Черкаський національний університет імені Богдана Хмельницького
Кафедра інформаційних технологій

КУРСОВА РОБОТА

З дисципліни "Програмування та алгоритмічні мови"

На тему: "Гра "Бики та корови""

Студента 1 курсу КН-19 групи
спеціальності 122 Комп'ютерні науки
Старікова М.В.
Керівник: ст. викладач Стабецька Т.А.

Національна шкала_____
Кількість балів: _____ Оцінка: ECTS_____

Члени комісії _____ _____
(підпис) (прізвище та ініціали)
_____ _____
(підпис) (прізвище та ініціали)
_____ _____
(підпис) (прізвище та ініціали)

м. Черкаси – 2020 рік

Зміст

Вступ

Комп'ютерні ігри є одними з найпопулярніших розваг у наш час. У наші дні мільйони користувачів грають і проводять у віртуальному світі не малу кількість часу. Перші відео ігри з'явилися в другій половині XX століття і вже тоді отримали велику популярність, не тільки серед дітей, але і серед чималої кількості дорослих. Якість комп'ютерних ігор стрімко зростає. Звичайно це так само пов'язано з постійним поліпшенням інструментів і ігрових рушіїв (двигунів), за допомогою яких в основному і створюють відеоігри. На сьогоднішній день багато розробників ігор розробляють ігри на таких ігрових рушіях як Source, Unity, Unreal Engine, CryEngine, Construct та інших. Вони мають інтуїтивно зрозумілий інтерфейс, реалістичну симуляцію фізики, а деякі з них майже не вимагають знань програмування, що дає низький поріг входження для тих, хто тільки почав розробляти свої ігри та інші продукти. Завдяки сучасним ігрових рушіям гри вже можуть похвалитися фото реалістичною графікою, яку раніше могли мати хіба що високобюджетні фантастичні фільми і бойовики. Зростання графічної складової комп'ютерних ігор закликає розробників ігрового заліза постійно вдосконалюватися

Відеоігри можуть задовольнити будь-якого користувача, так як вони мають дуже багато жанрів на будь-який смак. Тисячі ентузіастів кожен день створюють захоплюючі проекти, які можуть затягнути на кілька годин, або ж навіть днів.

Звичайно, є дуже багато простих ігор, які не вимагають такої кількості потужності, але вони залишаються не менш цікавими. За допомогою тих же інструментів і ігрових рушіїв можна з легкістю створити і їх. З цього приводу тема моєї роботи - це створення комп'ютерної гри Бики і Корови.

Розділ 1. Постановка задачі.

Мета роботи – створення комп'ютерної гри Бики і Корови, що повинна симулювати реальну гру.

Задачі роботи:

1. Провести аналіз ігрових рушіїв для винонання роботи
2. Зробити огляд правил гри у Бики і Корови
3. Скласти структурну схему гри
4. Скласти структурну схему до ігрового процесу
5. Реалізувати у вигляді програмного продукту гру
6. Проаналізувати розроблений додаток

1.1 Вибір середовища для створення гри

Так як тема і завдання роботи є створення відеоігри, її краще створювати за допомогою ігрового рушія. Серед популярних ігрових рушіїв зараз знаходяться CryEngine, Unreal Engine і Unity. Всі вони мають досить таки дружній до користувача інтерфейс, безліч інструментів і навчальних матеріалів. Всі вони вимагають знання певної мови програмування. CryEngine вимагає знання мови C # або C ++, Unreal Engine вимагає знання C ++, Unity вимагає знання C # чи JavaScript. CryEngine і Unreal Engine використовують для розробки в основному масштабних ігор і проектів. Unity має куди більше навчальних матеріалів. має змогу без проблем перенести весь твій проект на всі популярні платформи та операційні системи, має безліч інструментів для разробки саме в 2D і в основному саме його обирають для створення простих ігор. Вибрати його буде більш розумно.

Отже, для створення гри було вирішено вибрати ігровий движок Unity. Він кросплатформний та куди більше підходить під створення 2D гри, ніж інші популярні ігрові рушії.

1.2 Огляд правил гри “Бики і Корови”

Комп'ютер загадує таємне 4-значне число з неповторними цифрами. Гравець же намагається його відгадати за певну кількість спроб (наприклад за 10 спроб) і робить першу спробу. Спроба - це 4-значне число з неповторними цифрами, яке повідомляється комп'ютеру. Комп'ютер повідомляє у відповідь, скільки цифр вгадано без збігу з їх позиціями в таємному числі (тобто кількість корів) і скільки вгадано аж до позиції в таємному числі (тобто кількість биків). Наприклад:

Задумано таємне число «3219».

Спроба: «2310».

Результат: дві «корови» (дві цифри: «2» і «3» - вгадані на невірних позиціях) і один «бик» (одна цифра «1» вгадана аж до позиції).

При грі проти комп'ютера гравець вводить комбінації одну за одною, поки не відгадає всю послідовність (Тобто не отримає 4 бика) або поки не закінчиться кількість спроб. Скриншот реалізації подібної програми показано на рис. 1.1.

Рис. 1.1 Приклад гри Бики і Корови

Отже, для створення гри, цих правил вистачить.

1.3 Висновки

Отже, головна задача роботи – створити комп’ютерну гру “Бики та Корови” за допомогою кросплатформеного ігрового рушія Unity в 2D форматі, використовуючи правила гри для проектування та реалізації загального алгоритму ігрового процесу.

Розділ 2. Проектування гри.

2.1 Проектування загальних елементів

Так як мета роботи – створити відеогру, де гравець намагається вгадати число, то необхідно реалізувати такі елементи:

- Головне меню
- Зручний Інтерфейс
- Генерація числа комп’ютером
- Поле введення для гравца
- Перевірка введеного числа на умови
- Порівняння чисел та вивід статистики для гравця під час гри
- Кнопки рестарту, та повернення до меню під час гри.

2.2 Проектування меню

Схема взаємодії з головним меню буде реалізовано як на рис. 2.1:

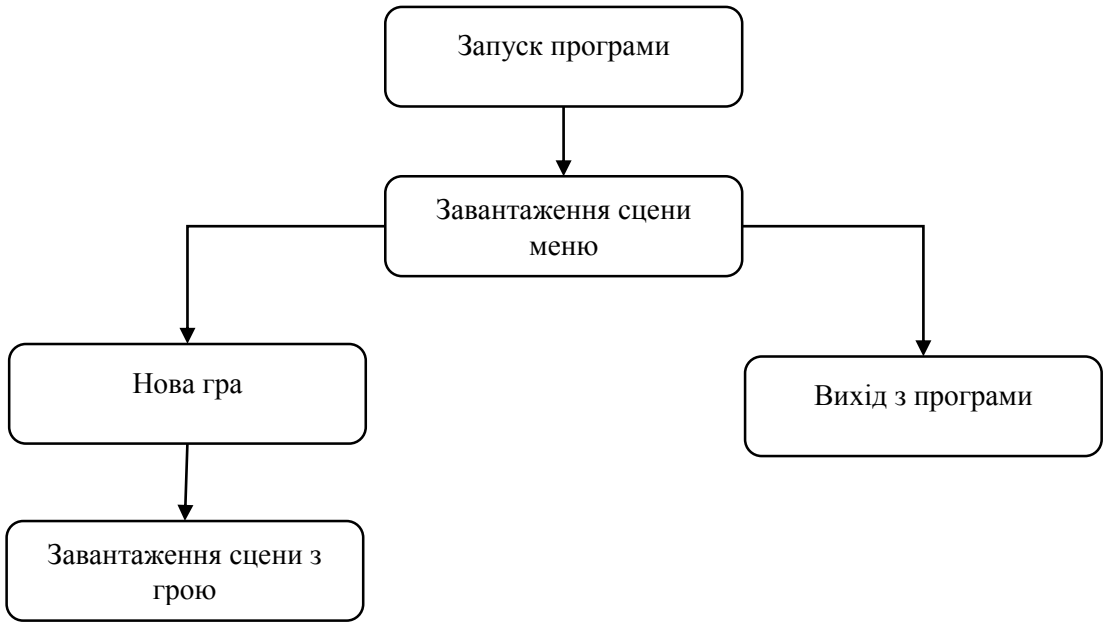

Рис. 2.1.Схема взаємодії з головним меню

2.3 Проектування схеми алгоритму ігрового процесу

Схема ігрового процесу буде реалізовано таким чином:

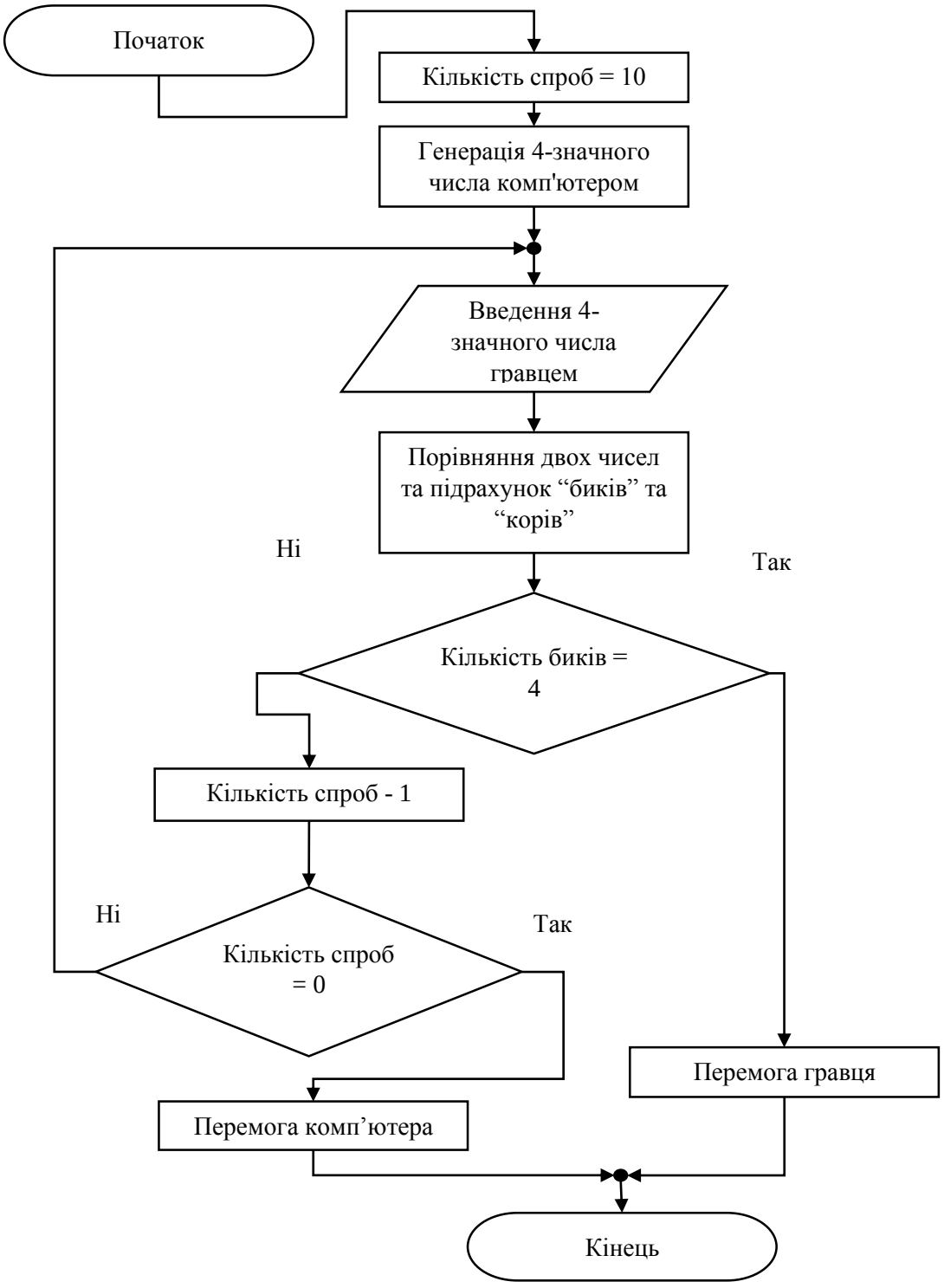

Рис. 2.2 Загальна блок-схема ігрового процесу

2.4 Висновки

Отже, В даному розділі були спроектовані основні елементи програми, які треба буде реалізувати в середовищі Unity. Розробили загальні блок-схеми ігрового процесу гри та схему взаємодії з ігровим меню.

Розділ 3. Реалізація і тестування програми.

На основі спроектованих елементів та схем розробимо програмну реалізацію гри, опишемо функції та методи, створимо головне меню та інтерфейс для користувача.

3.1 Процес розробки ігрового меню

Спочатку створюємо проект Unity. С самого початку створюється сцена Unity, додаємо в неї об'єкт Canvas, в якому вже додаємо компонент image, який стане нашим фоном в меню.

Для доповнення головного меню візьмемо декілька спрайтів та розмистимо їх поверх Canvas.

Після цього створюємо дві кнопки Play та Quit, на які буде прикріплено скріпт Game_Behaviour з функціями NewGame() і QuitGame() відповідно.

Результат можете спостерігати на рис 3.1.

Рис. 3.1. Головне меню програми

Лістинг скріпта має наступний вигляд:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class Main_Menu : MonoBehaviour
{

    public void NewGame()
    {
      SceneManager.LoadScene("Game");
    }

    public void QuitGame()
    {
       Application.Quit();
    }
}
```

Метод SceneManager.LoadScene() – завантажує відповідну сцену Game

Application.Quit() – здійснює вихід з додатка.

3.2 Процес розробки інтерфейса під час гри

Створюємо нову сцену Game. Також додаємо в неї об'єкт Canvas з компонентом image. Для покращення якості інтерфейса гри, будемо використовувати покращений текст TextMeshPro та піксельний шрифт задля дотримання стилістики гри. Додамо на сцену декілька текстів для відображення різної інформації одночасно. Після цього створюємо поле введення для гравца – InputField – TextMeshPro, та кнопки Enter, Back та Replay. Див. рис. 3.2.

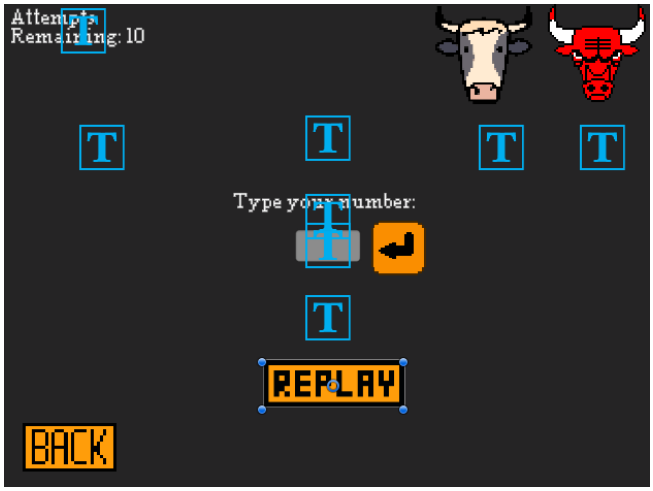

Рис. 3.2. Вигляд інтерфейсу на сцені

Кнопка Replay, як і декілька текстів будуть відображатись тільки після перемоги та поразки гравця. Див рис. 3.3

Рис. 3.3. Вигляд інтерфейсу на під час ігрового процесу

3.3 Опис методів і функцій

Створюємо новий скріпт Game_Behaviour, де реалізуємо наступні методи:

Start() – Функція, яка викликається одразу ж після завантаження сцени. Реалізована в цій программі для першої генерації таємного числа комп’ютером, див рис. 3.4.

```
public void Start()
{
    //Assignment of 9 elements in an array
    for (int i = 0; i < computerNumber.Length; i++)
    {
     computerNumber[i] = i;
    }
    //Creating a four-digit random number
    for(int i = 0; i < 4; i++)
    {
       Swap(ref computerNumber[i], ref computerNumber[UnityEngine.Random.Range(i,10)]);
    } //Checking to zero as the zero element of the array
    if (computerNumber[0] == 0) {Swap(ref computerNumber[0], ref computerNumber[UnityEngine.Random.Range(4,10)]);}
}
```

Рис. 3.4. Лістинг функції Start()

Replay() – Функція, яка обнуляє статистику і генерує нове випадкове 4-значне число при натисканні на кнопку Replay, див рис. 3.5.

```
public void Replay() //Replay button, reset statistics if pressed and generate a new random four-digit number
{
    bulls = 0; cows = 0; attempts = 10;
    countAttemptText.text = "Attempts Remaining: " + attempts;
    cowCountText.text = "";
    bullCountText.text = "";
    countNumberText.text = "";

    errorText.text = "";
    inputField.text = "";
    for(int i = 0; i < 4; i++)
    {
       Swap(ref computerNumber[i], ref computerNumber[UnityEngine.Random.Range(i,10)]);
    }
    if (computerNumber[0] == 0) {Swap(ref computerNumber[0], ref computerNumber[UnityEngine.Random.Range(4,10)]);}
    enterButton.interactable = true;
    retryButton.gameObject.SetActive(false);
}
```

Рис. 3.5. Лістинг функції Replay()

Swap() – Допоміжна функція для генерації числа функціями Replay() та Start(), див. рис. 3.6.

```
public void Swap(ref int firstElement, ref int secondElement)
{
    var temp = firstElement;
    firstElement = secondElement;
    secondElement = temp;
}
```

Рис. 3.6. Лістинг функції Swap()

Update() – Функція, яка викликається каждий кадр (фрейм). Реалізована в цій программі для перевірки заборонених символів в полі введення. Так як середовище Unity дає змогу заборонити більшу частину символів виставивши тип вхідних даних Integer в полі введення, було прийнято рішення заборонити тільки один символ, а саме «-». Див рис 3.7.

```
public void Update() //Checking for illegal character
{
    if (inputField.text.Contains('-'))
    {
        inputField.text = inputField.text.Replace("-", "");
    }
}
```

Рис. 3.7. Лістинг функції Update()

MenuLoad() – Функція, яка завантажує сцену меню при натисканні на кнопку Back. Див рис 3.8.

```
public void MenuLoad() //Back button and menu loading if pressed
{
    SceneManager.LoadScene("Main_menu");
}
```

Рис. 3.8. Лістинг функції MenuLoad()

GroupBy().Count() – Метод, для групування елементів за певними параметрами. відповідно до заданої функцією селектора ключа і створює результуюче значення для кожної групи і її ключа. Я ж створив групу всіх елементів по самому елементу, і отримую колекцію ключових значень, що дає змогу легко перевіріти на унікальність кожної цифри в числі.

Take(4).Contains() – Метод, який розглядує тільки перщі 4 елементи в групі, та повертає True чи False, в залежності від наявності елемента, який потрібно знайти.

TestFunction() – Головна функція, яка виконує основну логіку програми. Вона приймає число гравця з поля введення, перевіряє на унікальність кожної цифри в числі, порівнює обидва числа між собою, підраховує кількість биків та корів та показує результат гри в залежності від статистики. Див рис. 3.9

```
public void TestFunction() //enterButton function, Program basic logic
    {
        bulls = 0; cows = 0;
        int[] number = inputField.text.ToCharArray().Select(i =>
int.Parse(i.ToString())).ToArray(); //Getting a number from a player and checking that it
meets the conditions.
        if(number.Length == 4 && number.GroupBy(x=>x).Count() == 4) //Using LINQ methods
to simplify code readability, its implementation and thus to make code errors less
probable.
        {
        errorText.text = "";
        bulls = 0; cows = 0;
        for (int i = 0; i < 4; i++) //Comparison of the number of the player and the
computer's mystery number, cow and bull count
        {
            if (computerNumber[i] == number[i]) bulls++;
            else if (computerNumber.Take(4).Contains(number[i])) cows++;
        }
            cowCountText.text +=  "Cows= " + cows + "\n";   //Cow and bull count output
            bullCountText.text += "Bulls= " + bulls + "\n";
            for(int i = 0; i < 4; i++){
            countNumberText.text += number[i];
            }
            countNumberText.text += "\n";
//Checking to see if the player has won and outputting data if this is the case.
            if (bulls == 4) {
                winLoseText.text = "You won!";
                enterButton.interactable = false;
                retryButton.gameObject.SetActive(true);
                }
// Checking if the player's number of attempts has ended and outputting data if this is
the case.
            else {
                  attempts--;
                  countAttemptText.text = "Attempts Remaining: " + attempts;
                  if (attempts == 0)
                  {
                  winLoseText.text = "You Lost!";
                  enterButton.interactable = false; //Buttons Disabling & Enabling
                  retryButton.gameObject.SetActive(true);


                  }
                }
        }
        else
        { //Message output if a number is entered incorrectly.
            errorText.text = "Type 4 different digits!";
        }
       }
```

Рис. 3.9. Лістинг функції TestFunction().

3.4 Тестування

Ігрове меню працює коректно та відповідає усім вимогам. див. рис. 3.10.

Рис. 3.10. Ігрове меню під час ігрового процесу

Тестування програми пройшло успішно, програма коректно працює, отримуючи число від користувача, порівнюючи його зі своїм та виводячи точну статистику, як це показано на рис. 3.11 та рис. 3.12.

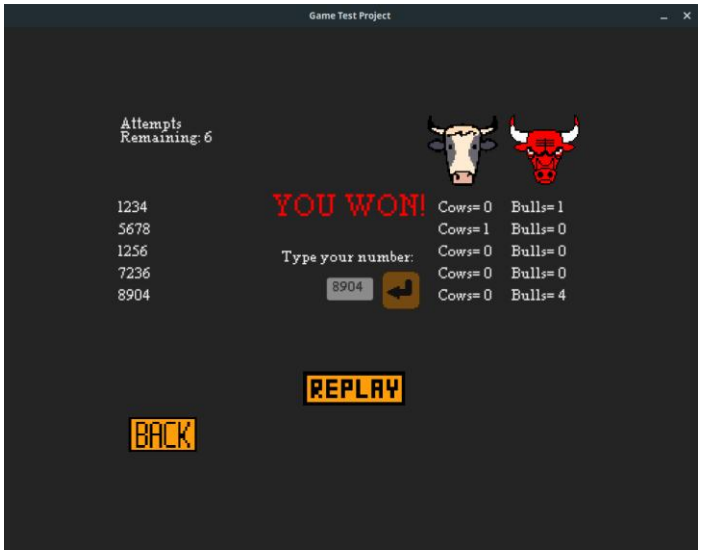

Рис. 3.11. Перемога в грі Бики і Корови

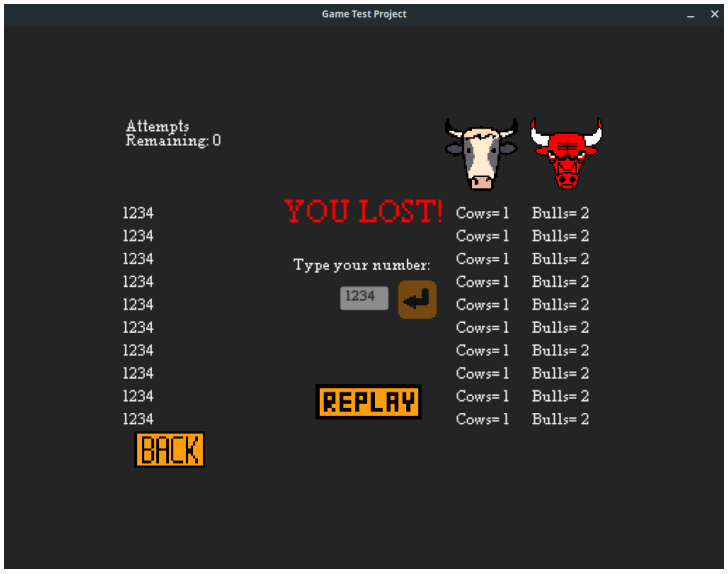

Рис. 3.11. Поразка в грі Бики і Корови

3.5 Висновки

Отже, ми реалізували ігрове меню, достатньо зрозумілий та зручний інтерфейс під час гри, створили скріпт, в якому реалізували всі необхідні функції та методи. Також програма успішно пройшла тестування на неправильні та правильні вхідні дані.

Висновок

Під час роботи була створена логічна комп’ютерна гра “Бики та Корови”. Програма на даний момент володіє простим та зручним інтерфейсом і відносно непоганим головним меню.

У подальшому можливо реалізувати такі можливості:

- Гра с другом, а не тільки з комп’ютером.
- Музика
- Більше режимів гри.
- Більш кращу графіку
- Меню паузи
- Таблиця рекордів

З недоліків можу сказати тільки про те, що гра доволі швидко наскучає любому гравцеві за досить короткий час. Також гравець має досить мало можливостей в ній, але це скоріше пов’язано з самою концепцією гри.

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.-32с.
2. Документація по C# [Електронний ресурс] Режим доступу: https://docs.microsoft.com/ru-ru/dotnet/csharp/ Перевірено 23.05.2020
3. Ще більше документації по C# [Електронний ресурс] Режим доступу: https://metanit.com/sharp/ Перевірено 23.05.2020
4. Документація по Unity3D [Електронний ресурс] Режим доступу: https://docs.unity3d.com/Manual/index.html Перевірено 24.05.2020
5. Форум з матеріалами по C# [Електронний ресурс] Режим доступу: https://habr.com/ru/search/ Перевірено 23.05.2020
6. Відеоматеріали по Unity3D та C# [Електронний ресурс] Режим доступу: https://www.youtube.com/ Перевірено 24.05.2020