindow.Bottom;
w.X = sizex / 2 - 5;
w.Y = sizey / 5;
SetConsoleCursorPosition(hStdOut, w);
cout << "MINESWEEPER";
w1.X = sizex / 2 - 2;
w1.Y = 2 + w.Y;
w2.Y = w1.Y + 2;
w2.X = w1.X;
w3.Y = w2.Y + 2;
w4.Y = w3.Y + 2;
w3.X = w1.X;
w4.X = w1.X;
Printmenu(46, w1, m[0]);
Printmenu(45, w2, m[1]);
Printmenu(45, w3, m[2]);
Printmenu(45, w4, m[3]);
while (q) {
   if (_kbhit()) {
      ch = _getch();
      switch (i) {
      case 0: {switch (ch) {
         ;
      case 72: {
         Printmenu(45, w1, m[0]);
         Printmenu(46, w4, m[3]);
         i = 3;
         break;
      }
      case 80: {
         Printmenu(45, w1, m[0]);
         Printmenu(46, w2, m[1]);
         i++;
         break; }
      case 13: {Difficulty(); q = 0; break; }
      default: break;
      }
          break; }
      case 3: {switch (ch) {
      case 72: {
```

```
            Printmenu(46, w3, m[2]);
            Printmenu(45, w4, m[3]);
            i--;
            break;
        }
        case 80: {
            Printmenu(46, w1, m[0]);
            Printmenu(45, w4, m[3]);
            i=0;
            break; }

        case 13: {q = 0;  SetConsoleTextAttribute(hStdOut, 45); return void(); }
        default:break;
        }
             break;
        }
        case 1: {switch (ch) {
        case 72: {
            Printmenu(45, w2, m[1]);
            Printmenu(46, w1, m[0]);
            i--;
            break;
        }
        case 80: {
            Printmenu(45, w2, m[1]);
            Printmenu(46, w3, m[2]);
            i++;
            break; }
        case 13: {
            system("cls");
            ChangeFont(57, 20, 500);
            system("mode con cols=120 lines=33");
            SetConsoleTextAttribute(hStdOut, 45);
            cout << "You open squares with the Enter and put flags on mines with the Space.\nPressing
the  the Space again changes your flag into a questionmark. \nWhen you open a square that does not
touch any mines, it will be empty \nand the adjacent squares will automatically open in all directions
until reaching squares that contain numbers.\nA common strategy for starting games is to randomly
click until you get a big opening with lots of numbers.\nThe three difficulty levels are : \nEasy(8x9 with
10 mines), \nMedium(13x15 with 35 mines)\nHard(17x21 with 75 mines).\nThe game ends when all
safe squares have been opened and all flags have been put.\nA counter shows the number of flags, and a
clock shows your time.";
            SetConsoleTextAttribute(hStdOut, 46);
```

```
            cout << "\nBack";
            q1 = 1;
            while (q1) {
               if (_kbhit()) {
                  ch = _getch();
                  if (ch == 13) {
                     q = 0;
                     q1 = 0;
                     Menu();

                  }
               }
            }
            break; }
         default: break;
         }
             break;
         }
         case 2: {switch (ch) {
         case 72: {
            Printmenu(45, w3, m[2]);
            Printmenu(46, w2, m[1]);
            i--;
            break;
         }
         case 80: {
            Printmenu(45, w3, m[2]);
            Printmenu(46, w4, m[3]);
            i++;
            break; }
         case 13: {
            system("cls");
            ChangeFont(57, 20, 500);
            system("mode con cols=120 lines=33");
            SetConsoleTextAttribute(hStdOut, 45);
            cout << "This game was created by Dmytro Soroka,\nstudent of National Technical
University of Ukraine <<Igor Sikorsky Kyiv Polytechnic Institute>>\n";
            SetConsoleTextAttribute(hStdOut, 46);
            cout << "\nBack";
            q1 = 1;
            while (q1) {
               if (_kbhit()) {
```

```
                ch = _getch();
                if (ch == 13) {
                    q1 = 0;
                    q = 0;
                    Menu();
                }
            }
        }
        break; }
    default: break;
    }
    break; }
    default:break;
    }
    }
  }
}
void Field(int xmax, int ymax) {
  int q = 1, i1 = 1, i = 1;;
  char c1 = 218, c2 = 191, c3 = 217, c4 = 192, c5 = 179, c6 = 196, c7 = 195, c8 = 194, c9 = 193, c10 =
197, c11 = 180;
  cout << "\n       " << c1;
  while (i < xmax) {
    cout << c6 << c6 << c6 << c8;
    i++;
  }
  cout << c6 << c6 << c6 << c2 << endl << "       ";
  i = 2;
  while (i < ymax) {
    if (i % 2)
      cout << c7;
    else
      cout << c5;
    while (i1 < xmax) {
      if (i % 2)
        cout << c6 << c6 << c6 << c10;
      else
        cout << "   " << c5;
        i1++;
    }
    if (i % 2)
      cout << c6 << c6 << c6 << c11 << endl << "       ";
```

```
        else
            cout << "   " << c5 << endl << "        ";
        i1 = 1;
        i++;
    }
    i = 1;
    cout << c4;
    while (i < xmax) {
        cout << c6 << c6 << c6 << c9;
        i++;
    }
    cout << c6 << c6<< c6 << c3;
}
void Random_start(int x, int y, int flags, saper*s) {
    const int b = flags;
    int i = 0, j = 0, k = 0, arr[75], a = 0;
    srand(time(NULL));
    while (i < flags) {
    here:
        j = 0;
        a = rand() % (x * y );
        while (j < i) {
            if (a == arr[j])
                goto here;
            j++;
        }
        arr[i] = a;
        (*(s + a)).close =10;
        i++;
    }
}
void Management(int x, int y, int flags, saper* s) {
    time_t t1=time(0);
    saper* b = s;
    COORD w1;
    int i1 = 0, i2 = 0, q = 1, i = 0, sizex, sizey,j=0, nflags=flags,q1=1, q2=1;
    char c1 = 218, c2 = 191, c3 = 217, c4 = 192, c5 = 179, c6 = 196, c7 = 195, c8 = 194, c9 = 193, c10 =
197, c11 = 180;
    char ch, fl = 443;
    w1.X = 8 + i1 * 5;
    w1.Y = 1 + i2 * 3;
    SetConsoleTextAttribute(hStdOut, 46);
```

```
SetConsoleCursorPosition(hStdOut, w1);
cout << c1 << c6 << c6 << c6 << c8;
w1.Y++;
SetConsoleCursorPosition(hStdOut, w1);
cout << c5;
w1.X += 4;
SetConsoleCursorPosition(hStdOut, w1);
cout << c5;
w1.X -= 4;
w1.Y++;
SetConsoleCursorPosition(hStdOut, w1);
cout << c7 << c6 << c6 << c6 << c10;
CONSOLE_SCREEN_BUFFER_INFO csbuf;
GetConsoleScreenBufferInfo(hStdOut, &csbuf);
sizex = csbuf.srWindow.Right;
sizey = csbuf.srWindow.Bottom;
w1.X = sizex / 5 *4;
w1.Y = sizey / 5;
SetConsoleTextAttribute(hStdOut, 32);
SetConsoleCursorPosition(hStdOut, w1);
cout << "MINESWEEPER";
w1.Y+=2;
SetConsoleCursorPosition(hStdOut, w1);
cout << "Time: 00:00";
w1.X += 1;
w1.Y+=2;
SetConsoleCursorPosition(hStdOut, w1);
cout << "Flags: "<<flags;
while (q) {
   if (_kbhit()) {
      ch = _getch();
      i = 0;
      if(ch!=13&&ch!=27 && ch != 32)
      while (i < 2) {
         if(i==0)
            SetConsoleTextAttribute(hStdOut, 45);
         else
            SetConsoleTextAttribute(hStdOut, 46);
         w1.X = 8 + i1 * 4;
         w1.Y = 1 + i2 * 2;
         if ((0 < i1&&i1 < x-1) && (0 < i2&& i2 < y-1)) {
            SetConsoleCursorPosition(hStdOut, w1);
```

```
    cout << c10 << c6 << c6 << c6 << c10;
    w1.Y++;
    SetConsoleCursorPosition(hStdOut, w1);
    cout << c5;
    w1.X += 4;
    SetConsoleCursorPosition(hStdOut, w1);
    cout << c5;
    w1.X -= 4;
    w1.Y++;
    SetConsoleCursorPosition(hStdOut, w1);
    cout << c10 << c6 << c6 << c6 << c10;
    goto end;
  }
  if (i1 == 0) {
    if ((i1 == 0) && (i2 == y-1 )) {
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c7 << c6 << c6 << c6 << c10;
      w1.Y++;
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c5;
      w1.X += 4;
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c5;
      w1.X -= 4;
      w1.Y++;
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c4 << c6 << c6 << c6 << c9;
      goto end;
    }
    if ((i1 == 0) && (i2 == 0)) {
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c1 << c6 << c6 << c6 << c8;
      w1.Y++;
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c5;
      w1.X += 4;
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c5;
      w1.X -= 4;
      w1.Y++;
      SetConsoleCursorPosition(hStdOut, w1);
      cout << c7 << c6 << c6 << c6 << c10;
```

```
            goto end;
        }
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c7 << c6 << c6 << c6 << c10;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X += 4;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X -= 4;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c7 << c6 << c6 << c6 << c10;
        goto end;
    }
    if (i2 == 0) {
        if ((i1 == 0) && (i2 == 0)) {
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c1 << c6 << c6 << c6 << c8;
            w1.Y++;
            cout << c5;
            w1.X += 4;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X -= 4;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c7 << c6 << c6 << c6 << c10;
            goto end;
        }
        if ((i1 == x-1) && (i2 == 0)) {
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c8 << c6 << c6 << c6 << c2;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X += 4;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X -= 4;
            w1.Y++;
```

```
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c10 << c6 << c6 << c6 << c11;
            goto end;
        }
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c8 << c6 << c6 << c6 << c8;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X += 4;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X -= 4;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c10 << c6 << c6 << c6 << c10;
        goto end;
    }
    if (i1 == x-1) {
        if ((i1 == x-1 ) && (i2 == 0)) {
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c8 << c6 << c6 << c6 << c2;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X += 4;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X -= 4;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c10 << c6 << c6 << c6 << c9;
            goto end;
        }
        if ((i1 == x-1 ) && (i2 == y-1 )) {
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c10 << c6 << c6 << c6 << c11;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X += 4;
            SetConsoleCursorPosition(hStdOut, w1);
```

```
            cout << c5;
            w1.X -= 4;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c9 << c6 << c6 << c6 << c3;
            goto end;
        }
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c10 << c6 << c6 << c6 << c11;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X += 4;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X -= 4;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c10 << c6 << c6 << c6 << c11;
        goto end;
    }
    if (i2 == y-1) {
        if ((i1 == x-1) && (i2 == y-1 )) {
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c10 << c6 << c6 << c6 << c11;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X += 4;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X -= 4;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c9 << c6 << c6 << c6 << c3;
            goto end;
        }
        if ((i1 == 0) && (i2 == y-1 )) {
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c7 << c6 << c6 << c6 << c10;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
```

```
            cout << c5;
            w1.X += 4;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c5;
            w1.X -= 4;
            w1.Y++;
            SetConsoleCursorPosition(hStdOut, w1);
            cout << c4 << c6 << c6 << c6 << c9;
            goto end;
        }
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c10 << c6 << c6 << c6 << c10;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X += 4;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c5;
        w1.X -= 4;
        w1.Y++;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << c9 << c6 << c6 << c6 << c9;
        goto end;
    }
end:
    if (i == 0) {
        switch (ch) {
        case 72: {
            if (i2 > 0)
                i2--;
            else
                i2 = y - 1;
            break;
        }
        case 80: {
            if (i2 < y - 1)
                i2++;
            else
                i2 = 0;
            break;
        }
        case 77: {
```

```
            if (i1 < x - 1)
               i1++;
            else
               i1 = 0;
            break;
         }
         case 75: {
            if (i1 > 0)
               i1--;
            else
               i1 = x - 1;
            break;
         }
         default:break;
         }
      }
      i++;
   }
   else switch (ch) {
      case 13: {
         if (q2==1)
            t1 = time(0);
         q2++;
         if (nflags == 0)
            Win(s, x, y);
         Open(i1, i2, x, y, s);
         break;
         }
      case 27: {
         q1 = 1;
         w1.X = sizex / 5 * 4-5;
         w1.Y = sizey / 5 + 8;
         SetConsoleTextAttribute(hStdOut, 32);
         SetConsoleCursorPosition(hStdOut, w1);
         cout<<"Are you sure to exit?";
         w1.X += 5;
         w1.Y += 2;
         SetConsoleCursorPosition(hStdOut, w1);
         SetConsoleTextAttribute(hStdOut, 46);
         cout << "Yes     ";
         SetConsoleTextAttribute(hStdOut, 32);
         cout << "No";
```

```
while (q1) {
    if (_kbhit()) {
        ch = _getch();
        switch (ch) {
        case 77:
        case 75: {
            if (j == 0) {
                w1.X = sizex / 5 * 4;
                w1.Y = sizey / 5 + 10;
                SetConsoleCursorPosition(hStdOut, w1);
                SetConsoleTextAttribute(hStdOut, 32);
                cout << "Yes     ";
                SetConsoleTextAttribute(hStdOut, 46);
                cout << "No";
                j++;
                break;
            }
            else {
                w1.X = sizex / 5 * 4;
                w1.Y = sizey / 5 + 10;
                SetConsoleCursorPosition(hStdOut, w1);
                SetConsoleTextAttribute(hStdOut, 46);
                cout << "Yes     ";
                SetConsoleTextAttribute(hStdOut, 32);
                cout << "No";
                j--;
                break;
            }
        }
        case 13:{
            if (j == 0) {
                q1 = 0;
                Menu();
                break;
            }
            else {
                q1 = 0;
                SetConsoleTextAttribute(hStdOut, 45);
                w1.X = sizex / 5 * 4 - 5;
                w1.Y = sizey / 5 + 8;
                SetConsoleCursorPosition(hStdOut, w1);
                cout << "                    ";
```

```
                w1.Y += 2;
                SetConsoleCursorPosition(hStdOut, w1);
                cout << "                  ";
                break;
              }
            }
            }
        }
    }
    break;
}
case 32: {
    if ((*(s + i1 * y + i2)).open == 0) {
        if (nflags == 0)
            break;
        w1.X = 9 + i1 * 4;
        w1.Y = 2 + i2 * 2;
        SetConsoleCursorPosition(hStdOut, w1);
        SetConsoleTextAttribute(hStdOut, 36);
        cout << " " << fl << " ";
        (*(s + i1 * y + i2)).open = 2;
        nflags--;
        SetConsoleTextAttribute(hStdOut, 32);
        w1.X = sizex / 5 * 4+8;
        w1.Y = sizey / 5+4;
        SetConsoleCursorPosition(hStdOut, w1);
        cout << nflags << " ";
        if (nflags == 0)
            Win(s, x, y);
        break;
    }
    if ((*(s + i1 * y + i2)).open == 2) {
        w1.X = 9 + i1 * 4;
        w1.Y = 2 + i2 * 2;
        SetConsoleCursorPosition(hStdOut, w1);
        SetConsoleTextAttribute(hStdOut, 36);
        cout << "   ";
        (*(s + i1 * y + i2)).open = 0;
        nflags++;
        w1.X = sizex / 5 * 4 + 8;
        w1.Y = sizey / 5 + 4;
        SetConsoleTextAttribute(hStdOut, 32);
```

```
                    SetConsoleCursorPosition(hStdOut, w1);
                    cout << nflags<<" ";
                    break;
                }
            }
        }
        }
        if (q2 > 1)
        myclock(t1);
        if (nflags == 0)
            Win(s, x, y);
}
}
void Open(int i1, int i2,int x, int y ,saper* s) {
    COORD w1;
    if ((*(s + i1 * y + i2)).open == 1)
        return void();
    if ((*(s + i1 * y + i2)).close == 0) {
        Open0(i1, i2,x,y,s);
        return void();
    }
    if ((*(s + i1 * y + i2)).close > 0 && (*(s + i1 * y + i2)).close < 10) {
        w1.X = 9 + i1 * 4;
        w1.Y = 2 + i2 * 2;
        SetConsoleCursorPosition(hStdOut, w1);
        SetConsoleTextAttribute(hStdOut, 97);
        cout << " " << (*(s + i1 * y + i2)).close << " ";
        (*(s + i1 * y + i2)).open = 1;
        return void();
    }
    if ((*(s + i1 * y + i2)).close = 10) {
        Lost(s,x,y);
        return void();
    }
}
void Open0(int i1, int i2, int x, int y,saper* s) {
    if ((*(s + i1 * y + i2)).open == 1)
        return void();
    COORD w1;
    w1.X = 9 + i1 * 4;
    w1.Y = 2 + i2 * 2;
    SetConsoleTextAttribute(hStdOut, 97);
```

```
SetConsoleCursorPosition(hStdOut, w1);
(*(s + i1 * y + i2)).open = 1;
cout << "   ";
if (0 < i1 && i1 < x - 1 && 0 < i2 && i2 < y - 1) {
   Open(i1 + 1, i2, x, y, s);
   Open(i1 + 1, i2 + 1, x, y, s);
   Open(i1 + 1, i2-1, x, y, s);
   Open(i1, i2 + 1, x, y, s);
   Open(i1, i2 - 1, x, y, s);
   Open(i1 - 1, i2 + 1, x, y, s);
   Open(i1 - 1, i2 - 1, x, y, s);
   Open(i1 - 1, i2, x, y, s);
   return void();
}
if (i1 == 0) {
   if ((i1 == 0) && (i2 == y - 1)) {
      Open(i1 + 1, i2, x, y, s);
      Open(i1 + 1, i2 - 1, x, y, s);
      Open(i1, i2 - 1, x, y, s);
      return void();
   }
   if ((i1 == 0) && (i2 == 0)) {
      Open(i1 + 1, i2, x, y, s);
      Open(i1 + 1, i2 + 1, x, y, s);
      Open(i1, i2 + 1, x, y, s);
      return void();
   }
   Open(i1 + 1, i2, x, y, s);
   Open(i1 + 1, i2 + 1, x, y, s);
   Open(i1 + 1, i2 - 1, x, y, s);
   Open(i1, i2 + 1, x, y, s);
   Open(i1, i2 - 1, x, y, s);
   return void();
}
if (i2 == 0) {
   if ((i1 == 0) && (i2 == 0)) {
      Open(i1 + 1, i2, x, y, s);
      Open(i1 + 1, i2 + 1, x, y, s);
      Open(i1, i2 + 1, x, y, s);
      return void();
   }
   if ((i1 == x - 1) && (i2 == 0)) {
```

```
            Open(i1 - 1, i2 + 1, x, y, s);
            Open(i1 - 1, i2, x, y, s);
            Open(i1, i2 + 1 , x, y, s);
            return void();
        }
        Open(i1 + 1, i2, x, y, s);
        Open(i1 + 1, i2 + 1, x, y, s);
        Open(i1, i2 + 1, x, y, s);
        Open(i1 - 1, i2 + 1, x, y, s);
        Open(i1 - 1, i2, x, y, s);
        return void();
    }
    if (i1 == x - 1) {
        if ((i1 == x - 1) && (i2 == 0)) {
            Open(i1 - 1, i2 + 1, x, y, s);
            Open(i1 - 1, i2, x, y, s);
            Open(i1, i2 - 1, x, y, s);
            return void();
        }
        if ((i1 == x - 1) && (i2 == y - 1)) {
            Open(i1, i2 - 1, x, y, s);
            Open(i1 - 1, i2 - 1, x, y, s);
            Open(i1 - 1, i2, x, y, s);
            return void();
        }
        Open(i1, i2 + 1, x, y, s);
        Open(i1, i2 - 1, x, y, s);
        Open(i1 - 1, i2 + 1, x, y, s);
        Open(i1 - 1, i2 - 1, x, y, s);
        Open(i1 - 1, i2, x, y, s);
        return void();
    }
    if (i2 == y - 1) {
        if ((i1 == x - 1) && (i2 == y - 1)) {
            Open(i1, i2 - 1, x, y, s);
            Open(i1 - 1, i2 - 1, x, y, s);
            Open(i1 - 1, i2, x, y, s);
            return void();
        }
        if ((i1 == 0) && (i2 == y - 1)) {
            Open(i1 + 1, i2, x, y, s);
            Open(i1 + 1, i2 - 1, x, y, s);
```

```
            Open(i1, i2 - 1, x, y, s);
            return void();
        }
        Open(i1 + 1, i2, x, y, s);
        Open(i1 + 1, i2 - 1, x, y, s);
        Open(i1, i2 - 1, x, y, s);
        Open(i1 - 1, i2 - 1, x, y, s);
        Open(i1 - 1, i2, x, y, s);
        return void();
    }
}
void myclock(time_t t1) {
    int s=0, m=0,q=1, sizex, sizey, sumseconds;
    COORD w1;
    time_t t2 = time(0);
    CONSOLE_SCREEN_BUFFER_INFO csbuf;
    GetConsoleScreenBufferInfo(hStdOut, &csbuf);
    sizex = csbuf.srWindow.Right;
    sizey = csbuf.srWindow.Bottom;
    sumseconds=difftime(t2, t1);
    m = sumseconds / 60;
    s = sumseconds % 60;
        w1.X = sizex / 5 * 4+6;
        w1.Y = sizey / 5+2;
        SetConsoleCursorPosition(hStdOut, w1);
        SetConsoleTextAttribute(hStdOut, 32);
        if (m == 0)
            cout << "00:";
        if(m>0&&m<10)
            cout << "0"<<m<<":";
        if(m>=10)
            cout << m << ":";
        if (s == 0)
            cout << "00";
        if (s > 0 && s < 10)
            cout << "0" << s;
        if (s >= 10)
            cout << s;
}
void Lost(saper*s, int x, int y) {
    int i1=0, i2=0, sizex,sizey;
    COORD w1;
```

```
    CONSOLE_SCREEN_BUFFER_INFO csbuf;
    SetConsoleTextAttribute(hStdOut, 97);
    while (i1 < x) {
        i2 = 0;
        while (i2 < y) {
            w1.X = 9 + i1 * 4;
            w1.Y = 2 + i2 * 2;
            SetConsoleCursorPosition(hStdOut, w1);
            if((*(s + i1 * y + i2)).close==0)
            cout << "   ";
            if ((*(s + i1 * y + i2)).close == 10) {
                SetConsoleTextAttribute(hStdOut, 36);
                cout << " * ";
                SetConsoleTextAttribute(hStdOut, 97);
            }
            if ((*(s + i1 * y + i2)).close > 0&& (*(s + i1 * y + i2)).close < 10)
            cout << " " << (*(s + i1 * y + i2)).close << " ";
            i2++;
        }
        i1++;
    }
    GetConsoleScreenBufferInfo(hStdOut, &csbuf);
    sizex = csbuf.srWindow.Right;
    sizey = csbuf.srWindow.Bottom;
    w1.X = sizex / 5 * 4 +1;
    w1.Y = sizey / 5 + 8;
    SetConsoleTextAttribute(hStdOut, 36);
    SetConsoleCursorPosition(hStdOut, w1);
    cout<<"YOU LOST";
    Sleep(5000);
    Menu();
}
void Win(saper* s, int x, int y) {
    CONSOLE_SCREEN_BUFFER_INFO csbuf;
    int i1 = 0, i2 = 0, sizex,sizey;
    COORD w1;
    while (i1 < x) {
        i2 = 0;
        while (i2 < y) {
            if (((*(s + i1 * y + i2)).close >= 0 && (*(s + i1 * y + i2)).close < 10 && (*(s + i1 * y +
i2)).open == 1) || ((*(s + i1 * y + i2)).close == 10 && (*(s + i1 * y + i2)).open == 2))
                i2++;
```

```
            else return void();
        }
        i1++;
        GetConsoleScreenBufferInfo(hStdOut, &csbuf);
        sizex = csbuf.srWindow.Right;
        sizey = csbuf.srWindow.Bottom;
        w1.X = sizex / 5 * 4 + 2;
        w1.Y = sizey / 5 + 8;
        SetConsoleTextAttribute(hStdOut, 46);
        SetConsoleCursorPosition(hStdOut, w1);
        cout << "YOU WON";
        Sleep(5000);
        Menu();
    }
}
```