Черкаський національний університет імені Богдана Хмельницького

Кафедра програмного забезпечення автоматизованих систем

КУРСОВА РОБОТА

з дисципліни «Програмування та алгоритмічні мови»

НА ТЕМУ «Розробка гри «Морський бій»»

Студентки 1 курсу, групи КС-19

напряму підготовки «Програмна інженерія»

спеціальності «Інженерія програмного забезпечення»

Ковтун Єлизавети Русланівни

Керівник старший викладач Гребенович Ю.Є.

Національна шкала _____

Кількість балів: _____ Оцінка: ECTS _____

Члени комісії:

_____ Гребенович Ю.Є.
(підпис) (прізвище та ініціали)

_____ Онищенко Б.О.
(підпис) (прізвище та ініціали)

_____ Порубльов І.М.
(підпис) (прізвище та ініціали)

м. Черкаси – 2020 рік

Зміст

Вступ

Мета:

Набуття та покращення навичок в розробці програмного забезпечення на об’єктно-орієнтовній мові програмування C#. Обробка первісного завдання, розробка та реалізація алгоритмів, які зазначені в завданні. Дослідження функціоналу. Складання блок-схем до основних алгоритмів.

Завдання

Написати гру «Морський бій» для двох осіб. Гра має містити автоматичну та ручну розстановку кораблів, метод перевірки, чи убитий корабель, метод потоплення корабля та перевірку, чи закінчена гра після кожного влучного попадання.

Постановка задач курсової роботи

1. Створити алгоритми для реалізації гри.
2. Побудувати блок-схеми до основних алгоритмів.
3. Реалізувати у вигляді програмного продукту.
4. Написати пояснювальну записку.
5. Скласти коротку та змістовну доповідь на захист.

Актуальність теми

«Морський бій»- це гра на логіку, яка включає в себе стратегію та азарт гравця та є гарним методом, як з користю провести дозвілля. Комп’ютерна версія даної гри є гарною альтернативою канонічному аркушу паперу. Комп’ютерний «Морський бій» користується великим попитом як серед дітей та підлітків, так і серед старшого покоління.

Правила гри

1. При розстановці кораблі не можуть торкатися іншого корабля.
2. Якщо вистріл влучний, противник пропускає хід.

3. Якщо всі палуби корабля потоплені, корабель вважається потопленим, та всі клітинки навколо корабля, що його торкаються, вважаються відкритими.
4. Перемагає той гравець, що першим потопить кораблі противника.

Розділ 1. Огляд програмного забезпечення для реалізації проекту та існуючих ігор на дану тематику

1.1 Вибір алгоритмів

Під час вибору алгоритмів необхідно перебрати багато варіантів, проаналізувати та вибрати найкраще рішення. При виконанні даного проекту я неодноразово робила вищезазначених рішення.

Всю свою роботу я почала з автоматичної розстановки кораблів. Свій вибір я зупинила на варіанті, який було описано в циклі відео на Ютюб.[2]

Опис алгоритму:

Все відбувається в двох вкладених циклах. Перший цикл- розмір корабля, другий- кількість кораблів. Рандомно обираються координати початку та напрям. Далі ведеться перевірка, чи не накладається цей корабель на інші. В разі позитивного результату корабель ставиться на обране місце, інакше-дії повторюються за допомогою оператора goto.

Під час створення коду неодноразово з'являлась проблема, що клітинки навколо корабля виходили за границі поля та виникали проблеми реалізації. Тому я змінила розмір масиву 12 на 12 та на початку заповнення 0.

Потім було написано вистріл гравця, перевірка чи закрита клітинка. Наступним етапом створення перевірки, чи повністю затоплено корабель. Вибір був між перевіркою на чотири клітинки та у всіх напрямах та додатковим масивом для кожного гравця, під кожним індексом зазначена кількість клітинок корабля. Легшим в реалізації став другий варіант. При кожному влучному попаданні кількість значення зменшується, поки значення не буде рівне нулю. В результаті цього потрібно було змінити позначення корабля в масиві. Якщо до цього моменту кораблі позначалися своїм розміром, то тепер індексами від 1 до 10. [3]

Останній алгоритм це відкриття клітинок навколо корабля. Він був написаний на основі розстановки кораблів, але для цього потрібно було знати

розмір, напрям та місце знаходження корабля. Для напряму перевіряються 4 сторони навколо останньої влучно відкритої клітинки. Коли знаходиться напрям необхідно знайти початок корабля. Проводиться перевірка вверх або вліво, в залежності від напряму корабля, до моменту, поки не знайдеться першої клітинки, що не належать кораблю.

Залишається відшукати лише розмір. У мене було два варіанти: прохід від початку, поки не буде знайдено першу відмінну клітинку. Але це набагато складніше, ніж створити ще один масив, під індексом якого будуть зберігатися розміри. Тому перед зміною значення відкритої клітинки запам’ятовується індекс, за яким ми знаходимо розмір корабля.

Після кожного затопленого корабля проводиться перевірка на виграш. Виповнюється прохід по масиву та кожну затоплену палубу корабля рахує. В разі, якщо кількість таких палуб рівна 20, гра закінчена.

Щодо позначень в масивах:

1. 0-пуста клітинка.
2. «1-10»- не відкриті клітинки кораблів
3. «11»-клітинки навколо корабля(використовуються лише при розстановці кораблів).
4. «12»-відкрита пуста клітинка.
5. «13»-відкрита клітинка корабля.

Щодо кількості масивів:

Є два основні масиви- так би мовити поля, в яких ведеться вся гра, два масиви в яких зазначена кількість закритих клітинок, два масиви типу char для графічного зображення масиву. Два масиви завдяки якому створюється коректне зображення та масив з розмірами.

Графічне зображення- мій вибір зупинився в консольному вигляді. Плюси: звичний для мене інтерфейс та лістинг. Мінуси: відсутність графічного зображення.

Графічне зображення поля:

1. «0»- пуста відкритта клітинка
2. «#»-корабель
3. «x»-поранений(потоплений) корабель
4. «.»- пуста не відкрита клітинка

1.2 Огляд існуючих ігор на тему «Морський бій»

Якщо враховувати, те що «Морський бій» досить популярна гра, то знайти аналоги на всі сучасні платформи(Windows, Mac, Linux-комп'ютерні;IOS, Android-мобільні) досить не складно.Ось декілька прикладів:

1. **Battle of Warships**- гра сумісна з Windows 7, 8, 10. Має звуковий супровід, підтримує гру по мережі, наявна анімація. [4]
2. **Морський бій(в оригіналі Морской бой)** для платформи Android-найпопулярніший додаток на дану тему на мобільних телефонах. Присутня приємна графіка, музичний супровід, розумний бот з різними рівнями складності.[5]

1.3 Висновок до першого розділу

Отже, в цьому розділі було розібрано шлях вибору алгоритмів, прописані всі аргументи даного вибору. Проведено моніторинг ігор на дану тему на різних операційних системах.

Розділ 2. Проектування програмного продукту

2.1 Загальний алгоритм роботи програми

1. Запуск гри.
2. Вибір як розставити кораблі першому гравцю.
3. Вибір як розставити кораблі другому гравцю.
4. Хід гравців.
5. Перевірка, чи закрита ця клітинка.
6. Якщо клітинка відкрита, гравцю пропонується поварити вистріл.
7. Якщо клітинка пуста, виконується хід противника
8. Якщо гравець попав в палубу корабля, то йде перевірка, чи він потоплений.
9. Якщо корабель потоплений, відкриваються клітинки навколо корабля.
10. Перевірка на виграш.
11. Якщо кількість відкритих палуб корабля рівна 20, гра завершена виграшом останнього гравця.

2.2 Опис основного алгоритму гри

Після запуску гри, гравцю номер 1 пропонується обрати метод розстановки кораблів: автоматичну чи ручну. Якщо розстановка ручна, виводиться повідомлення «Розташуйте корабель розміром «розмір»», «Введіть координати початку корабля» та обрати напрім, якщо корабель торкається, або находить, на палуби інших кораблів чи границі, виводиться повідомлення: «Корабель не можна поставити, повторіть заново», якщо можна : «Корабель можна поставити» після це все виконується під гравцем номер 2.

Після розстановки починається сама гра. Є два поля, всі пронумеровані від 1 до 10Вистріли ведуться у вигляді «А 1» по черзі до моменту, доки один з гравців не влучить. Коли гравець підбив корабель, перевіряється чи потоплений корабель, якщо так, поле навколо корабля помічається як відкрите. Далі вистріл

гравця, що попередньо підбив, або потопив, корабель. У випадку, якщо гравець хоче відкрити вже відкриту клітинку, виводиться повідомлення, що дана за цими. Координатами вже все було відкрито виводиться повідомлення: «Ця клітинка вже відкрита, повторіть спробу». Якщо гравець влучно потрапив, але не затопив корабель виводиться повідомлення: «Поранено!», а якщо корабель повністю вбито: «Потоплено!». Коли гравець відкриває пусту клітинку виводиться повідомлення: «Промах!». Коли на полі одного з гравців, наприклад 1, відкриті усі клітинки кораблів гра вважається завершеною, виводиться повідомлення: «Перемога гравця 2!!! Дякую за гру!!!».

2.3 Опис допоміжних алгоритмів у грі

Функція на перевірку чи відкрита клітинка Check. При пострілі гравця, йде перевірка, чи закрита ця клітинка. В разі, якщо вона закрита, повертається значення true булевої функції і далі відбувається вищезазначений алгоритм, інакше гравцю необхідно повторити дії з іншими координатами.

Функція FindBeginningShipDirection-функція яка шукає початок потопленого корабля, щоб завершити процес потоплення корабля. В програмі зазначено дві схожі функції з різними індексами. 0- для горизонтальних кораблів, 1-вертикальних.

2.4 Висновок до другого розділу

В цьому розділі було сформовано список головних задач які виконує даний продукт. Було коротко описаний алгоритм. Створена основна блок-схема(додаток «А»). Можна виділити основні цілі реалізації:

1. Автоматична та ручна розстановка кораблів.
2. Функція перевірки, чи потоплений корабель.
3. Функція затоплення корабля.
4. Функція перевірки на завершення гри.

Розділ 3. Реалізація гри

3.1 Реалізація основних алгоритмів

Лістинг 1- функція вибору виду розстановки кораблів

```
public static void Field(int[,] Player, char[,] player)
{
    Console.WriteLine("Хочете розтавити кораблі автоматично?");
    Console.WriteLine("Відповідь формату так або ні");
    string answer = Console.ReadLine(); //Рядок для відповіді
    if (answer == "так")
    {
        ShipsRandom(Player);            //Автоматична розстановка
    }
    else
    {
        KeyboardShips(Player, player);  //Ручна розстановка
    }
    Console.WriteLine("Поле:");
    Output(Player, player);             //Вивід поля гравця
    for (int i = 0; i < 30; i++)        //Розділення полей
    {                                   //задля збереження таємниці
        Console.WriteLine();            //розстановки кораблів противника
    }
}
```

Рис. 3.1- Функція вибору розстановки кораблів

При відповіді «так» кораблі автоматично розставляються.

Лістинг 2- автоматична розстановка кораблів

```
static void ShipsRandom(int[,] Field)
{
    Zero(Field);
    int count = 1;
    int row = 0;
    int column = 0;
    int direction, size;
    Random rand = new Random();
    for (int i = 4; i >= 1; i--)         //розмір коробля
    {
        for (int j = 1; j <= 5 - i; j++) //кількість кораблів
        {
        mark:
            row = rand.Next(1, 11);      //рядок початку корабля
            column = rand.Next(1, 11);   //стовпець початку корабля
            direction = rand.Next(0, 2); //напрямок(0-горизональний, 1-вертекальний)
            size = i;                    //розмір корабля
            bool flag = CheckTheAbilityToShip(row, column, direction, size, Field);
            if (flag == true)            //Функція перевірки, чи можна поставити корабель
            {                            //Якщо можна
                PutShip(row, column, direction, size, Field, count);
                                        //Поставити корабель
```

Рис. 3.2- функція автоматичної розстановки кораблів

```
                                        //Поставити корабель
                count++;                //Змінна,якою помічаються кораблі
            }
            else                        //В іншому випадку поворити
            {
                goto mark;
            }
        }
    }
}
```

Рис. 3.2.1- продовження рис.2

Інакше кораблі розставляються в ручну. Необхідно написати координати у. вигляді «А 1». Потім вибрати напрямок 1-верикально, 0-горизонтально. В разі якщо можна поставити корабель виводиться повідомлення: «Корабель можна поставити». В. інакшому випадку виводиться повідомлення: «Корабель не можна поставити, повторіть заново» і за допомогою оператора goto повертаємось до моменту вводу координат.

Лісинг-3 Ручна розстановка кораблів

```
static void KeyboardShips(int[,] Field, char[,] field)
{
    Zero(Field);//функція, що завняє масив нулями
    int count = 1;// змінна, яккою помічаться кораблі в числовому масиві
    for (int i = 4; i >= 1; i--)//розмір корабля
    {
        for (int j = 1; j <= 5 - i; j++)//кількість кораблів
        {
            Console.WriteLine("Розташуйте корабель розміром {0}", i);
        mark:
            int size = i; //Розмір корабля
            Console.WriteLine("Введіть координати початку корабля");
            Console.WriteLine("В форматі A 10");
            string coordinates = Console.ReadLine();//ввід координат
            int row = RetRow(coordinates);//рядок початку корабля
            int column = Convert.ToInt32(coordinates.Split(' ')[1]);//стовпець початку корабля
            Console.WriteLine("Виберіть напрям 1- вертикально, 0-горизонтально");
            int direction = int.Parse(Console.ReadLine());//напрям корабля
            bool flag = CheckTheAbilityToShip(row, column, direction, size, Field);//функція перевірки
            if (flag == true)           //якщо можно поставити корабль
            {
                Console.WriteLine("Корабель можна поставити");
                PutShip(row, column, direction, size, Field, count);//функція
                Output(Field, field);//вивід поля
                count++;

            }
            else
            {
                Console.WriteLine("Корабель не можна поставити, повторіть заново");
                goto mark;
            }
        }
    }
}
```

Рис. 3.3-ручна розстановка кораблів

Коли перший гравець розставив бажаним способом кораблі, наступає черга другого гравців розставляти.

Коли всі кораблі розставлені, перший гравець починає гру з вистрілу. Проводиться перевірка, чи закрита ця клітинка. Якщо клітинка виявиться відкрита, виведеться повідомлення, про необхідність повторити вистріл. Якщо клітинка закрита. Проводиться перевірка і якщо клітинка пуста, вона помічається, як відкрита та проводиться за цим же принципом вистріл другого гравця. Якщо ж вистріл влучний проводиться перевірка, чи потоплений корабель. Якщо потоплено, виконується функція потоплення та виводиться повідомлення: «Потоплено», інакше «Поранено» та проводиться ще один вистріл гравця. Якщо вистріл не влучний виводиться повідомлення: «Промах» та надається прало вистрілу іншому гравцю за цим же принципом.

Лістинг 4-Функція вистрілу гравців

```
mark:
mark2:
    Console.WriteLine("Хід 1 гравця");
    Console.WriteLine("Введіть координати в форматі А 10");
    string coordinates = Console.ReadLine();
    int row = Ships.RetRow(coordinates);
    int column = Convert.ToInt32(coordinates.Split(' ')[1]);
    bool cheak = Program.Check(Player2, row, column);//Перевірка, чи закриця кліипнка
    if (cheak == false)
    {
        Console.WriteLine("Ця клітинка вже відкрита, повторіть спробу");
        goto mark;
    }
    else
    {
        if (Player2[row, column] == 0 || Player2[row, column] == 11)//якщо клітика пуста
        {
            Player2[row, column] = 12;      //позначається як відкритта пуста
            Pl2[row, column] = 12;
            Program.OutPut(Pl1, Pl2, player1, player2); //вивід поля
            Console.WriteLine("Промах!");
            Opponent2.Brain(Player1, player1, Player2, player2, mas1, mas2, mas,Pl1, Pl2);
        }//хід другого гравця
        else
        {
            int index = Player2[row, column];
            mas2[index]--;
            Player2[row, column] = 13;
            Pl2[row, column] = 13;
            bool check_killed = Program.CheckShipKilled(Player2, mas2, index);
            if (check_killed == true)//перевірка  чи потоплений корабель
            {                        //якщо потопленно
                Program.SinkingShip(Player2, row, column, mas, index, Pl2);//функція, що позначить клітинки навколо
                Program.OutPut(Pl1, Pl2, player1, player2);
                Console.WriteLine("Потоплено!");
            }
            else
            {
                Program.OutPut(Pl1, Pl2, player1, player2);
                Console.WriteLine("Поранено!");
            }
```

Рис. 3.4-Функція вистрілу гравців

Після кожного потопленого корабля проводиться перевірка на перемогу. В разі, якщо кількість потоплених клітинок корабля рівна 20, гра завершена з виграшом останнього гравця.

Лістинг 5- Функція потоплення корабля(відкриття клітинок навколо потопленого корабля)

```
public static void SinkingShip(int[,] Field, int row, int column, int[] mas, int index, int [,] Pl)
{
    int direction;//напрям(1-вертикальний, 0-горизонатльний)
    if (Field[row - 1, column] == 13 || Field[row + 1, column] == 13)
    {//якщо кліинка зверху, або  знизу-корабель верикальний
        direction = 1;
    }//інакше, горизонтальний
    else direction = 0;
    if (direction == 1)
    {//функція пошуку початку корабля(при різних напрямах)
        row = FindBeginningShipDirection1(Field, row, column);//вертикальний
    }
    else column = FindBeginningShipDirection0(Field, row, column);//горизонтальний
    int size = mas[index];//розмір корабля
    MarkingCellsArroundShip(Field, direction, row, column, size, Pl);
}          //функція, що відкриває клітинки навколо
```

Рис. 3.5-потоплення корабля

Лістинг 6- перевірка на завершення гри

```
public static bool CheckWinner (int[,] Player)
{
    bool check = false;
    int count = 0;
    for(int i=1;i<11;i++)
    {
        for (int j = 1; j < 11; j++)
        {
            if(Player[i,j]==13)
            {//якщо клітинка є відкритим кораблем(або його частиною)
                count++;//рахунок збільшується
                if(count==20)
                {//якщо ккількість таких  кліинок рівна 20
                    check = true;//повертається значення true
                    break;//фуункція зупиняється
                }//гра закінченна
            }
        }
    }
    return check;
}
```

Рис. 3.6-перевірка, чи завершена гра

3.2 Реалізація інтерфейсу гри

Гра написана у консольному вигляді. На початку гри виводяться повідомлення та запитання на яке необхідно відповісти : «так» або «ні»

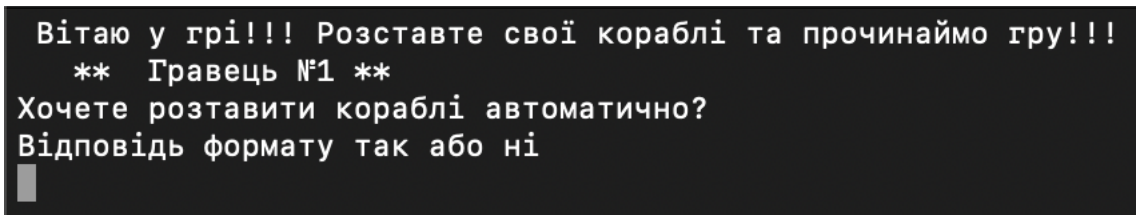

Рис. 3.7-початок гри

Якщо гравець відповідає так, відразу починається розстановка кораблів гравця номер 2. В інакшому випадку гравець 1 розставляє свої кораблі в ручну.

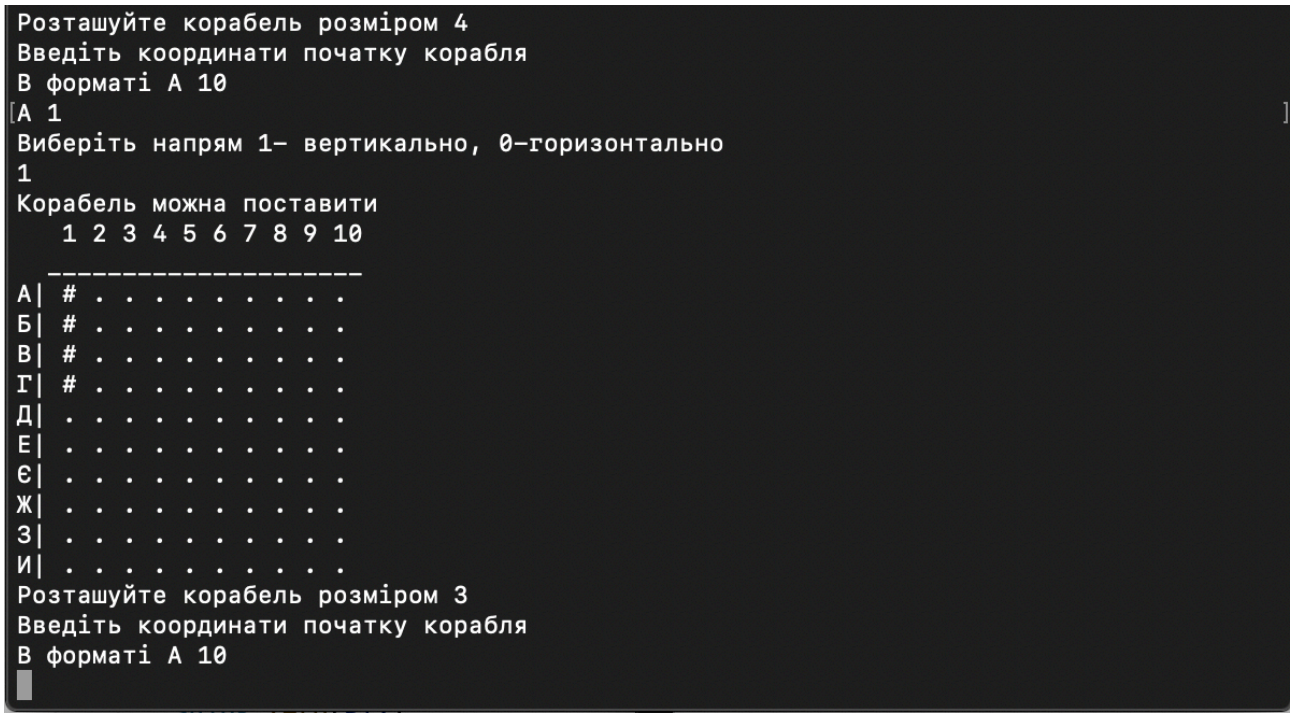

Рис.3.8-Вигляд, ручної розстановки кораблів

В разі якщо корабель поставити не можна, виводиться повідомлення.

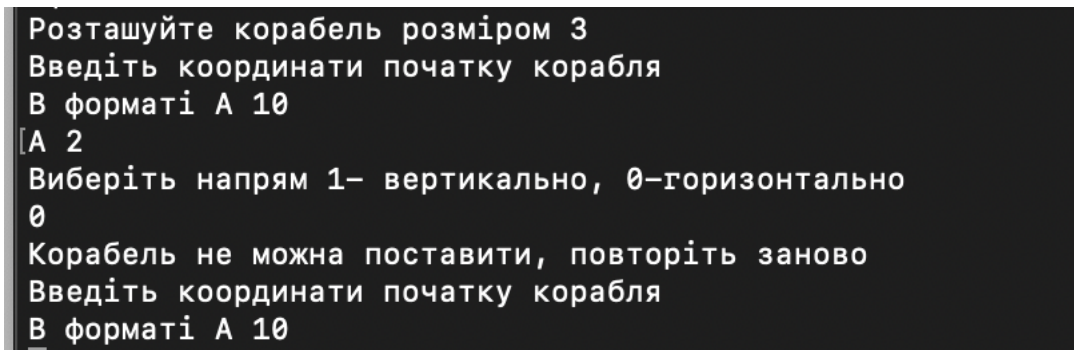

Рис. 3.9-Якщо корабель поставити не можливо

Коли кораблі розставили обидва гравці, починається гра.

Якщо гравець промахнувся, виводиться повідомлення а надається право вистрілу іншому гравцю.

```
Хід 1 гравця
Введіть координати в форматі А 10
А 1
   ** Поле гравця №1 **
   1 2 3 4 5 6 7 8 9 10
  --------------------
А| . . . . . . . . . .
Б| . . . . . . . . . .
В| . . . . . . . . . .
Г| . . . . . . . . . .
Д| . . . . . . . . . .
Е| . . . . . . . . . .
Є| . . . . . . . . . .
Ж| . . . . . . . . . .
З| . . . . . . . . . .
И| . . . . . . . . . .
   ** Поле гравця №2 **
   1 2 3 4 5 6 7 8 9 10
  --------------------
А| 0 . . . . . . . . .
Б| . . . . . . . . . .
В| . . . . . . . . . .
Г| . . . . . . . . . .
Д| . . . . . . . . . .
Е| . . . . . . . . . .
Є| . . . . . . . . . .
Ж| . . . . . . . . . .
З| . . . . . . . . . .
И| . . . . . . . . . .
Промах!
Хід 2 гравця
Введіть координати в форматі А 10
```

Рис. 3.10-Коли гравець не влучив

Коли гравець потрапляє влучно, йому надається ще один вистріл.

Якщо корабель потоплено виводиться повідомлення та навколо корабля відкриваються клітинки.(рис.3.11)

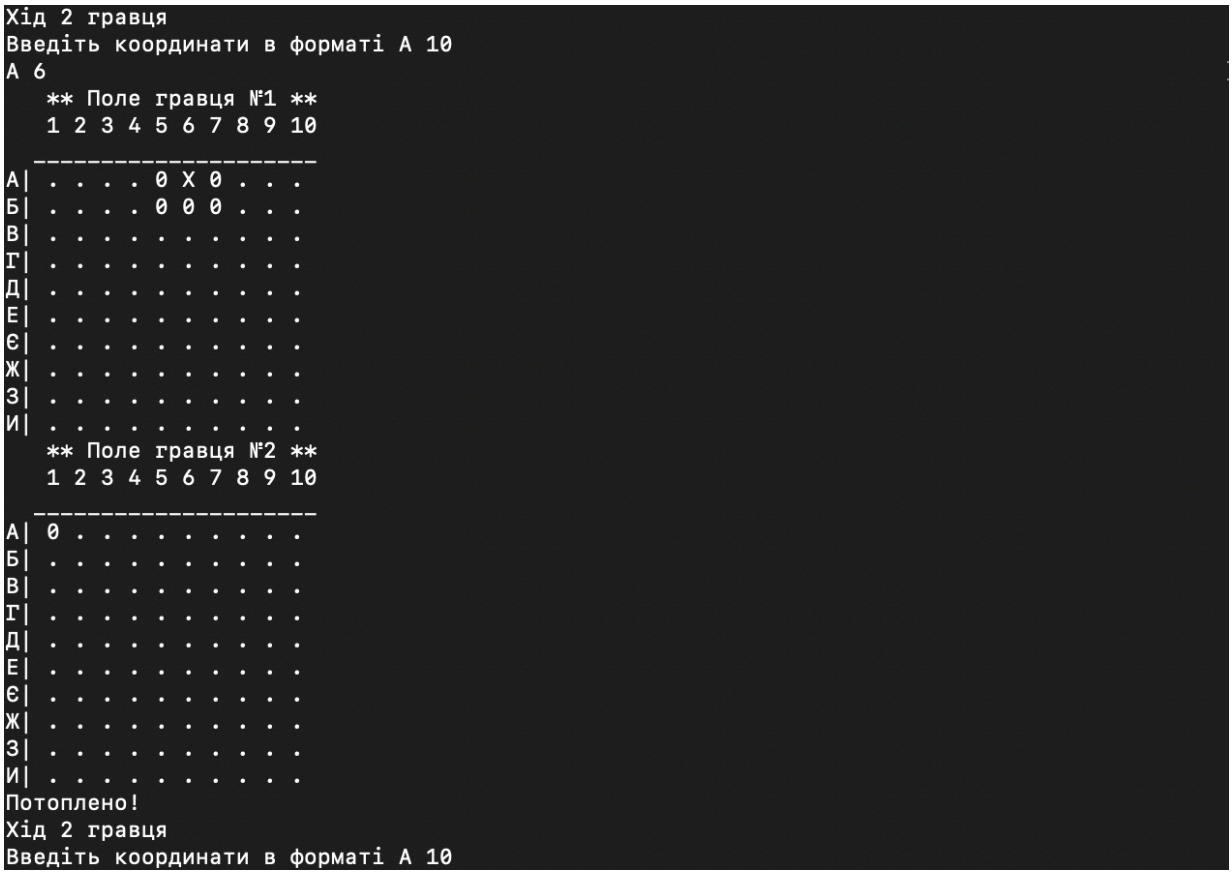

Рис. 3.11-Потоплений корабель

Якщо корабель поранено, але не потоплено виводиться повідомлення «Поранено».

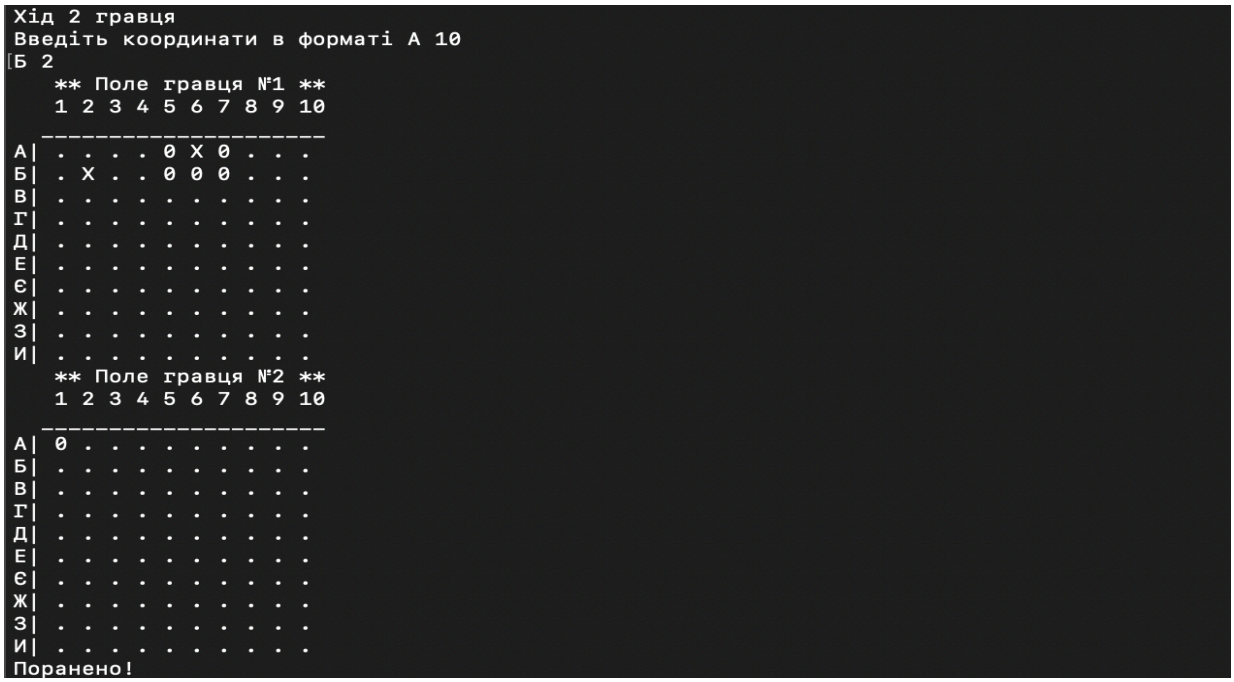

Рис. 3.12-Поранений корабель

В разі, якщо гравець намагається відкрити вже й так відкриту клітинку, виводиться повідомлення з проханням. Повторити.

Рис. 3.13-Відкрита клітинка

Коли один із гравців топить всі кораблі противника, гра завершена.(рис. 3.14)

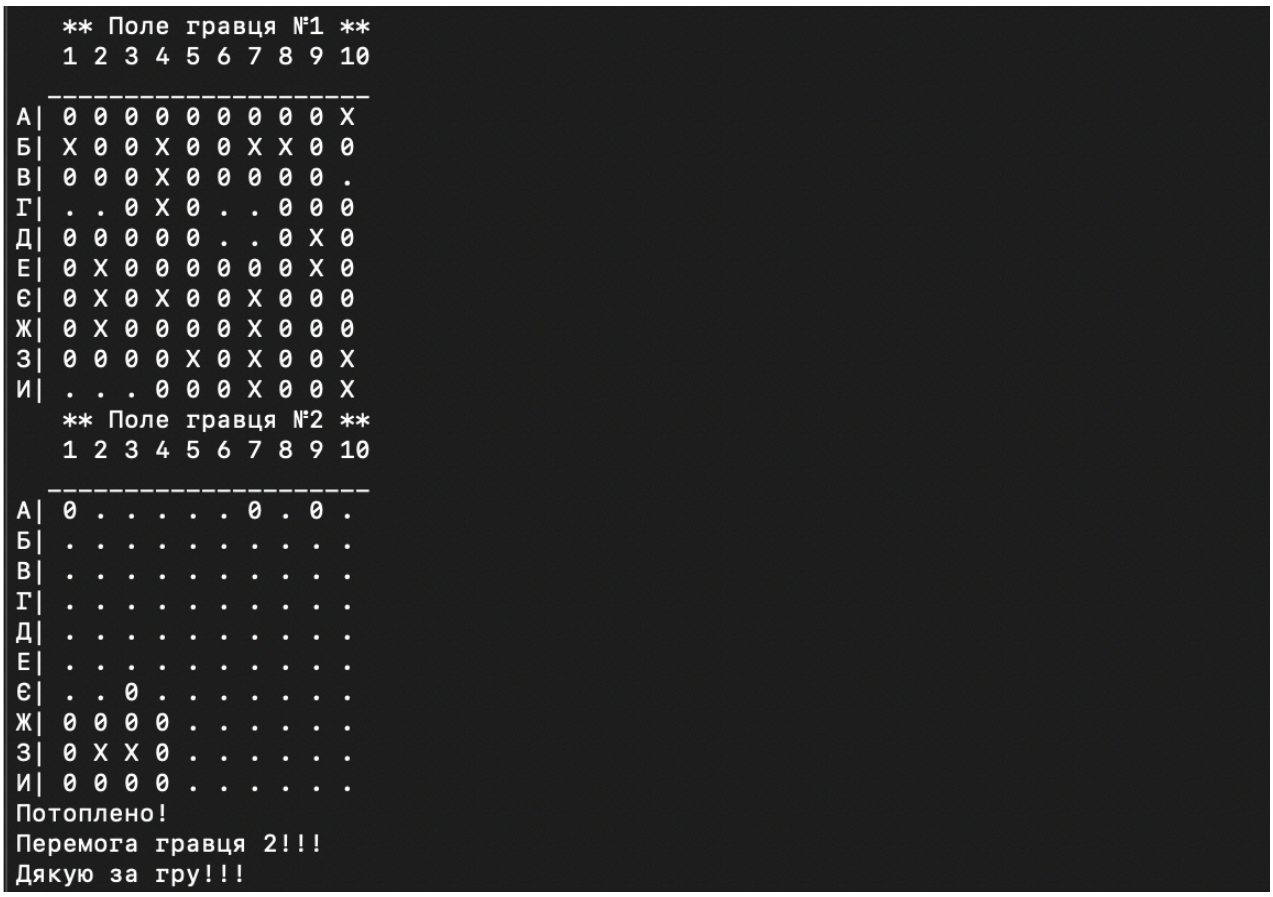

Рис. 3.14-Завершена гра

3.3 Висновок до третього розділу

Отже, в даному розділі було розібрано код до основних алгоритмів. Зроблена перевірка гри. В результаті гра виконує усі поставлені завдання коректно та швидко. Інтерфейсне зображення полів є інтуїтивно зрозумілим гравцям.

Висновки

Отже, проведено досить кропітку роботу на створені цього продукту. Було проаналізовано велика кількість інформації. Вибрані та створені найбільш підходящі алгоритми. Прописані всі аргументи даного вибору. Проведено моніторинг ігор на дану тему на різних операційних системах.

Було сформовано список головних задач які виконує даний продукт. Було коротко описаний алгоритм. Створена основна блок-схема(додаток «А»). Можна виділити основні цілі реалізації:

1. Автоматична та ручна розстановка кораблів.
2. Функція перевірки, чи потоплений корабель.
3. Функція затоплення корабля.
4. Функція перевірки на завершення гри

Було розібрано код до основних алгоритмів. Зроблена перевірка гри. В результаті гра виконує усі поставлені завдання коректно та швидко. Інтерфейсне зображення полів є інтуїтивно зрозумілим гравцям.

Планується:

1. Створити чесного та розумного бота для гри з системою.
2. Створення сучасного інтерфейсу та анімації.

Список використаних джерел:

1. Методичні вказівки до виконання та оформлення курсової роботи з дисциплін. “Основи програмування”, “Програмування та алгоритмічні мови”, “Алгоритмізація та програмування” Черкаси: Вид. від ЧНУ імені Богдана Хмельницького , 2016. – 32 с.
2. Создаем игру морской бой. Часть 1. Дизайн (планирование) первой итерации [Електронний ресурс] Режим доступу: https://youtu.be/cwvpua9awuY Перевірено: 16.05.2020
3. Программирование на C++. Урок 67. Морской бой. Полностью ли потоплен корабль? [Електронний ресурс] Режим доступу: https://youtu.be/R3IIVquoFvo Перевірено 16.05.2020
4. Battle of Warships: Морской бой на компьютер [Електронний ресурс] Режим доступу: https://playmarket-pc.com/battle-of-warships-morskoy-boy/ Перевірено: 16.05.2020
5. Sea Battle - Apps on Google Play [Електронний ресурс] Режим доступу: https://play.google.com/store/apps/details?id=com.byril.seabattle Перевірено: 16.05.2020