

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**🙠🙟🕮🙝🙢**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, icon

Description automatically generated

**BÁO CÁO ĐỒ ÁN CUỐI KÌ**

**MÔN HỌC: ĐỒ ÁN CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI: XÂY DỰNG GAME ĐOÁN SỐ**

**GVHD : THS. TỪ TUYẾT HỒNG**

**SINH VIÊN THỰC HIỆN : MAI TẤN TÀI 21110851**

**MÃ HỌC PHẦN : PROJ215879\_23\_1\_10CLC**

*TP.Hồ Chí Minh, tháng 12 năm 2023*

*Thủ Đức, tháng 05 năm 2022*

*Thủ Đức, tháng 07 năm 2022*

*Thủ Đức, tháng 05 năm 2022*

**NHẬN XÉT CỦA GIẢNG VIÊN**

***TP.Hồ Chí Minh, ngày … tháng … năm 2023***

**Giảng viên chấm điểm**

**……………………….**

**MỤC LỤC**

[CHƯƠNG I: TỔNG QUAN VỀ ĐỀ TÀI 7](#_Toc152602116)

[**1.1** **Đặc tả của đề tài** 7](#_Toc152602117)

[**1.2** **Cách thức trò chơi hoạt động** 7](#_Toc152602118)

[**1.3** **Giao diện người chơi dự kiến** 7](#_Toc152602119)

[**1.3.1** **Màn hình Main menu** 7](#_Toc152602120)

[**1.3.2** **Màn hình Game Playing** 8](#_Toc152602121)

[**1.4** **Logic trò chơi và xử lí sự kiện** 8](#_Toc152602122)

[CHƯƠNG II: MÔI TRƯỜNG LÀM VIỆC 10](#_Toc152602123)

[2.1 Ngôn ngữ lập trình C# 10](#_Toc152602124)

[2.1.1 Tổng quan 10](#_Toc152602125)

[2.2.2 Lịch sử phát triển 10](#_Toc152602126)

[2.2.3 Ưu điểm 10](#_Toc152602127)

[2.2.4 Nhược điểm 11](#_Toc152602128)

[2.2 Công cụ lập trình Unity 12](#_Toc152602129)

[2.2.1 Tổng quan 12](#_Toc152602130)

[2.2.2 Một số ưu điểm của lập trình game Unity 12](#_Toc152602131)

[CHƯƠNG III: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 14](#_Toc152602132)

[3.1 Yêu cầu chức năng 14](#_Toc152602133)

[3.2 Yêu cầu phi chức năng 14](#_Toc152602134)

[3.2.1 Khả năng sử dụng 14](#_Toc152602135)

[3.2.2 Khả năng bảo trì và khả năng mở rộng 14](#_Toc152602136)

[CHƯƠNG IV: THIẾT KẾ 15](#_Toc152602137)

[4.1 Xây dựng giao diện: 15](#_Toc152602138)

[4.1.2 Xây dựng giao diện Main Menu 15](#_Toc152602139)

[4.1.3 Xây dựng Window Instructions 15](#_Toc152602140)

[4.1.4 Xây dựng giao diện Game Playing: 16](#_Toc152602141)

[4.1.5 Xây dựng Window Pause Game 17](#_Toc152602142)

[4.1.6. Xây dựng Window Continue Quit 17](#_Toc152602143)

[4.1.7. Xây dựng Window Game Over: 18](#_Toc152602144)

[4.1.8 Xây dựng Window You Win 19](#_Toc152602145)

[4.2 C# Script 19](#_Toc152602146)

[4.2.1 MenuManager 19](#_Toc152602147)

[4.2.2 GameManager 20](#_Toc152602148)

[4.2.3 AttackOrc 27](#_Toc152602149)

[4.2.4 KeyInputHandler 30](#_Toc152602150)

[4.2.5 Answer 31](#_Toc152602151)

[4.2.6 MovePrincess 33](#_Toc152602152)

[4.2.7 SavingData 36](#_Toc152602153)

[4.2.8 DialogResult 37](#_Toc152602154)

[CHƯƠNG V: CÀI ĐẶT VÀ KIỂM THỬ 39](#_Toc152602155)

[6.1 Kết luận 40](#_Toc152602156)

[6.2 Ưu điểm và nhược điểm 40](#_Toc152602157)

[6.2.1 Ưu điểm 40](#_Toc152602158)

[6.2.2 Nhược điểm 40](#_Toc152602159)

[6.3 Hướng phát triển 41](#_Toc152602160)

[6.4 Các khó khăn gặp phải 41](#_Toc152602161)

[TÀI LIỆU THAM KHẢO 42](#_Toc152602162)

**LỜI MỞ ĐẦU**

Game trên máy tính hay thiết bị di động là trò chơi giải trí đang đưuọc ưu chuộng mạnh, nhất là trong những năm gần đây, khi mà thiết bị di động thông minh được phát triển rất mạnh mẽ. Tổng giá trị thị trường game ở Hàn Quốc là 9,16 tỷ USD (tương đương 180 ngàn tỷ đồng) và ở Nhật Bản được ước tính giá trị đó gấp 7 lần. Và Việt Nam là nước có thị trường game khá non trẻ, tuy nhiên Việt Nam cũng bắt đầu hội nhập với một số hãng game lớn đã xuất khẩu ra thị trường thế giới điển hình như là VinaGame. Ngoài ra, với sự tìm tòi sáng tạo của các cá nhân cũng dã tự viết một số game từ cơ bản đến phúc tạp để đáp ứng nhu cầu của bản thân cũng như nhu cầu của xã hội.

Hiện nay với cường độ làm việc và học tập ngày càng cao thì nhu cầu giảm stress, thư giản đầu óc cũng như nhu cầu giải trí của con người ngày càng được chú tâm và nâng cao hơn. Để đáp ứng yêu cầu, mong muốn của xã hội cũng như tạo ra công cụ giải trí cho con người. Thì em đã đưa ra quyết định chọn đề tài cuối kì cho môn Đồ Án Công Nghệ Thông Tin là tạo ra một game nhỏ, đơn giản nhằm đáp ứng nhu cầu giải trí cho con người và game đó có tên là “Đoán Số”.

**LỜI CẢM ƠN**

Game Đoán Số là một trò chơi giải trí cho phép người chơi thỏa sức tư duy để có thể tìm kiếm ra được những con số bí ẩn một cách nhanh chóng và chính xác nhất. Với giao diện bắt mắt, thân thiện, dễ dàng sử dụng tạo cho người chơi dễ dàng tiếp cận gia tăng sự thích thú và có thể dùng để thư giản sau nhiều giờ làm việc mệt mỏi. Mong rằng với đồ án nhỏ của em sẽ có thể đóng góp được một phần nào đó vào phát triển các game hiện hành ngày nay.

Để hoàn thành tốt đề tài này, em xin gửi lời cảm ơn chân thành đến giảng viên, ThS Từ Tuyết Hồng, người đã trực tiếp hỗ trợ em trong suốt quá trình làm đề tài. Em cảm ơn cô đã đưa ra những lời khuyên từ kinh nghiệm thực tiễn của mình để định hướng cho em đi đúng với yêu cầu của đề tài đã chọn, luôn giải đáp thắc mắc và đưa ra những góp ý, chỉnh sửa kịp thời giúp em khắc phục nhược điểm và hoàn thành tốt cũng như đúng thời hạn đã đề ra.

Đề tài được em thực hiện trong khoảng thời gian ngắn, với những kiến thức còn hạn chế cùng nhiều hạn chế khác về mặt kỹ thuật và kinh nghiệm trong việc thực hiện một dự án lập trình game. Do đó, trong quá trình làm nên đề tài có những thiếu sót là điều không thể tránh khỏi nên em rất mong nhận được những ý kiến đóng góp quý báu của các quý thầy cô để kiến thức của em được hoàn thiện hơn và em có thể làm tốt hơn nữa trong những lần sau.

Em xin chân thành cảm ơn!

# CHƯƠNG I: TỔNG QUAN VỀ ĐỀ TÀI

* 1. **Đặc tả của đề tài**

Mục đích: Thực hiện thiết kế và cài đặt phần mềm **Game Guessing Number.**

Trò chơi được phát triển với mục tiêu chính là tạo ra một trải nghiệm giải trí độc đáo và thú vị thông qua trò chơi đoán số. Vì là trò chơi với thuật toán đơn giản, nên ưu tiên hướng tới việc tạo ra một giao diện người chơi thân thiện, dễ sử dụng để tối ưu hóa trải nghiệm người dùng và đặc biệt tăng thêm sự thu hút người chơi. Phần mềm không chỉ đơn giản là một trò chơi giải trí mà còn là một công cụ hỗ trợ người chơi trong việc phát triển và cải thiện kỹ khả năng tư duy của họ.

* 1. **Cách thức trò chơi hoạt động**

Trò chơi có một giao diện được thiết kế sáng tạo, mang lại sự mới mẻ và thú vị. Khi bắt đầu mỗi vòng chơi, trò chơi sẽ ngẫu nhiên chọn một con số từ 1 đến 100, và người chơi sẽ có 10 lượt để đoán con số đó. Nhiệm vụ của người chơi là tìm ra con số bí mật này. Nếu con số mà người chơi chọn khác với con số của trò chơi, hệ thống sẽ đưa ra phản hồi, gợi ý xem con số đúng có lớn hơn hay nhỏ hơn con số trước đó. Như vậy, người chơi sẽ mất đi một lượt đoán. Khi người chơi sử dụng hết 10 lượt mà không tìm ra con số chính xác, trò chơi kết thúc.

Để làm cho trò chơi thêm phần hấp dẫn, giao diện được thiết kế dựa trên câu chuyện về một công chúa bị quái vật bắt cóc. Công chúa bị bắt giam trong lồng và được treo trên một sợi dây và quái vật đang chuẩn bị giết công chúa. Để giải cứu công chúa, hiệp sĩ phải đoán đúng con số mà quái vật đưa ra. Sau mỗi 2 lần hiệp sĩ đoán sai con số thì quái vật sẽ từ từ hạ dây xuống gần một lò lửa.

* 1. **Giao diện người chơi dự kiến**

Về phần giao diện thì trò chơi được thiết kế, xây dựng bao gồm hai màn hình chính là ***Main menu*** và ***Game Playing***

* + 1. **Màn hình Main menu**

Đây là giao diện bắt đầu của trò chơi. Tại màn hình *Main menu* được thiết kế chủ yếu để xử lý các nút tùy chọn. Thiết kế Main Menu bao gồm ba nút tùy chọn:

**New Game** (Trò Chơi Mới): Bấm vào nút này để bắt đầu một vòng chơi mới, nơi một con số ngẫu nhiên sẽ được chọn và bạn sẽ có 10 lượt để đoán.

**Instructions** (Hướng Dẫn): Bấm vào nút này để xem hướng dẫn cách chơi và các quy tắc của trò chơi.

**Quit** (Thoát): Bấm vào nút này để thoát khỏi trò chơi.

* + 1. **Màn hình Game Playing**

Đây là màn hình chính xử lý logic trò chơi, màn hình được thiết kế với các yếu tố sau:

**Hiệp Sĩ** (Player): Hiển thị hình ảnh của hiệp sĩ, người chơi trong câu chuyện. Có thể sử dụng hình ảnh hoặc biểu tượng đại diện cho hiệp sĩ. Hiệp sĩ sẽ có một bóng chat để đưa ra câu trả lời.

**Quái Vật**: Hiển thị hình ảnh hoặc biểu tượng đại diện cho quái vật giữ công chúa. Quái vật đại diện cho thách thức mà người chơi phải vượt qua bằng cách đoán đúng con số. Quái vật có một bóng chat để đưa ra các thông báo của trò chơi.

**Pause Game Button** (Nút Tạm Dừng Trò Chơi): Nút này cho phép người chơi tạm dừng trò chơi và quay trở lại màn hình Main Menu hoặc cài đặt khác.

**Heart Icon** (Biểu Tượng Trái Tim): Hiển thị số lượt đoán còn lại của người chơi. Mỗi lượt đoán không thành công sẽ làm giảm số lượng trái tim, và khi số lượng này giảm về 0, trò chơi kết thúc. Biểu tượng trái tim thường được sắp xếp theo dạng "♥ X" (với X là số lượng trái tim còn lại).

Ngoài ra, trò chơi còn có các đối tượng để phục vụ cho giao diện và các Dialog như Pause, Game Over, Win để thông báo trạng thái của trò chơi cùng với các thông số khác, các Dialog cũng có các nút tùy chọn cho trò chơi.

* 1. **Logic trò chơi và xử lí sự kiện**

- Logic trò chơi:

+ Tạo Số Ngẫu Nhiên: Trò chơi tạo số ngẫu nhiên và cách nó được tính toán để đảm bảo tính ngẫu nhiên và công bằng.

+ Kiểm Tra Đoán Số: Trò chơi kiểm tra đoán số của người chơi, bao gồm quy tắc để

xác định liệu đoán là cao hơn, thấp hơn hay chính xác.

+ Điều Kiện Chiến Thắng hoặc Kết Thúc Trò Chơi: Điều kiện cần đạt để chiến thắng và cách trò chơi kết thúc nếu người chơi không đạt được mục tiêu.

- Xử lý sự kiện:

+ Sự Kiện Người Dùng: Trò chơi xử lý sự kiện từ người chơi, chẳng hạn như khi họ nhấp nút đoán hay nhập số vào ô.

+ Phản Hồi và Cập Nhật Giao Diện: Mô tả cách phản hồi từ logic trò chơi được chuyển đến giao diện người dùng, bao gồm cập nhật kết quả và hiển thị thông báo

# CHƯƠNG II: MÔI TRƯỜNG LÀM VIỆC

## 2.1 Ngôn ngữ lập trình C#

### 2.1.1 Tổng quan

C# (C Sharp) là một dạng ngôn ngữ lập trình hiện đại thường được sử dụng trên Windows .NET framework hoặc một số nền tảng mã nguồn mở. Với khả năng thực hiện nhiều công việc trong mọi lĩnh vực khác nhau nên C# được xem là một ngôn ngữ lập trình hướng đối tượng OOP khá linh hoạt và đáng tin cậy.

### 2.2.2 Lịch sử phát triển

Năm 2000, một kỹ sư Đan Mạch có tên Microsoft’s Anders Hejlsberg đã sáng tạo ra C#. Ngoài ra, ông còn nổi tiếng về các phát minh nổi bật như tạo ra các công cụ và ngôn ngữ lập trình đáng tin cậy gồm Microsoft’s TypeScript và Delphi, một sự thay thế phù hợp cho Turbo Pascal.

Tháng 2 năm 2019, ngôn ngữ lập trình C# đã đứng hạng thứ 4 vượt PYPL và chỉ đứng sau Java và [JavaScript](https://bkhost.vn/blog/javascript-js/). Trong đó, [dữ liệu](https://bkhost.vn/blog/data-du-lieu/) chính được sử dụng để biên dịch chỉ mục này dựa trên tần suất tìm kiếm hướng dẫn về các ngôn ngữ lập trình khác nhau trong Google.

Hiện nay, C# ngày càng trở nên phổ biến trong danh sách mười ngôn ngữ lập trình hàng đầu của TIOBE Index được chứng minh từ các công cụ tìm kiếm phổ biến như Google, YouTube và Bing.

## 2.2.3 Ưu điểm

- Đồng nhất với nền tảng Microsoft:

C# được phát triển bởi Microsoft, nó hoàn toàn tương thích với các sản phẩm và dịch vụ của họ, đặc biệt là trên nền tảng .NET. Điều này giúp tối ưu hóa tích hợp với hệ sinh thái của Microsoft.

- Quản lý bộ nhớ tự động:

C# sử dụng Garbage Collection (GC) để quản lý bộ nhớ tự động, giúp giảm áp lực đối với lập trình viên khi không còn cần quản lý bộ nhớ thủ công.

- Ngôn ngữ hướng đối tượng:

C# hỗ trợ lập trình hướng đối tượng, giúp tạo ra mã nguồn có cấu trúc, dễ đọc và dễ duy trì.

- Thư viện làm việc phong phú:

C# kết hợp chặt chẽ với .NET Framework, mang lại một loạt các thư viện và công cụ mạnh mẽ, giúp lập trình viên tăng hiệu suất và giảm thời gian phát triển.

- Đa nhiệm và đa luồng:

C# hỗ trợ đa nhiệm và đa luồng, cho phép xử lý đồng thời nhiều công việc, cải thiện hiệu suất ứng dụng.

- An toàn kiểu dữ liệu:

Có kiểu dữ liệu mạnh mẽ và kiểm soát kiểu dữ liệu tốt, giúp tránh được nhiều lỗi liên quan đến kiểu dữ liệu trong quá trình biên dịch.

- Phát triển ứng dụng web và desktop

C# không chỉ hỗ trợ phát triển ứng dụng desktop mà còn là lựa chọn phổ biến cho phát triển ứng dụng web thông qua ASP.NET.

### 2.2.4 Nhược điểm

- Giới hạn trên nền tảng Microsoft:

C# và .NET Framework tập trung chủ yếu vào hệ điều hành Windows, điều này có thể là một hạn chế đối với những ứng dụng cần chạy trên các nền tảng khác.

- Tính di động:

So với một số ngôn ngữ lập trình khác, C# có thể có tính di động kém do phụ thuộc nhiều vào môi trường Microsoft.

- Tốc độ chậm hơn so với ngôn ngữ biên dịch cao cấp khác:

Trong một số trường hợp, C# có thể chậm hơn so với các ngôn ngữ biên dịch cao cấp khác như C++ do quá trình Garbage Collection có thể tạo ra đợt gián đoạn không hiệu quả.

- Khả năng quản lý thấp trực tiếp bộ nhớ:

Mặc dù có Garbage Collection, nhưng C# không cung cấp khả năng quản lý trực tiếp bộ nhớ như một số ngôn ngữ khác, điều này có thể làm giảm hiệu suất trong một số ứng dụng yêu cầu quản lý bộ nhớ chặt chẽ.

## 2.2 Công cụ lập trình Unity

### 2.2.1 Tổng quan

Lập trình game với Unity là phần mềm làm game sử dụng nhiều nền tảng khác nhau, các nền tảng được hỗ trợ hiện nay là IOS, macOS, Android, Linux Widows, Windows Phone.,…

Unity mang đến hệ thống toàn diện cho các lập trình viên, từ soạn thảo văn bản, xây dựng công cụ tự động hóa đến quá trình sửa lỗi nên cũng khá dễ sử dụng. Ngôn ngữ lập trình chính mà Unity sử dụng đó là C#, ngoài ra còn được hỗ trợ bởi Javascript

Unity tận cùng các chức năng của thư viện phần mềm như engine mô phỏng vật lý Physx của Opengl, Nvidia, và cả Directx 3D,…để tạo nên những hình ảnh 3D, OpenAL cho âm thân,… Vì thế nó hỗ trợ rất mạnh cho việc lập trình game.

### 2.2.2 Một số ưu điểm của lập trình game Unity

- Phần mềm sử dụng ngôn ngữ lập trình phổ biến là C#, giúp người dùng sử dụng một cách dễ dàng

- Lập trình game Unity hỗ trợ đa dạng các nền tảng khác nhau. Nhờ đó giúp bạn rút ngắn được thời gian xây dựng và sáng tạo game. Bên cạnh đó còn giúp nâng cao mức độ an toàn khi lập trình game.

- Một số nền tảng mà Unity cung cấp đó là Android, PlayStation 3, Windows, Xbox 360, OS X, Wii U, Blackberry 10, Linux, trình duyệt web.

- Unity rất dễ sử dụng, ngay cả khi bạn lập trình nghiệp dư. Unity cung cấp hệ thống toàn diện từ soạn thảo mã nguồn, sửa lỗi cho các lập trình game cho đến xây dựng công cụ tự động hóa.

- Phần mềm mang lại hiệu quả kinh tế khá cao, tiết kiệm chi phí đối với hầu hết các doanh nghiệp

- Unity tích hợp thư viện đa dạng, các công cụ hỗ trợ làm game nhanh chóng, phong phú và vô cùng thông minh. Nhờ đó các lập trình viên về game có thể thỏa sức sáng tạo và tạo nên những sản phẩm tuyệt vời cho bản thân.

- Unity còn hỗ trợ Networking để phát triển dòng MMO game

- Unity là phần mềm được nhiều người sử dụng và khá lớn mạnh trên thị trường quốc tế. Vì thế mọi vướng mắc của người dùng đều sẽ nhanh chóng được giải quyết để gia tăng trải nghiệm cho người dùng.

# CHƯƠNG III: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## 3.1 Yêu cầu chức năng

Hiển thị đúng kết quả: Kết quả sau khi đoán đúng phải là 1 số đúng trong khoảng xác định.

Sự ngẫu nhiên: Số cần phải đoán là 1 số ngẫu nhiên, không có quy luật trong khoảng xác định.

Chơi lại: Người chơi có thể thử lần chơi mới khi có thông báo Game Over hay You Win, ngoài ra trong quá trình chơi người chơi cũng có thể tải câu đố mới và lượt đó không được tính vào số lượt đã chơi.

Tạm dừng trò chơi: Người chơi có thể tạm dừng trò chơi bất cứ lúc nào.

## 3.2 Yêu cầu phi chức năng

### 3.2.1 Khả năng sử dụng

- Giao diện người dùng phải thân thiện, đơn giản, logic vầ dễ sử dụng.

- Phải có đa dạng nhiều cấp độ chơi để tăng sự thu hút đối với người chơi.

### 3.2.2 Khả năng bảo trì và khả năng mở rộng

- Mã nguồn phải tuân theo quy ước viết mã để giúp các nhà phát triển đọc, hiểu và dễ bảo trì hoặc sửa đổi mã.

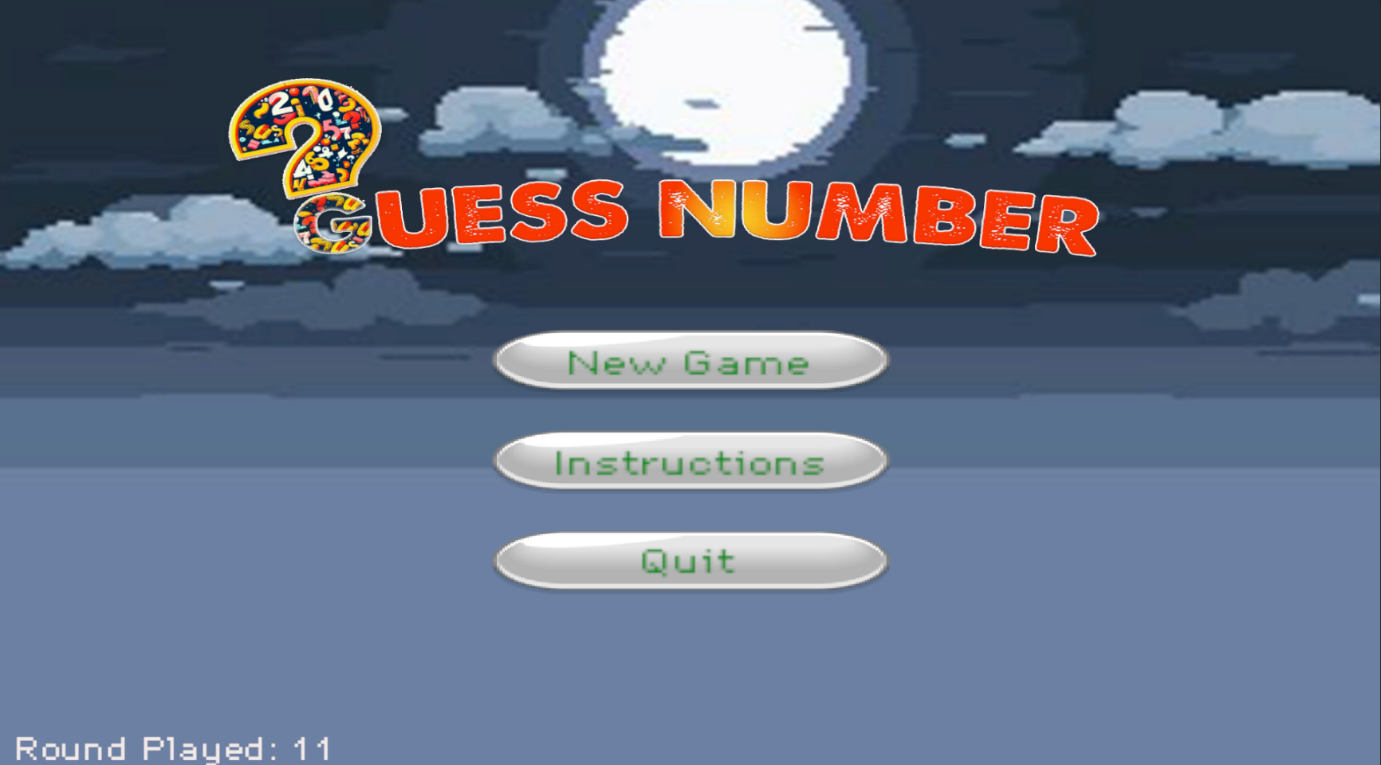
- Nhật ký hệ thông báo lỗi để theo dõi lỗi mỗi khi ứng dụng xảy ra sự cố.

# CHƯƠNG IV: THIẾT KẾ

## 4.1 Xây dựng giao diện:

Trước tiên cần phải xây dừng giao diện trò chơi và cần liệt kê ra tất cả các chức năng của trò chơi để có thể biết trò chơi cần những thuật toán gì và tạo ra những Class phù hợp.

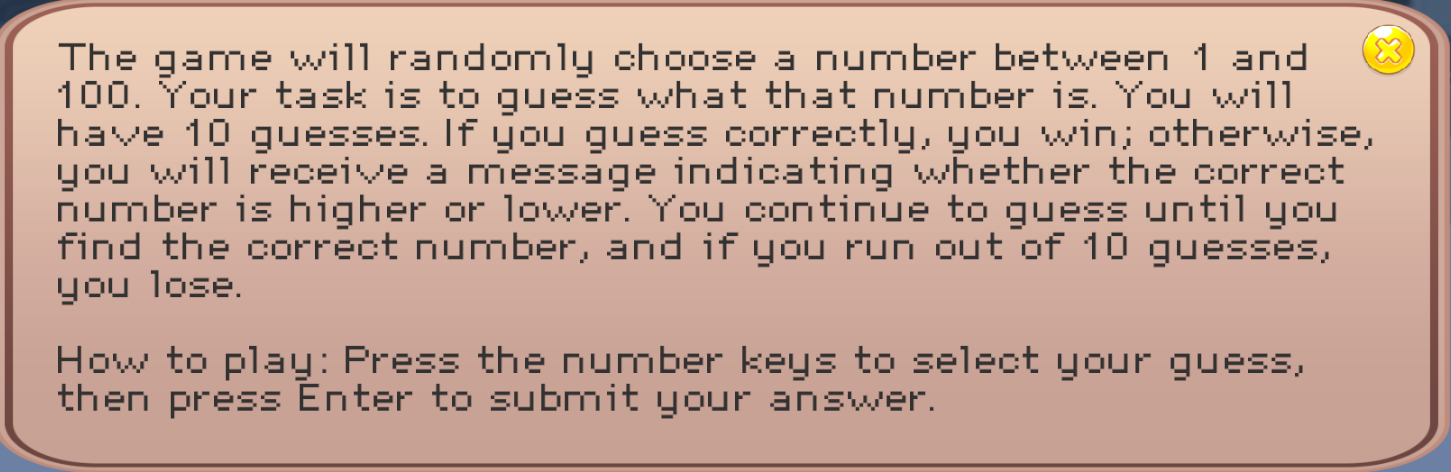
### 4.1.2 Xây dựng giao diện Main Menu



***Hình ảnh Màn hình diện Main Menu***

Vì đây là giao diện chuyển đổi trạng thái của trò chơi nên cần phải có các nút cơ bản của trò chơi khi bắt đầu, trò chơi cũng sẽ cập nhật số lượt chơi mà người chơi đã chơi.

### 4.1.3 Xây dựng Window Instructions

****

***Hình ảnh Window Instructions***

Cửa sổ cần phải thể hiện một cách ngắn gọn và đầy đủ ý chính về giới thiệu trò chơi và luật chơi đồng thời cũng phải hướng dẫn các phím đưa ra con số dự đoán và nộp câu trả lời cho hệ thống xử lý.

Cửa sổ phải có một nút Quit để quay lại màn hình Main Menu sau khi đã đọc hướng dẫn.

### 4.1.4 Xây dựng giao diện Game Playing:

****

***Hình ảnh Màn hình Game Playing***

Để có thể thu hút người chơi, giao diện phải được thiết kế hài hòa, mặc khác cần có sự sáng tạo, đổi mới để không nhàm chán. Màn hình cần phải hiển thị đầy đủ các thông báo, câu trả lời và số lượt đoán còn lại của người chơi.

Trong thiết kế trên của tôi, tôi đã xây dựng Icon Heart để hiển thị số lượt đoán còn lại. Bong bóng chat của hiệp sĩ để người chơi có thể thấy họ đang nhập con số nào vào trò chơi. Bong bóng chat của quái vật là thông báo mà trò chơi sẽ đưa ra cho chúng ta khi ta đưa ra câu trả lời. Bên cạnh đó để tăng thêm sự thú vị thì tôi đã thiết kế thêm một sợi dây treo lồng giam giữ công chúa trên một nồi lửa, sau mỗi 2 lần người chơi đoán sai sợ dây treo sẽ từ từ di chuyển xuống cho đến khi công chúa bị thiêu rụi.

Ngoài ra, tôi đã thêm những vật thể khác để giao diện thêm sinh động.

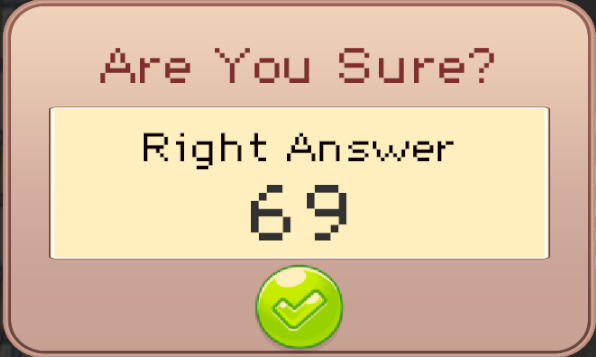
### 4.1.5 Xây dựng Window Pause Game

***Hình ảnh Window Pause Game***

Cửa sổ Pause Game cần phải hiển thị tên của cửa sổ, số lượt chơi hiện tại mà người chơi còn lại. Cửa sổ cũng có các nút tùy chọn bao gồm tiếp tục trò chơi, bắt đầu một trò chơi mới và thoát khỏi trò chơi. Với trường hợp người chơi muốn kết thúc trò chơi trong khi trò chơi chưa kết thúc thì trò chơi sẽ hiển thị một cửa sổ để người chơi xác nhận tiếp tục thoát hoặc ở lại. Nếu tiếp tục thoát thì trò chơi sẽ hiển thị con số chính xác mà trò chơi đã tạo.

### 4.1.6. Xây dựng Window Continue Quit

***Hình ảnh Window Continue Quit***

Tại cửa sổ này người chơi sẽ có hay lựa chọn, nếu người chơi muốn tiếp tục thoát trò chơi thì chọn nút OK (dấu tick) ngay sau đó trò chơi sẽ hiển thị sang cửa sổ tiếp theo, nếu người chơi muốn hủy thì chọn nút Cancel (dấu X) để quay lại cửa sổ Pause Game.

***Hình ảnh Cửa sổ tiếp theo***

Tại cửa sổ tiếp theo của cửa sổ Continue Quit Game, người chơi có thể xem đán án chính xác của trò chơi và chọn nút OK để quay lại màn hình Main Menu

### 4.1.7. Xây dựng Window Game Over:

***Hình ảnh Window Game Over***

Cửa sổ Game Over cần phải hiển thị được tên của Window, đưa ra con số chính xác mà trò chơi đã tạo để hiển thị minh bạch cho người chơi. Cửa sổ cũng có các nút tùy chọn bao gồm bắt đầu trò chơi mới hoặc quay lại Main Menu.

### 4.1.8 Xây dựng Window You Win

***Hình ảnh Cửa sổ You Win***

Cửa sổ được thiết kế bao gồm tên của cửa sổ, thông số TurnRemaining để hiển thị số lượt đoán còn lại - để người chơi biết họ đã dùng bao nhiêu lượt để tìm ra câu trả lời chính xác, các nút tùy chọn bắt đầu trò chơi mới và quay lại Main Menu.

## 4.2 C# Script

### 4.2.1 MenuManager

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | MenuManger | Chuyển đổi các tùy chọn của trò chơi |

*Các phương thức trong Class: MenuManager*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | btnNewGameClick() | Khởi tạo trò chơi mới khi người chơi nhấn nút NewGam ở MainMenu |
| 2 | btnQuitClick() | Thoát khỏi trò chơi khi người chơi nhấn nút Quit ở MainMenu |
| 3 | btnInstructionsClick() | Hiển thị hướng dẫn của trò chơi khi người chơi nhấn nút Instructions ở MainMenu |

*Source Code*

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using System.Xml.Serialization;  using UnityEngine;  using UnityEngine.SceneManagement;  public class MenuManager : MonoBehaviour  {  public GameObject instructions;  public void btnNewGameClick()  {  SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex+1);  }  public void btnQuitClick()  {  Application.Quit();  }  public void btnInstructionsClick()  {  instructions.SetActive(true);  }  } |

### 4.2.2 GameManager

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | GameManger | Xử lý các logic và chuyển trạng thái trò chơi |

*Các phương thức có trong class GameManager*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | Void Start() | Khởi tạo parameter |
| 2 | Void Init() | Khởi tạo các thông số khi bắt bắt đầu trò chơi mới |
| 3 | Bool CheckAnswer(int number) | Xử lý logic khi người chơi đưa ra câu trả lời |
| 4 | Void GamePause() | Tạm dừng trò chơi khi người chơi nhấn nút Pause |
| 5 | Void Continue() | Tiếp tục trò chơi khi người chơi nhấn nút Continue trong cửa sổ Game Pause |
| 6 | IEnumerator checkGameOver(string lives) | Xử lý trò chơi khi người chơi thua cuộc |
| 7 | IEnumerator WinState(bool orcDead) | Xử lý trò chơi khi người chơi chiến thắng và sau khi hiệp sĩ đã hạ được quái vật. |
| 8 | Void PauseGameBackToHome() | Chuyển sang cửa sổ Continue Quit khi người chơi thoát khỏi trò chơi tại cửa sổ Game Pause |
| 9 | Void Update() | Update lại khi có cập nhật mới |
| 10 | Void checkDialogResult() | Xử lý trò chơi khi người chơi click nút OK hay Cancel tại cửa sổ Continue Quit |

*Source Code:*

|  |
| --- |
| using System;  using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.Events;  using UnityEngine.SceneManagement;  using UnityEngine.UI;  public class GameManager : MonoBehaviour  {  public Text str\_lives;  public Text prince\_talk;  public int quizz;  public int lives;  private int score;  public List<int> check;  public int Score { get => score; }  public AudioClip wrongAnswer;  public AudioClip correctAnswer;  public AudioClip burnAcid;  public AudioClip lose;  private AudioSource audioSource;  public GameObject gameOverDialog;  public GameObject gamePauseDialog;  public GameObject winDialog;  public GameObject dialog;  public GameObject princess;  public GameObject prince;  private Animator animator;  public Text rightAnswer;  public GameObject MainMenu;  public GameObject InsideDialog;  public GameObject btnOK2;  private SavingData saveData;  private void Start()  {  Init();  lives = int.Parse(str\_lives.text);  prince\_talk.text = "";  audioSource = this.GetComponent<AudioSource>();  saveData = this.GetComponent<SavingData>();  }  public void Init()  {  str\_lives.text = "10";  quizz = UnityEngine.Random.Range(1, 101);  GameObject princess = GameObject.Find("Princess");  GameObject GamePlayUI = GameObject.Find("GamePlayUI");  princess.transform.position = new Vector2(1920-250, 540);  gamePauseDialog.SetActive(false);  gameOverDialog.SetActive(false);  winDialog.SetActive(false);  AudioSource audioFire = GameObject.Find("Fire").GetComponent<AudioSource>();  audioFire.Play();  Time.timeScale = 1;  }  public string CheckAnswer(int number)  {  string msg = null;  if (number == 0) return msg;  if (number == quizz)  {  audioSource.clip = correctAnswer;  audioSource.volume = 0.5f;  audioSource.Play();  msg = "Nooooooo!";  }  else if (number < quizz)  {  audioSource.clip = wrongAnswer;  audioSource.volume = 0.5f;  audioSource.Play();  if (quizz - number > 10)  msg = "Too low!";  else  msg = "Higher!";  lives--;  str\_lives.text = lives.ToString();  }  else  {  audioSource.clip = wrongAnswer;  audioSource.volume = 0.5f;  audioSource.Play();  if (number - quizz > 10)  msg = "Too high!";  else  msg = "Lower!";  lives--;  str\_lives.text = lives.ToString();  }  if (lives == 0)  {  audioSource.clip = burnAcid;  audioSource.Play();  audioSource.clip = lose;  audioSource.Play();  }  return msg;  }  public void GamePause()  {  gamePauseDialog.SetActive(true);  Text numberOfGuess = GameObject.FindGameObjectWithTag("NumberOfGuess").GetComponent<Text>();  numberOfGuess.text = str\_lives.text;  AudioSource audioFire = GameObject.Find("Fire").GetComponent<AudioSource>();  audioFire.Pause();  Time.timeScale = 0;  }  public void Continue()  {  gamePauseDialog.SetActive(false);  AudioSource audioFire = GameObject.Find("Fire").GetComponent<AudioSource>();  audioFire.Play();  Time.timeScale = 1;  }  public IEnumerator checkGameOver(string lives)  {  if (lives == "0")  {  animator = prince.GetComponent<Animator>();  animator.SetTrigger("Dead");  yield return new WaitForSeconds(2);  gameOverDialog.SetActive(true);;  rightAnswer.text = quizz.ToString();  AudioSource audioFire = GameObject.Find("Fire").GetComponent<AudioSource>();  audioFire.Pause();  saveData.SaveData();  Time.timeScale = 0;  }  }  public IEnumerator WinState(bool orcDead)  {  if (orcDead)  {  yield return new WaitForSeconds(2);  winDialog.SetActive(true);  AudioSource audioFire = GameObject.Find("Fire").GetComponent<AudioSource>();  audioFire.Pause();  Time.timeScale = 0;  Text numberOfGuess = GameObject.FindGameObjectWithTag("NumberOfGuess").GetComponent<Text>();  numberOfGuess.text = str\_lives.text;  saveData.SaveData();  }  }  public void PauseGameBackToHome()  {  dialog.SetActive(true);  }  public void BackToHome()  {  SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex-1);  }  private void Update()  {  StartCoroutine(checkGameOver(str\_lives.text));  }  public void Reset()  {  SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);  }  public void checkDialogResult()  {  DialogResult dialogResult = dialog.GetComponent<DialogResult>();  if (dialogResult.getOK() == true)  {  InsideDialog.SetActive(true);  GameObject number = GameObject.FindGameObjectWithTag("NumberOfRightAnswer");  Text num = number.GetComponent<Text>();  num.text = quizz.ToString();  GameObject btnOK = GameObject.Find("btnOK");  btnOK.SetActive(false);  GameObject btnCancel = GameObject.Find("btnCancel");  btnCancel.SetActive(false);  btnOK2.SetActive(true);  }  else  {  dialog.SetActive(false);  gamePauseDialog.SetActive(true);  }  }  } |

### 4.2.3 AttackOrc

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | AttackOrc | Xử lý hành động hiệp sĩ tiêu diệt quái vật khi chiến thắng |

*Các phương thức trong class AttackOrc*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | Void Start() | Khởi tạo parameter |
| 2 | Void Update() | Cập nhật lại khi có thay đổi |
| 3 | IEnumrator WinState() | Nhận mthod WinState trong class GameManager |
| 4 | void OnCollisionEnter2D(Collision2D collision) | Kiểm tra va chạm khi hiệp sĩ tiêu diệt quái vật  và xử lý sự kiện này |

*Source Code:*

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.UI;  public class AttackOrc : MonoBehaviour  {  public Text orcTalk;  public GameObject orcBubble;  public GameObject princeBubble;  private Rigidbody2D rb;  private float velocity = 0f;  private bool canmove = true;  private Animator animator;  private Animator orcAnimator;  public GameObject orc;  private bool orcDeath = false;  private GameManager gameManager;  public GameObject princessWin;  public GameObject princess;  private AudioSource orcSound;  public void Start()  {  rb = GetComponent<Rigidbody2D>();  animator = GetComponent<Animator>();  orcAnimator = orc.GetComponent<Animator>();  orcSound = orc.GetComponent<AudioSource>();  gameManager = GameObject.Find("GameManager").GetComponent<GameManager>();  orcDeath = false;  Debug.Log(orcDeath);  }  private void Update()  {  rb.velocity = new Vector2(velocity \* 1, rb.velocity.y);  if (orcTalk.text == "Nooooooo!" && canmove)  {  GameObject.Destroy(princeBubble);  velocity = 3f;  animator.SetBool("Run", true);  }  if (rb.transform.position.x >= 0f && !orcDeath)  {  canmove = false;  velocity = 0f;  animator.SetBool("Atk", true);  }  if (orcDeath != false)  {  StartCoroutine(gameManager.WinState(orcDeath));  }  }  IEnumerator WinState()  {  yield return new WaitForSeconds(3f);  gameManager.WinState(true);  }  private void OnCollisionEnter2D(Collision2D collision)  {  if (collision.gameObject.CompareTag("Orc"))  {  GameObject.Destroy(orcBubble);  GameObject.Destroy(princess);  orcSound.Play();  princessWin.SetActive(true);  orcDeath = true;  orcAnimator.SetTrigger("Dead");  animator.SetBool("Run", false);  animator.SetBool("Atk", false);  }  }  } |

### 4.2.4 KeyInputHandler

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | KeyInputHandler | Nhận sự kiện KeyInput |

*Các phương thức trong KeyInputHandler*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | Void Update() | Nhận sự kiện KeyInput |

### 4.2.5 Answer

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | Answer | Xử lý các sự kiện KeyDown |

*Các phương thức trong Answer*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | void OnEnable() | Đăng ký sự kiện KeyInputHandler |
| 2 | void OnDisable() | Xóa sự kiện KeyInputHandler |
| 3 | void HandleKeyDown(KeyCode key) | Xử lý sự kiện nếu có KeyDown có KeyCode tương ứng |
| 4 | IEnumerator change\_msg(string msg, float displaytime) | Thay đổi message trong bong bóng chat của quái vật về mặc định sau khoảng thời gian displaytime |
| 5 | void SetText(string msg) | Thay đổi message trong bong bóng chat của quái. |
| 6 | void OnNewMsg(string newmsg) | Kiểm tra xem có message mới không nếu có thì cập nhật lại. |

*Source Code*

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using System.Security.Cryptography;  using Unity.VisualScripting;  using UnityEngine;  using UnityEngine.UI;  public class Answer : MonoBehaviour  {  public Text answer;  public Text msg\_check;  public int number;  private bool isNewMessage;  private void OnEnable()  {  ReadInput.OnKeyDown += HandleKeyDown;  }  private void OnDisable()  {  ReadInput.OnKeyDown -= HandleKeyDown;  }  private void HandleKeyDown(KeyCode key)  {  if (Time.timeScale != 0)  {  if (key == KeyCode.Backspace && answer.text.Length > 0)  {  answer.text = answer.text.Substring(0, answer.text.Length - 1);  }  if (key >= KeyCode.Alpha0 && key <= KeyCode.Alpha9)  {  string keyValue = key.ToString().Replace("Alpha", "");  if (answer.text == null || answer.text == "")  {  answer.text = keyValue;  }  else  {  if (int.Parse(answer.text + keyValue) <= 100)  {  answer.text += keyValue;  }  }  }  if (key >= KeyCode.Keypad0 && key <= KeyCode.Keypad9)  {  string keyValue = key.ToString().Replace("Keypad", "");  if (answer.text == null || answer.text == "")  {  answer.text = keyValue;  }  else  {  if (int.Parse(answer.text + keyValue) <= 100)  {  answer.text += keyValue;  }  }  }  if (key == KeyCode.Return)  {  if (answer.text != "")  {  number = int.Parse(answer.text);  GameManager gameManager = GetComponent<GameManager>();  Debug.Log(number);  answer.text = "";  string orc\_talk = gameManager.CheckAnswer(number);  StartCoroutine(change\_msg(orc\_talk, 1f));  }  }  }  }  IEnumerator change\_msg(string msg, float displaytime)  {  if (msg != null)  {  OnNewMsg(msg);  }  if (msg == "Nooooooo!")  yield break;  yield return new WaitForSeconds(displaytime);  isNewMessage = false;  if (!isNewMessage)  {  msg\_check.text = "Choose a number from 1 to 100!";  }  }  public void SetText(string msg)  {  msg\_check.text = msg;  }  public void OnNewMsg(string newmsg)  {  SetText(newmsg);  isNewMessage = true;  }  } |

### 4.2.6 MovePrincess

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | MovePrincess | Di chuyển công chúa xuống sau mỗi 2 lượt đoán sai |

*Các phương thức trong MovePrincess*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | void Start() | Khởi tạo các Parameter |
| 2 | void Update() | Cập nhật lại khi có thay đổi |
| 3 | void moveDown() | Di chuyển công chúa và dựng lại khi đi được một khoảng nhất định |
| 4 | void setV\_move(float axisVertical) | Thay đổi trạng thái di chuyển và dừng |
| 5 | bool checkMove(int live) | Kiểm tra xem có thể di chuyển không |

*Source Code*

|  |
| --- |
| using System;  using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.UI;  public class MovePrincess : MonoBehaviour  {  private int velocity = 100;  public float totalDistance = 0f;  public float v\_move;  public string current\_lives;  public Rigidbody2D rb;  public GameObject burning;  public Text NumberOfLives;  // Update is called once per frame  private void Start()  {  totalDistance = 0f;  current\_lives = NumberOfLives.text;  v\_move = 0;  }  void Update()  {  if (current\_lives != NumberOfLives.text)  {  bool kt = checkMove(int.Parse(NumberOfLives.text));  if (kt == true)  {  setV\_move(-1);  current\_lives = NumberOfLives.text;  }  }  moveDown();  if (NumberOfLives.text == "0" && v\_move == 0)  {  burning.SetActive(true);  }  }  private void moveDown()  {  if (rb != null)  {  rb.velocity = new Vector2(rb.velocity.x, velocity \* v\_move);  }  totalDistance += Math.Abs(velocity \* v\_move);  if (totalDistance >= 28000f)  {  v\_move = 0;  totalDistance = 0f;  }  }  private void setV\_move(float axisVertical)  {  if (axisVertical!=0)  {  v\_move = axisVertical;  }  }    private bool checkMove(int live)  {  if (live<10 && live%2 == 0 && live!=0)  {  return true;  }  return false;  }  } |

### 4.2.7 SavingData

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | SavingData | Cập nhật số lượt chơi mà người chơi đã chơi |

*Các phương thức trong SavingData*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | void Start() | Lấy lượt chơi đã lưu trong PlayerPrefs |
| 2 | void SaveData() | Cập nhật số lượt chơi |
| 3 | void LoadData () | Hiển thị số lượt chơi |

*SourceCode*

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.SocialPlatforms.Impl;  using UnityEngine.UI;  public class SavingData : MonoBehaviour  {  public string key\_numberOfRound = "NumberOfRound";  public int numberOfRound = 0;  public Text roundPlayed;  // Start is called before the first frame update  void Start()  {  numberOfRound = PlayerPrefs.GetInt(key\_numberOfRound);  if (roundPlayed != null)  {  roundPlayed.text = "Round Played: " + numberOfRound.ToString();  }  }    public void SaveData()  {  numberOfRound++;  PlayerPrefs.SetInt(key\_numberOfRound, numberOfRound);  PlayerPrefs.Save();  Debug.Log("Save: " + numberOfRound);  }  public void LoadData()  {  int num = PlayerPrefs.GetInt(key\_numberOfRound);  Debug.Log("Load num: "+ num);  }  } |

### 4.2.8 DialogResult

|  |  |  |
| --- | --- | --- |
| STT | Tên Class | Mục đích |
| 1 | DialogResult | Kiểm tra xem người chơi click OK hay Cancel trong cửa sổ Continue Quit |

*Các phương thức trong SavingData*

|  |  |  |
| --- | --- | --- |
| STT | Tên phương thức | Mục đích |
| 1 | bool getOK() | Lấy giá trị để kiểm tra OK hay Cancel |
| 2 | void setOK(bool kt) | Thay đổi giá trị biến để xem click OK hay Cancle |

*Source Code*

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class DialogResult : MonoBehaviour  {  private bool OK;  public bool getOK()  {  return OK;  }  public void setOK(bool kt)  {  OK = kt;  }  } |

# CHƯƠNG V: CÀI ĐẶT VÀ KIỂM THỬ

**Các tình huống kiểm thử:**

|  |  |  |
| --- | --- | --- |
| STT | Tình huống | Mục đích |
| 1 | Dữ liệu đầu vào là các phím không phải là số khi người chơi trả lời câu hỏi  Kết quả dự kiến: Không bắt sự kiện | Kiểm tra trò chơi có bắt sự kiện đúng hay không |
| 2 | Nhập liên tục các câu trả lời nhanh  Kết quả dự kiến: Màn hình Update đúng  Nhận thấy: Còn hạn chế nếu nhập liên tục các câu trả lời thì màn hình update vị trí của công chúa không theo dự kiến | Kiểm tra xem màn hình có được update đúng không khi dữ liệu đầu vào liên tục |
| 3 | Đưa ra câu trả lời sai mà nhấn Pause sau đó tạo trò chơi mới.  Kết quả dự kiến: Màn hình Game Playing được trả lại trạng thái ban đầu và câu hỏi thay đổi. | Kiểm tra trò chơi có được tải lại và câu hỏi có được làm mới và số lượt chơi có được phục hồi |

**CHƯƠNG VI: KẾT LUẬN**

## 6.1 Kết luận

Trò chơi đã được thiết kế với một giao diện độc đáo và hấp dẫn, tạo ra một trải nghiệm giải đố thú vị. Từ màn hình Main Menu đến màn hình Game Playing, mỗi yếu tố được tích hợp nhằm tối ưu hóa sự tương tác của người chơi và tăng cường cảm giác hồi hộp của câu chuyện.

Màn hình Main Menu cung cấp các tùy chọn dễ sử dụng, giúp người chơi dễ dàng bắt đầu trò chơi mới, xem hướng dẫn hoặc thoát khỏi trò chơi. Trong khi đó, màn hình Game Playing không chỉ tập trung vào việc đoán số mà còn đưa vào câu chuyện với nhân vật hiệp sĩ, quái vật và biểu tượng trái tim, tạo nên một trải nghiệm đa chiều và gần gũi với người chơi.

Với sự kết hợp giữa yếu tố giải đố và yếu tố câu chuyện, trò chơi mang lại cảm giác hứng thú và thách thức. Tổng cộng, thiết kế của trò chơi không chỉ tập trung vào khía cạnh giải đố mà còn chú trọng đến việc tạo nên một trải nghiệm chơi game độc đáo và đầy tính nghệ thuật.

## 6.2 Ưu điểm và nhược điểm

### 6.2.1 Ưu điểm

**-** Giải trí và giáo dục: Trò chơi đoán số không chỉ mang lại tính giải trí, giải stress mà còn giúp người chơi phát triển khả năng tư duy, suy luận.

- Giao diện dễ nhìn: Giao diện của trò chơi thân thiện, được thiết kế một cách đơn giản với chủ đích giúp người chơi dễ dàng tiếp cận.

- Thu hút người chơi: trò chơi được kết hợp giữa yếu tố giải đố và câu chuyện nên tạo cảm giác thích thú hơn.

- Âm thanh sống động.

### 6.2.2 Nhược điểm

**-** Vì đây là trò chơi đoán số từ 1 đến 100, nên không đa dạng chế độ chơi, khiến người chơi chỉ chơi vài vòng.

- Trò chơi còn gặp một số lỗi khi liên tục nhập quá nhanh câu trả lời thì vị trí của công chúa không được như mong đợi

## 6.3 Hướng phát triển

- Tiếp tục cải tiến, nâng cấp, cập nhật giao diện để có thể thu hút người chơi hơn.

- Xây dựng thêm các chức năng mở rộng, tương tác xã hội chia sẻ kết quả qua các trên mạng xã hội hay gửi lời mời chơi cùng với bạn bè để có thể tăng tính cạnh tranh.

- Xây dựng thêm chức năng thưởng như bảng xếp hạng, huy chương hay điểm tích lũy để có thể giúp người chơi tăng thêm động lực và sự hứng thú mỗi khi chơi.

- Xây dựng thêm chức năng phản hồi và đánh giá để thể khắc phục và chỉnh sửa kịp thời những lỗi mà trò chơi xảy ra để có thể kịp thời sửa chữa

## 6.4 Các khó khăn gặp phải

**-** Bắt đầu với môi trường làm việc mới như unity còn khó khăn trong việc làm quen.

- Gặp xung đột giữa các sự kiện khi update màn hình.

**Biện pháp khắc phục**

**-** Tìm hiểu các cách sử dụng unity cơ bản.

- Xem các video hướng dẫn sử dụng unity trên YouTube.

- Hiển thị các thông số ra màn hình để tìm và vá lỗi.

# TÀI LIỆU THAM KHẢO

[1] C Sharp là gì? Đặc điểm & ứng dụng của ngôn ngữ lập trình C#

Link: <https://bkhost.vn/blog/c-sharp/>

[2] Lập trình Game Unity là gì? Một số điều mà bạn cần biết về lập trình game Unity

Link: <https://aptech.fpt.edu.vn/lap-trinh-game-unity.html>

[3]

Link: