Do an lap trinh

Lập trình hướng đối tượng (University of Information Technology)

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

.

ĐỒ ÁN MÔN HỌC

LẬP TRÌNH TRỰC QUAN

TRÒ CHƠI RẮN SĂN MỒI

Giảng viên hướng dẫn : Nguyễn Thị Xuân Hương

Sinh viên thực hiện 1 : Lê Quang Phúc

Mã sinh viên 1 : 22521118

Sinh viên thực hiện 2 : Hoàng Gia Phong

Mã sinh viên 2 : 22521084

Sinh viên thực hiện 3 : Nguyễn Tấn Phúc

Mã sinh viên 3 : 22521132

Lớp : IT008.N13.PMCL

Bộ môn : Phát triển phần mềm

TP. HỒ CHÍ MINH, THÁNG 1 NĂM 2023

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

NHIỆM VỤ ĐỒ ÁN MÔN HỌC

Họ và tên SV 1: Hoàng Gia Phong MSSV: 22521084

Họ và tên SV 2: Lê Quang Phúc MSSV: 22521118

Họ và tên SV 2: Nguyễn Tấn Phúc MSSV: 22521132

Lớp: IT008.O11

Tên đề tài: Trò chơi rắn săn mồi

Giảng viên giảng dạy: Nguyễn Thị Xuân Hương

Thời gian thực hiện: **từ 15/11 đến 31/12**

Nhiệm vụ đồ án môn học: (phụ thuộc vào từng chủ đề)

1. Xây dựng CSDL trong SQL Server.
2. Thiết kế giao diện phần mềm.
3. Lập trình xử lý phần mềm với các chức năng sau:
   - Đăng nhập, đăng kí
   - Chọn map
   - Chọn đồ ăn
   - Chọn rắn
   - Điều khiển rắn
4. Nộp file nén (*.rar) lưu sản phẩm đề tài bao gồm:
   - File báo cáo word (*.docx)
   - File thuyết trình (*.pptx)
   - Thư mục chứa dự án (project), các class thư viện, CSDL, hình ảnh, ...)

Tp.HCM, ngày … tháng … năm 2023

GIẢNG VIÊN GIẢNG DẠY

(Ký và ghi rõ họ tên)

BẢNG PHÂN CÔNG THỰC HIỆN ĐỒ ÁN MÔN HỌC

(Nếu đồ án chỉ có 1 SV thực hiện thì không làm trang này)

Họ tên SV1: **Hoàng Gia Phong** MSSV: **22521084**	Họ tên SV2: **Lê Quang Phúc** MSSV: **22521118**	Họ tên SV3: **Nguyễn Tấn Phúc** MSSV: **22521132**
Thiết kế giao diện Làm báo cáo word	Xử lí trò chơi Làm báo cáo word	Thiết kế cơ sở dữ liệu Làm powerpoint
SV thực hiện 1 *(Ký tên)*	**SV thực hiện 2** *(Ký tên)*	**SV thực hiện 3** *(Ký tên)*

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn chân thành đến Thầy/Cô vì sự hỗ trợ và dành thời gian quý báu trong suốt quá trình thực hiện đồ án game của chúng em.

Hướng Dẫn và Chỉ Bảo

Chúng em trân trọng sự hướng dẫn và chỉ bảo kỹ thuật, kiến thức chuyên sâu mà Cô đã chia sẻ. Những lời giải thích và hướng dẫn chi tiết đã giúp chúng em hiểu rõ hơn về quy trình phát triển game.

Phản Hồi Xây Dựng

Chúng em đánh giá cao những phản hồi chi tiết và xây dựng mà Cô đã cung cấp. Điều này giúp chúng em cải thiện kỹ năng lập trình và thiết kế game, từ đó nâng cao chất lượng của đồ án.

Hỗ Trợ Kỹ Thuật

Sự hỗ trợ kỹ thuật của Cô trong việc giải quyết các vấn đề và khó khăn kỹ thuật là một phần quan trọng, giúp chúng em vượt qua những thách thức trong quá trình phát triển.

Khuyến Khích và Động Viên

Lời động viên và khuyến khích từ Cô là nguồn động viên lớn giúp chúng em vượt qua những thời kỳ khó khăn và không chắc chắn.

Sự Nhiệt Huyết và Cam Kết

Sự nhiệt huyết và cam kết của Cô với việc giảng dạy đã tạo động lực mạnh mẽ, khích lệ chúng em chăm chỉ hơn trong quá trình nghiên cứu và phát triển.

Chúng em rất tự hào và biết ơn vì đã có cơ hội được học hỏi và làm việc dưới sự hướng dẫn của Cô. Đồ án game này không chỉ là kết quả của sự cố gắng của chúng em mà còn là kết quả của sự hỗ trợ chân thành từ Cô. Chúng em rất biết ơn điều đó và hân hoan gửi lời cảm ơn sâu sắc.

Chân thành cảm ơn, cô Nguyễn Thị Xuân Hương

Nhóm sinh viên thực hiện

Nhóm 3

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
…………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
…………
………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………

Tp.HCM, ngày … tháng … năm …

GVHD

MỤC LỤC

DANH MỤC CÁC HÌNH

DANH MỤC CÁC BẢNG

Chương 1 Giới thiệu đề tài

1.1. Tên đề tài

- Trò chơi rắn săn mồi

1.2. Mô tả đề tài

- Trò chơi rắn săn mồi là một trò chơi cổ điển, trong đó người chơi điều khiển một con rắn di chuyển xung quanh màn hình để ăn các vật phẩm.
- Khi ăn được một vật phẩm, con rắn sẽ tăng kích thước và tốc độ di chuyển. Người chơi sẽ thua cuộc nếu con rắn chạm vào chướng ngại vật, bẫy hoặc tự cắn vào chính bản thân.
- Các nhiệm vụ mà đề tài thực hiện
  - Tạo ra một trò chơi rắn săn mồi với các chức năng chính sau :
    - Điều khiển con rắn di chuyển xung quanh màn hình
    - Ăn các vật phẩm để tăng kích thước và tốc độ con rắn
    - Tránh các chướng ngại vật và bẫy
    - Thuật toán để con rắn tự di chuyển
  - Tạo ra các map khác nhau với mức độ khó tăng dần
- Các yêu cầu cần đạt được :
  - Trò chơi phải có giao diện trực **quan, dễ nhìn và dễ sử dụng**
  - Trò chơi phải có lối chơi hấp dẫn, lôi cuốn người chơi
  - Trò chơi phải có độ khó tăng dần để người chơi có thể thử thách bản thân

1.3. Lý do chọn đề tài

- Trò chơi rắn săn mồi là một trò chơi cổ điển, đã có mặt từ rất lâu và vẫn được nhiều người yêu thích

- Trò chơi có lối chơi đơn giản nhưng hấp dẫn, phù hợp với nhiều đối tượng người chơi
- Trò chơi có thể được phát triển trên nhiều nền tảng khác nhau, bao gồm máy tính, điện thoại, máy tính bảng

1.4. Các chức năng chính của đề tài

- Điều khiển con rắn di chuyển: Người chơi có thể điều khiển con rắn di chuyển bằng các phím mũi tên trên bàn phím hoặc các nút điều khiển trên màn hình cảm ứng.
- Ăn các vật phẩm: Khi con rắn di chuyển vào vị trí của một vật phẩm, vật phẩm đó sẽ biến mất và con rắn sẽ tăng kích thước.
- Tránh các chướng ngại vật và bẫy: Trên màn hình sẽ xuất hiện các chướng ngại vật và bẫy. Nếu con rắn chạm vào các chướng ngại vật hoặc bẫy, người chơi sẽ thua cuộc.
- Thuật toán để con rắn tự di chuyển: Con rắn sẽ tự di chuyển theo một thuật toán đã được định sẵn. Người chơi có thể thay đổi thuật toán này để tạo ra các lối chơi khác nhau.

1.5. Công nghệ sử dụng

- Công nghệ đồ họa: Trò chơi sẽ sử dụng công nghệ đồ họa 2D để tạo ra hình ảnh và hiệu ứng trong game.
- Ngôn ngữ C# lập trình WindowForm
- SQL Server

1.6. Môi trường lập trình

- Microsoft Visual Studio 2022

1.7. Công cụ hỗ trợ (nếu có)

- **Draw.io**: Thiết kế sơ đồ thuật toán
- **GitHub** : Quản lý project

Chương 2 GIỚI THIỆU CÔNG NGHỆ

2.1. Window Form

2.1.1. Giới thiệu

- Windows Forms (WinForms) là GUI mã nguồn mở và miễn phí được bao gồm như một phần của Microsoft.NET Framework hoặc Mono Framework, cung cấp nền tảng để viết các ứng dụng khách phong phú cho máy tính. Mặc dù nó được coi là sự thay thế cho Thư viện lớp nền tảng Microsoft Foundation của C ++ trước đây và phức tạp hơn, nhưng nó không cung cấp mô hình tương đương và chỉ hoạt động như một nền tảng cho tầng giao diện người dùng trong một giải pháp nhiều tầng.

2.1.2. Lịch sử

- Windows Forms phát hành lần đầu vào 13 - 2 – 2002. Giống như Tóm tắt Window Toolkit (AWT), API Java tương đương, các biểu mẫu Windows là một cách sớm và dễ dàng để cung cấp các thành phần giao diện người dùng đồ họa cho .NET Framework. Các biểu mẫu Windows được xây dựng trên API Windows hiện có và một số điều khiển chỉ quấn các thành phần Windows bên dưới. Một số phương thức cho phép truy cập trực tiếp vào các cuộc gọi lại Win32, không khả dụng trong các nền tảng không phải Windows.

2.1.3. Chức năng

- Cung cấp các control hỗ trợ tính năng kéo thả để thiết kế giao diện người dùng, nhờ vậy, việc xây dựng, phát triển các ứng dụng trở nên dễ dàng.

2.1.4. Áp dụng

- Thiết kế cũng như xây dựng các form trong ứng dụng ví dụ như: FormChinhDe, FormChinhVua, FormChinhKho,… và hỗ trợ tạo các class như Ran, VatCan,…

2.2. SQL server

2.2.1. Giới thiệu

- Microsoft SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ được phát triển bởi Microsoft. Là một máy chủ cơ sở dữ liệu, nó là một sản phẩm phần mềm có chức năng chính là lưu trữ và truy xuất dữ liệu theo yêu cầu của các ứng dụng phần mềm 4 khác. Có thể chạy trên cùng một máy tính hoặc trên một máy tính khác trên mạng (bao gồm cả Internet).

2.2.2. Lịch sử

- Microsoft SQL Server bắt đầu với sản phẩm Microsoft SQL Server đầu tiên SQL SQL Server 10, máy chủ 16-bit cho hệ điều hành OS/2 vào năm 1989 và kéo dài đến ngày hiện tại.

2.2.3. Chức năng

- Lưu trữ và truy xuất dữ liệu theo yêu cầu của các ứng dụng phần mềm khác.

2.2.4. Áp dụng

- Tạo cơ sở dữ liệu TaiKhoan dùng để lưu trữ phần lớn các thông tin của ứng dụng ví nhụ như: Tên tài khoản, mật khẩu, gmail, level của tài khoản và điểm cao nhất ở mức dễ, vừa, khó của tài khoản đó.

Chương 3 THIẾT KẾ HỆ THỐNG

3.1. Mô tả bài toán

- Snake Game là một trò chơi điện tử cổ điển được phát hành lần đầu tiên vào năm 1976. Trò chơi này yêu cầu người chơi điều khiển một con rắn ăn các hạt thức ăn xuất hiện trên màn hình. Khi con rắn ăn các hạt thức ăn, nó sẽ phát triển chiều dài. Nếu con rắn chạm vào chính nó hoặc các cạnh của màn hình, trò chơi sẽ kết thúc.
- Snake Game vẫn là một trong những trò chơi điện tử phổ biến nhất mọi thời đại. Nó có sẵn trên nhiều nền tảng khác nhau, bao gồm điện thoại thông minh, máy tính bảng, máy tính và các thiết bị chơi game.

Độ phổ biến

- Theo một cuộc khảo sát do Statista thực hiện vào năm 2022, Snake Game là trò chơi điện tử được chơi nhiều nhất trên thế giới. Cuộc khảo sát cho thấy rằng 72% người chơi điện tử đã chơi Snake Game ít nhất một lần.
- Trò chơi này phổ biến ở mọi lứa tuổi và giới tính. Nó đặc biệt phổ biến ở trẻ em và thanh thiếu niên.

Các phiên bản hiện đại

- Snake Game đã được làm mới và cải tiến nhiều lần trong những năm qua. Các phiên bản hiện đại của trò chơi thường có đồ họa đẹp hơn, các chế độ chơi mới và các tính năng bổ sung.

3.2. Xác định các chức năng của hệ thống

- Các chức năng của hệ thống trong trò chơi Snake bao gồm:
  - Đăng nhập, đăng kí: Người chơi sử dụng tài khoản để có thể tham gia trò chơi.
  - Chọn map, chọn rắn và chọn thức ăn
  - Khởi tạo trò chơi: Tạo màn hình trò chơi, khởi tạo các đối tượng cần thiết, chẳng hạn như con rắn, thức ăn, v.v.

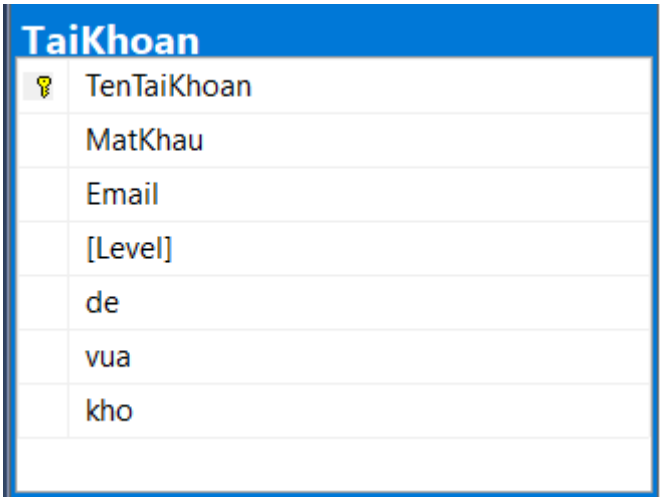
- Điều khiển con rắn: Nhận đầu vào từ người chơi và điều khiển con rắn di chuyển.
- Tạo thức ăn: Tạo thức ăn mới trên màn hình khi con rắn ăn hết thức ăn hiện có.
- Kiểm tra va chạm: Kiểm tra xem con rắn có va chạm với chính nó hoặc các chướng ngại vật hay không.
- Kết thúc trò chơi: Kết thúc trò chơi khi con rắn va chạm với chính nó hoặc các chướng ngại vật, lưu lại điểm cao nhất ứng với mỗi map

3.3. Thiết kế cơ sở dữ liệu

```
create table TaiKhoan
(
    TenTaiKhoan varchar(50) not null primary key,
    MatKhau varchar(50),
    Email varchar(50),
    Level varchar(50),
    de varchar(50),
    vua varchar(50),
    kho varchar(50),
)
```

- Ta sẽ tạo bảng TaiKhoan với các thuộc tính:
    - TenTaiKhoan: Tài khoản người dùng.
    - MatKhau: Mật khẩu người dùng.
    - Email: Email người dùng.
    - Level: Lưu trữ level account.
    - de: Lưu trữ điểm cao nhất của người chơi tại map dễ.
    - vua: Lưu trữ điểm cao nhất của người chơi tại map vừa.
    - kho: Lưu trữ điểm cao nhất của người chơi tại map khó.

3.4. Database diagram trong SQL



Hình 3.4.1. Datatabase diagram trong AQL

3.5. Cấu trúc các bảng dữ liệu trong SQL

Bảng 3.5.1 Cấu trúc bảng TaiKhoan

Field Name	Field Type	Descriptions
TenTaiKhoan	varchar(50)	Tên tài khoản
MatKhau	varchar(50)	Mật khẩu
Email	varchar(50)	Email
Level	varchar(50)	Level của tài khoản
de	varchar(50)	Mức điểm cao nhất ở mức dễ
vua	varchar(50)	Mức điểm cao nhất ở mức vừa
kho	varchar(50)	Mức điểm cao nhất ở mức khó

3.6. Dữ liệu mẫu

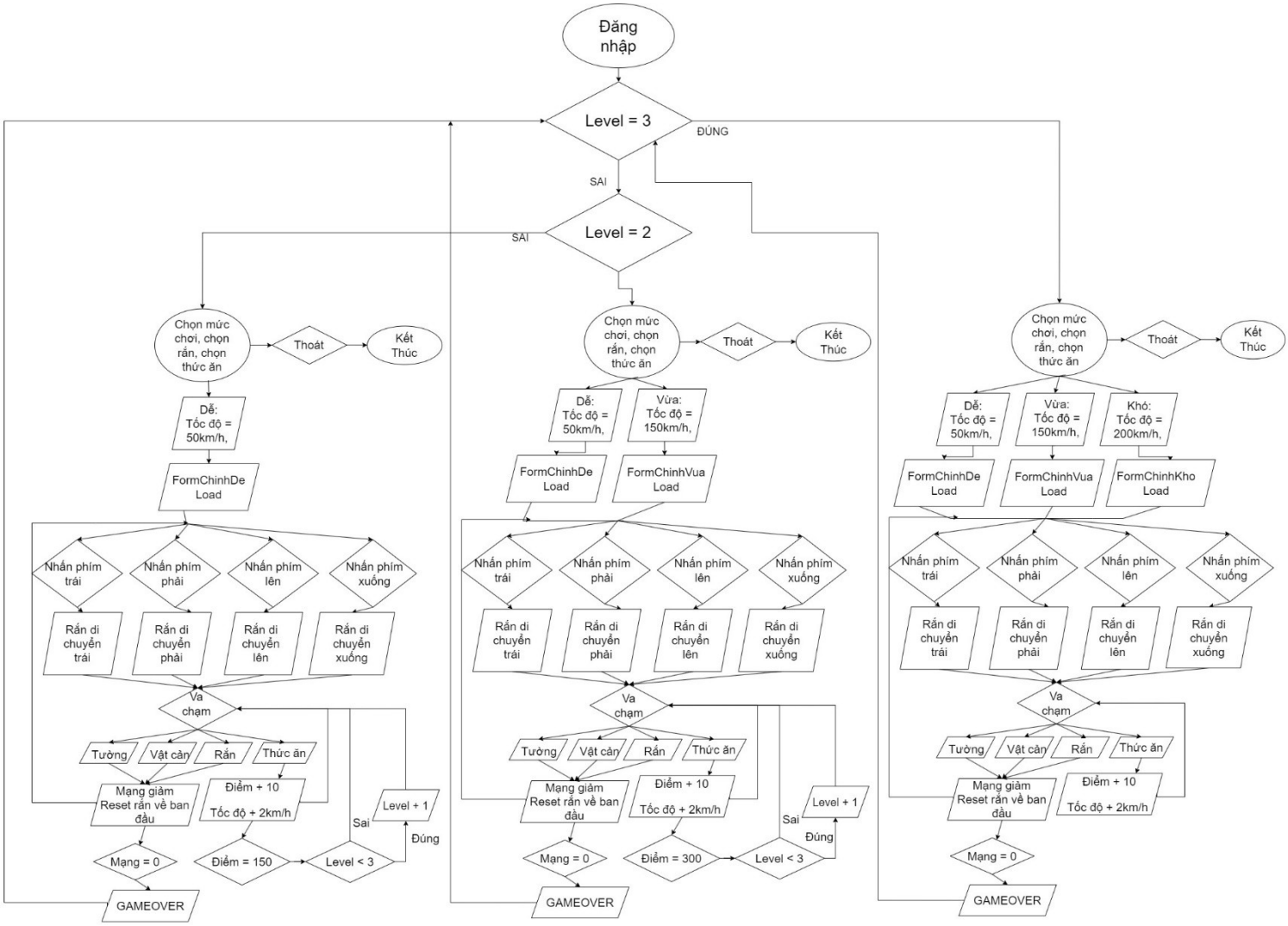
Bảng 3.6.2 Dữ liệu mẫu của bảng TaiKhoan

TenTaiKhoan	MatKhau	Email	Level	de	vua	kho
aaa123	123456	phonglon@gmail.com	3	200	1000	200
abc1234	abc123	abc1234@gmail.com	3	150	2000	300
abc12345	abc123	abc12345@gmail.com	2	300	2000	400
huong123	123456789	huong123@gmail.com	2	800	4000	100

3.7. Thuật toán

- Bước 1: Khởi tạo con rắn gồm các đốt và thiết kế đầu rắn để có thể phân biệt được giữa đầu rắn và đuôi rắn.
- Bước 2: Lập trình để rắn có thể di chuyển theo hướng các phím mũi trên trên bàn phím máy tính và lập trình hàm rắn lớn.
- Bước 3: Khởi tạo mồi để rắn có thể ăn được.
- Bước 4: Lập trình để khi rắn ăn mồi sẽ dài thêm một đốt đồng thời mồi của rắn ngẫu nhiên xuất hiện ở vị trí khác.
- Bước 5: Khởi tạo vật cản.
- Bước 6: Khởi tạo bức tường .
- Bước 7: Lập trình hàm hàm rắn chết.
- Bươc 8: Lập trình để khi rắn cắn vào thân sẽ chết.
- Bước 9: Lập trình để khi rắn chạm vào bức tường sẽ chết.
- Bước 10: Xây dựng hệ thống tính điểm trong game, lưu điểm cao nhất
- Bước 11: Xây dựng chức năng tăng tốc trong game.
- Bước 12: Xây dựng tính mạng cho rắn.
- Bước 13: Kết thúc.

3.8. Sơ đồ phân tích trò chơi



Hình 3.8.2 Sơ đồ phân tích trò chơi

Chương 4 XÂY DỰNG ỨNG DỤNG

4.1. Thiết kế và tạo các class

4.1.1. Thiết kế lớp Setting

- Đoạn mã này định nghĩa một lớp có tên Setting trong namespace C# là Snake. Lớp này được thiết kế theo mô hình singleton, có nghĩa là chỉ có một thể hiện của lớp Setting có thể tồn tại trong toàn bộ ứng dụng.

- Dưới đây là phân tích về chức năng của lớp này:

Mô hình Thiết kế Singleton

- Lớp này tuân theo mô hình singleton sử dụng một thể hiện tĩnh riêng của nó (instance). Thuộc tính Instance làm cho chỉ có một thể hiện của lớp Setting được tạo.

```
private static Setting instance;

7 references
public static Setting Instance
{
    get
    {
        if (instance == null)
        {
            instance = new Setting();
        }
        return instance;
    }
}
```

Hình 4.1.1.3. Code lớp Setting với Singleton

Biến thành viên

- Lớp này có một số biến thành viên công khai, được sử dụng để lưu trữ các cài đặt liên quan đến trò chơi Snake. Điều này bao gồm các cài đặt cho diện mạo của con rắn (canvasHead và canvasBody), loại rắn (typeOfSnake), loại bản đồ (mapType), và hình ảnh / âm thanh được sử dụng trong trò chơi.

```
public Image food;
public  int mapType=1;

// luu tai khoan, mat khau, level
public string curLV;
public  string curTK;
public  string curMK;
public  string curde;
public  string curvua;
public  string curkho;

//
public  SoundPlayer soundMenu ;
public  SoundPlayer eatFood ;
public  SoundPlayer intersecSound ;
```

Hình 4.1.1.4 Code các biến thành viên hàm Setting

Constructor

- Constructor khởi tạo giá trị mặc định cho một số cài đặt, như màu của con rắn (canvasHead và canvasBody), các bộ phát âm thanh cho các sự kiện trò chơi (soundMenu, eatFood, intersecSound), và cấp độ ban đầu (curLV). Nó cũng tải một hình ảnh cho thức ăn từ một tệp và đặt nó vào thuộc tính food.

```
1 reference
public Setting()
{
    canvasHead = new SolidBrush(Color.Red);
    canvasBody = new SolidBrush(Color.Blue);
    soundMenu = new SoundPlayer("Form1music.wav");
    eatFood = new SoundPlayer("eatFoodSound.wav");
    intersecSound = new SoundPlayer("intersecSound.wav");
    curLV = "1";
    food = Image.FromFile("apple.png");
}
```

Hình 4.1.1.5. Code constructor lớp Setting

- Tóm lại, lớp này đóng gói các cài đặt và tài nguyên khác nhau cho một trò

chơi Snake, và được thiết kế dưới dạng singleton để đảm bảo một thể hiện toàn cục duy nhất trong toàn bộ ứng dụng. Các cài đặt bao gồm các khía cạnh hình ảnh của trò chơi, hiệu ứng âm thanh và thông tin tài khoản người dùng.

4.1.2. Thiết kế lớp Connection

- Mục đích của lớp này là cung cấp một phương thức để lấy đối tượng SqlConnection, mà thường được sử dụng trong ứng dụng .NET để thiết lập kết nối đến cơ sở dữ liệu Microsoft SQL Server.

```
namespace Snake
{
    2 references
    class Connection
    {
        private static string stringConnection = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename="
        2 references
        public static SqlConnection GetSqlConnection()
        {
            return new SqlConnection(stringConnection);
        }
    }
}
```

Hình 4.1.2.6 Code lớp Connection

4.1.3. Thiết kế lớp Modify

- Ta thiết kế lớp Modify để có thể truy vấn và thực hiện câu lệnh SQL trong database

```
6 references
public Modify()
{
}
SqlCommand sqlCommand;//dung de truy van cac cau lenh
SqlDataReader dataReader;//dung de doc du lieu trong bang
14 references
public List<TaiKhoan> TaiKhoans (string query)
{
    List<TaiKhoan> taiKhoans = new List<TaiKhoan>();
    using (SqlConnection sqlConnection = Connection.GetSqlConnection())
    {
        sqlConnection.Open();
        sqlCommand = new SqlCommand(query, sqlConnection);
        dataReader = sqlCommand.ExecuteReader();
        while(dataReader.Read())
        {
            taiKhoans.Add(new TaiKhoan(dataReader.GetString(0),dataReader.GetString(1),dataReader.GetString(
        }
        sqlConnection.Close();
    }
        return taiKhoans;
}
```

Hình 4.1.3.7. Code để truy vấn lớp Modify

```
6 references
public void Command(string query)
{
    using (SqlConnection sqlConnection =Connection.GetSqlConnection())
    {
        sqlConnection.Open();
        sqlCommand=new SqlCommand(query, sqlConnection);
        sqlCommand.ExecuteNonQuery();
        sqlConnection.Close();
    }
}
```

Hình 4.1.3.8. Code để thực hiện các câu lệnh SQL

4.1.4. Thiết kế lớp Tài Khoản

- Ta thiết kế lớp Tài Khoản để có thể lưu trữ các thông tin tài khoản trong database

```
6 references
class TaiKhoan
{
    private string tenTaiKhoan;
    private string matKhau;
    private string email;
    private string level;
    private string de;
    private string vua;
    private string kho;
```

Hình 4.1.4.9 Code thiết kế lớp TaiKhoan

4.1.5. Thiết kế lớp Rắn

- Để thiết kế nên con Rắn ta đã khởi tạo lớp mới có tên là **Ran.cs** với mảng một chiều có tên **MangRan[ ]** gồm các ô vuông độ lớn của mỗi ô vuông là 9 đơn vị được sắp xếp liền kề nhau. Tọa độ khởi tạo con rắn ở vị trí x = 30 và y = 10 và có 3 phần tử được khởi tạo.

```
20 references
class Ran
{
    //Khoi tao mang ran
    private Rectangle[] MangRan;
    private SolidBrush CoVeThanRan = new SolidBrush(Color.Black);
    private SolidBrush CoVeDauRan = new SolidBrush(Color.Yellow);
    private int x, y, width, height;
    42 references
    public Rectangle[] TraVeMangRan...
    1 reference
    public SolidBrush getCoVeThanRan()...
    1 reference
    public SolidBrush getCoVeDauRan()...
    1 reference
    public void setCoVeThanRan(SolidBrush tmp)...
    1 reference
    public void setCoVeDauRan(SolidBrush tmp)...
```

Hình 4.1.5.10 Code thiết kế lớp Rắn (1)

```
public Ran(SolidBrush head, SolidBrush body)
{
    MangRan = new Rectangle[3];

    // truyen toa do con ran

    this.x = 30;
    this.y = 10;
    CoVeDauRan = head;
    CoVeThanRan = body;
    width = 9;
    height = 9;

    // sap xep dot cua con ran
    for (int i = 0; i < MangRan.Length; i++)
    {
        MangRan[i] = new Rectangle(x, y, width, height);
        x -= width + 1;
    }
}
```

Hình 4.1.5.11 Code thiết kế lớp Rắn (2)

```
//Ve va to mau cho ran
4 references
public void VeRan(Graphics g)
{
    g.FillRectangle(CoVeDauRan, MangRan[0]);
    for (int i = 1; i < MangRan.Length; i++)
    {
        g.FillRectangle(CoVeThanRan, MangRan[i]);
    }
}
```

Hình 4.1.5.12. Code thiết kế lớp Rắn (3).

Lập trình di chuyển trên rắn

- Để toàn bộ rắn có thể di chuyển ta cần phải duyệt từng đốt từ đuôi rắn đến đầu rắn trong mảng Rắn và để các đốt phía sau đầu di chuyển tiếp theo đầu rắn.

```
4 references
public void VeRanChay()
{
    for (int i = MangRan.Length - 1; i > 0; i--)
    {
        MangRan[i] = MangRan[i - 1];
    }
}
```

Hình 4.1.5.13. Code làm rắn di chuyển

- Để rắn di chuyển xuống phía dưới ta gọi hàm rắn chạy để duyệt từng phần tử trong mảng rắn theo đầu rắn và sau đó cho tọa độ Y của đầu rắn liên tiếp tăng theo độ dài của 1 đơn vị của rắn sẽ là width +1 đơn vị.
- Để rắn di chuyển lên phía trên ta gọi hàm rắn chạy để duyệt từng phần tử trong mảng rắn theo đầu rắn và sau đó cho tọa độ Y của đầu rắn liên tiếp giảm width +1 đơn vị.
- Để rắn di chuyển sang trái ta gọi hàm rắn chạy để duyệt từng phần tử trong mảng rắn theo đầu rắn và sau đó cho tọa độ X của đầu rắn liên tiếp giảm width +1 đơn vị.
- Để rắn di chuyển sang phải ta gọi hàm rắn chạy để duyệt từng phần tử trong mảng rắn theo đầu rắn và sau đó cho tọa độ X của đầu rắn liên tiếp tăng width +1 đơn vị.

```
3 references
public void dichuyenxuong()
{
    VeRanChay();
    MangRan[0].Y += width + 1;

}
3 references
public void dichuyenlen()
{
    VeRanChay();
    MangRan[0].Y -= width + 1;
}
3 references
public void dichuyentrai()
{

    VeRanChay();
    MangRan[0].X -= width + 1;
}
3 references
public void dichuyenphai()
{
    VeRanChay();
    MangRan[0].X += width + 1;
}
```

Hình 4.1.5.14. Code di chuyển đầu rắn.

Điều khiển rắn

- Khi bấm mũi tên xuống ta gọi hàm dichuyenxuong() và đồng thời vô hiệu hóa nút mũi tên lên.

```
if (e.KeyData == Keys.Down && len == false)
{
    len = false;
    xuong = true;
    trai = false;
    phai = false;
}
```

```
if (xuong == true)
{
    ran.dichuyenxuong();
}
```

Hình 4.1.5.15. Code di chuyển đầu rắn đi xuống khi nhấn nút mũi tên xuống.

- Khi bấm mũi tên lên ta gọi hàm dichuyenlen() và đồng thời vô hiệu hóa nút mũi tên tiến xuống.

```
if (e.KeyData == Keys.Up && xuong == false)
{
    len = true;
    xuong = false;
    trai = false;
    phai = false;
}
```

```
if (len == true)
{
    ran.dichuyenlen();
}
```

Hình 4.1.5.16. Code di chuyển đầu rắn đi lên khi nhấn phím mũi tên lên.

- Khi bấm mũi tên phải ta gọi hàm dichuyenphai () và đồng thời vô hiệu hóa nút mũi tên tiến trái.

```
if (e.KeyData == Keys.Right & trai == false)
{
    len = false;
    xuong = false;
    trai = false;
    phai = true;
}
```

```
if (phai == true)
{
    ran.dichuyenphai();
}
```

Hình 4.1.5.17. Code di chuyển đầu rắn sang phải khi nhấn phím mũi tên phải

- Khi bấm mũi tên trái ta gọi hàm dichuyentrai() và đồng thời vô hiệu hóa nút mũi tên tiến phải.

```
if (e.KeyData == Keys.Left && phai == false)
{
    len = false;
    xuong = false;
    trai = true;
    phai = false;
}
```

```
if (trai == true)
{
    ran.dichuyentrai();
}
```

Hình 4.1.5.18 Code di chuyển đầu rắn sang phải khi nhấn phím mũi tên trái.

Lập trình để rắn lớn hơn

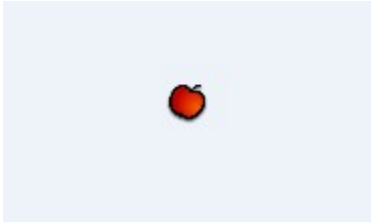
- Để rắn có thể lớn hơn thì ta chỉ cần thêm một đốt rắn vào mảng rắn ở cuối mảng rắn hiện thời.

```
//Ran lon len khi an thuc an
3 references
public void RanLon()
{
    List<Rectangle> rec = MangRan.ToList();
    rec.Add(new Rectangle(MangRan[MangRan.Length - 1].X, MangRan[MangRan.Length - 1].Y, width, height));
    MangRan = rec.ToArray();
}
```

Hình 4.1.5.19 Code làm rắn lớn hơn khi ăn thức ăn.

4.1.6. Khởi tạo mồi rắn

- Để khởi tạo mồi rắn em tạo thêm lớp mới có tên là ThucAn.cs với độ lớn là dài và rộng là 15 đơn vị và vị trí xuất hiện là ngẫu nhiên trong khoảng 710x441 trên form và mỗi khi rắn va chạm vào thức ăn thì sẽ xuất hiện ngẫu nhiên một thức ăn khác trong khoảng 710x441 từ vị trí hiện thời.
- Ở đây ta dùng lớp Image và khai báo đối tượng Image, dưới đây ví dụ là 1 trái táo đỏ từ file “apple.png”, dùng đối tượng graphics VeFood và hàm DrawImage để vẽ Image theo hình vuông ThucAnRec.



Hình 4.1.6.20 Hình trái táo

```
private int x, y, width, height;

public Rectangle ThucAnRec;

//Ham khoi tao thuc an
3 references
public ThucAn(Random randThucAn)
{
    x = randThucAn.Next(10, 710);
    y = randThucAn.Next(10, 441);
    width = 15;
    height = 15;
    ThucAnRec = new Rectangle(x, y, width, height);
}

18 references
public void RandomVitriThucAn(Random randThucAn)
{
    x = randThucAn.Next(10, 710);
    y = randThucAn.Next(10, 441);
}
4 references
public void VeThucAn(Graphics VeFood, Image foodImg)
{
    ThucAnRec.X = x;
    ThucAnRec.Y = y;
    Image curimage = foodImg;
    VeFood.DrawImage(curimage, ThucAnRec);
}
```

Hình 4.1.6.21. Code lớp ThucAn (ThucAn.cs)

- Ta có trường hợp nếu thức ăn ngẫu nhiên xuất hiện ở vị trí trùng với thân rắn thì phải dời ngẫu nhiên thức ăn sang nơi khác bằng cách họi lại hàm RandomViTriThucAn(randThucAn)

```
for (int i = 1; i < ran.TraVeMangRan.Length; i++)
{
    if (ran.TraVeMangRan[i].IntersectsWith(thucan.ThucAnRec))
    {
        thucan.RandomVitriThucAn(randthucan);
    }
}
```

Hình 4.1.6.22. Code dùng hàm RandomVitriThucAn

```
//Neu đầu rắn va chạm với thức ăn
if (ran.TraVeMangRan[0].IntersectsWith(thucan.ThucAnRec))
{
    setting.eatFood.Play();
    if (timer3.Interval > 2)
    {
        timer3.Interval -= 2;
    }
    diem += 10;
    if (diem >= int.Parse(setting.curvua))
    {
        highestScore.Text = diem.ToString();
    }
    ran.RanLon();
    thucan.RandomVitriThucAn(randthucan);
}
```

Hình 4.1.6.23. Xử lí thức ăn trong form chính khi trùng với thân rắn

4.1.7. Lập trình rắn dài thêm một đốt khi ăn thức ăn và xuất hiện ngẫu nhiên thức ăn ở vị trí khác

- Rắn ăn thức ăn nghĩa là khi tọa độ của đầu rắn ở trùng vị trí của thức ăn và trong C# có hàm phát hiện va chạm **IntersectsWith().**
- Khi rắn có va chạm vào thức ăn thì ta gọi hàm **ranlon()** để rắn lớn và gọi thêm hàm **RandomViTriThucAn()** để có thể xuất hiện thức ăn ở vị trí khác.

```
0 references
class Tuong
{
    private int x, y,z, width, height;
    public Rectangle tuong1;
    3 references
    public void VeTuong(Graphics veTuong)
    {
        z = 10;
        x = 5;
        y = 5;
        width = 723;
        height = 454;
        Pen p = new Pen(Color.Black, z);
        tuong1 = new Rectangle(x, y, width, height);
        veTuong.DrawRectangle(p, tuong1);
    }
}
```

Hình 4.1.7.24 Code độ dài rắn tăng thêm một đốt và xuất hiện ngẫu nhiên thức ăn ở vị trí khác khi rắn ăn thức ăn.

4.1.8. Viết hàm rắn chết

- Ta lập trình hiển thị thông báo rắn chết và gọi lại form mới và chơi lại với hàm Restart(), nếu mạng lớn hơn 0 sẽ chơi tiếp, nếu mạng bằng không sẽ thua và xuất hiện form GAMEOVER

Lập trình để khi rắn va chạm vào tường và chết

- Với tọa độ của bức tương là bao quanh 723x454 form và có viền bức tường là 10 nên ở phía trong có tọa độ là 713x444 nên ta sẽ lập trình rắn di chuyển ổn định ở vị trí 713x444 nếu rắn di chuyển ngoài tọa độ 713x444 hoặc rắn di chuyển ở tọa độ sau 10x10 thì sẽ gọi hàm Restart().

```
}
//Neu cham vao tuong se bi thua
if (ran.TraVeMangRan[0].X < 10 || ran.TraVeMangRan[0].X > 713)
{
    setting.intersecSound.Play();
    Restart();
}
if (ran.TraVeMangRan[0].Y < 10 || ran.TraVeMangRan[0].Y > 444)
{
    setting.intersecSound.Play();
    Restart();
}
```

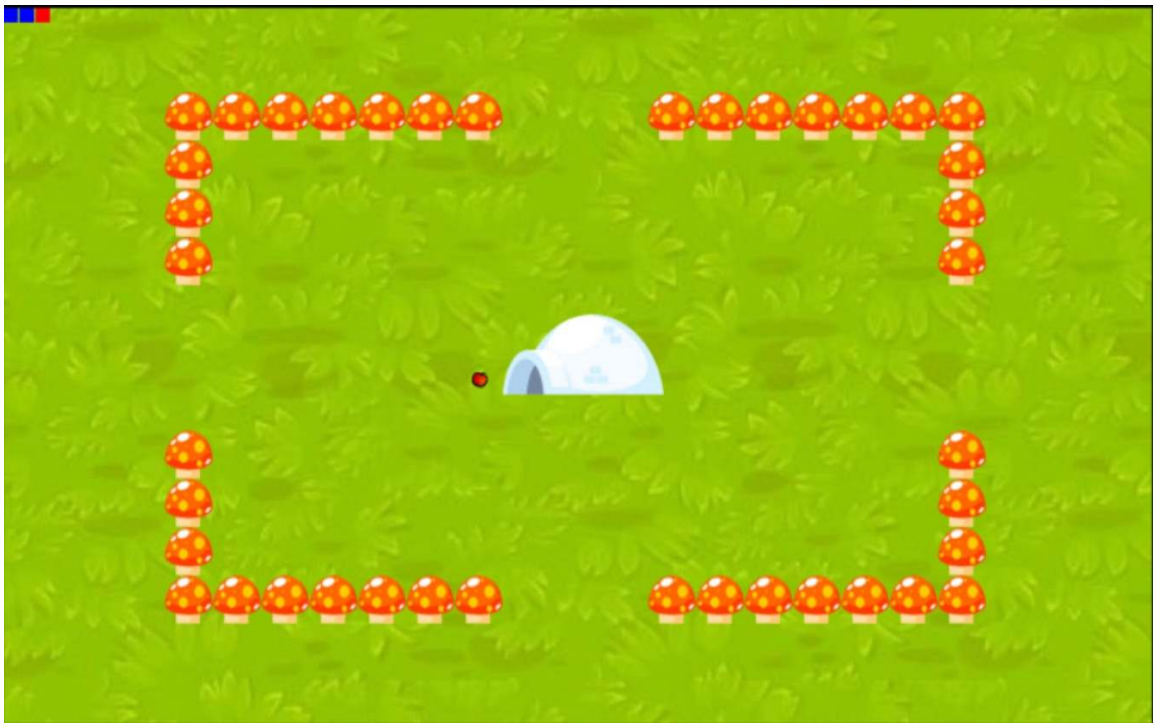
Hình 4.1.8.25 Code rắn va chạm với tường.

4.1.9. Khởi tạo lớp vật cản

Khởi tạo lớp vật cản

- Ở form FormChinhDe ta sẽ không tạo vật cản nhằm tạo sự dễ dàng cho người chơi.
- Ở form FormChinhVua ta sẽ dùng lớp Image và 2 đối tượng Image là nấm và vòm tuyết kết hợp với đối tượng Graphics “paper” để vẽ mảng 40 hình nấm và 1 vòm tuyết theo hình vuông với cú pháp ví dụ như sau:

```
Graphics Paper ;
Vatcanrec = new rectangle();
Image vatcan = Image.FromFIile(“FilePath”);
Paper.DrawImage = new(vatcan, Vatcanrec);
```



Hình 4.1.9.26. Giao diện vật cản vừa (VatCanVua.cs)

- Ở form FormChinhKho ta cũng làm tương tự nhưng với số lượng nhiều vật cản hơn sắp xếp lộn xộn.

Hình 4.1.9.27 Giao diện vật cản khó (VatCanKho.cs)

Lập trình trên vật cản

- Có 2 vấn để cần đặt ra khi ở lớp vật cản:
  - Vấn đề thứ nhất: Khi rắn va chạm vào vật cản thì rắn sẽ chết và gọi hàm **Restart().**
  - Vấn đề thứ 2: Khi thức ăn xuất hiện ngẫu nhiên trên vật cản thì cần phải gọi lại hàm xuất hiện ngẫu nhiên thức ăn khác **randomThucan()** tránh trường hợp nếu xuất hiện thức ăn trong vật cản thì rắn sẽ không thể ăn được.
- Lập Trình Giải Quyết Vấn Đề
  - Vấn đề thứ Nhất:

```
//Neu cham vao vat can se bi thua
for (int i = 0; i < vatcan.VatcanRec.Length; i++)
{
    if (ran.TraVeMangRan[0].IntersectsWith(vatcan.VatcanRec[i]))
    {
        setting.intersecSound.Play();
        Restart();
    }
}
for (int i = 0; i < vatcan.VatcanRec2.Length; i++)
{
    if (ran.TraVeMangRan[0].IntersectsWith(vatcan.VatcanRec2[i]))
    {
        setting.intersecSound.Play();
        Restart();
    }
}
```

Hình 4.1.9.28. Gọi hàm restart khi rắn chạm vào vật cản (Lấy ví dụ từ form FormChinhVua)

  - Vấn đề thứ Hai:

```
//Nếu thức ăn trùng với vật cản thì random ra ngoài
for (int i = 0; i < vatcan.VatcanRec.Length; i++)
{
    if (thucan.ThucAnRec.IntersectsWith(vatcan.VatcanRec[i]))
    {

        thucan.RandomVitriThucAn(randthucan);
    }
}
for (int i = 0; i < vatcan.VatcanRec2.Length; i++)
{
    if (thucan.ThucAnRec.IntersectsWith(vatcan.VatcanRec2[i]))
    {

        thucan.RandomVitriThucAn(randthucan);
    }
}
```

Hình 4.1.9.29. Code xuất hiện thức ăn ở vị trí ngẫu nhiên khác nếu thức ăn trùng với vật cản (lấy ví dụ từ form FormChinhVua)

4.1.10. **Xây dựng hệ thống tính điểm và mạng**

- Đặt điểm ban đầu là 0 nếu ăn thức ăn sẽ cộng 10 điểm (Thể hiện điểm trong toolStripStatusLabel1). Khi điểm cao hơn điểm cao nhất được lưu trong label highscored thì ta sẽ gán điểm hiện tại vào label highscored luôn.
- Đặt mạng của rắn bằng 3 nếu rắn chết sẽ bị trừ 1 mạng (Thể hiện mạng trong toolStripStatusLabel2)

```
}
//Neu đầu rắn va chạm với thức ăn
if (ran.TraVeMangRan[0].IntersectsWith(thucan.ThucAnRec))
{
    setting.eatFood.Play();
    if (timer3.Interval > 2)
    {
        timer3.Interval -= 2;
    }
    diem += 10;
    if (diem >= int.Parse(setting.curvua))
    {
        highestScore.Text = diem.ToString();
    }
    ran.RanLon();
    thucan.RandomVitriThucAn(randthucan);
}
```

Hình 4.1.10.30 Code tăng điểm và tăng tốc độ của rắn khi ăn thức ăn.

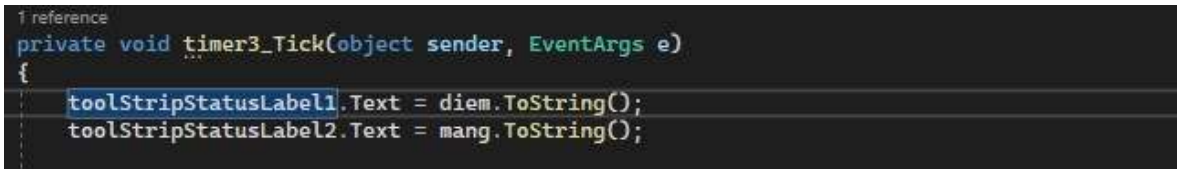
```
void Restart()
{
    mang -= 1;
    if (mang > 0)
    {
        label1.Text = "Ban chỉ còn " + mang.ToString() + " mạng nhấn Enter\n để tiếp tục!!";
        timer3.Enabled = false;
        ran = new Ran(setting.canvasHead, setting.canvasBody);
    }
    else
    {
        //tắt nhạc nền

        timer3.Enabled = false;

        GAMEOVER formganeover = new GAMEOVER();
        formganeover.ShowDialog();

        this.Close();
        string query = "Select * from TaiKhoan where TenTaiKhoan ='" + setting.curTK + "' and MatKhau = '" + setting.curMK + "'";
        if (Convert.ToInt32(modify.TaiKhoans[query][0].Vua) < diem)
        {
            setting.curvua = diem.ToString();
            string query1 = "Update TaiKhoan Set vua = '" + setting.curvua + "' where TenTaiKhoan ='" + setting.curTK + "' and MatKhau = '" + setting.curMK + "'";
            modify.Command(query1);
        }
        toolStripStatusLabel1.Text = "0";
        diem = 0;
        label1.Text = "Nhấn Enter Để Bắt đầu Chơi";
        ran = new Ran(setting.canvasHead, setting.canvasBody);
    }
}
```

Hình 4.1.10.31. Code khởi tạo lại rắn nếu còn mạng và kết thúc nếu hết mạng.



```
1 reference
private void timer3_Tick(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = diem.ToString();
    toolStripStatusLabel2.Text = mang.ToString();
```

Hình 4.1.10.32. Code thể hiện điểm và mạng trong toolStripStatusLabel1 và toolStripStatusLabel2.

4.1.11.Xây dựng hệ thống âm thanh

Khởi tạo âm thanh:

- Ở class Setting chúng em tiến hành gán file âm thành vào các đối tượng như sau:
  - soundMenu = new SoundPlayer("Form1music.wav");: Đối tượng soundMenu được sử dụng để phát âm thanh cho menu của game. Âm thanh được lấy từ tệp “Form1music.wav”.
  - eatFood = new SoundPlayer("eatFoodSound.wav");: Đối tượng eatFood được sử dụng để phát âm thanh khi rắn ăn thức ăn trong game. Âm thanh được lấy từ tệp “eatFoodSound.wav”.
  - intersecSound = new SoundPlayer("intersecSound.wav");: Đối tượng intersecSound có thể được sử dụng để phát âm thanh khi xảy ra một sự kiện cụ thể trong game, có thể là khi rắn va chạm với một vật thể nào đó. Âm thanh được lấy từ tệp “intersecSound.wav”.

Hình 4.1.11.33. Code khởi tạo gán file Âm Thanh.

Lập trình chạy âm thanh:

- Có 2 vấn để cần đặt ra khi ở lớp âm thanh:
- Vấn đề thứ nhất: Khi từ FormMenu ra các Form khác thì nhạc ngừng với **btnThoat_Click().**
- Vấn đề thứ 2: Khi hết 3 mạng trò chơi sẽ chuyển form GAMEOVER, sẽ có một đoạn nhạc ngắn làm nền.
- Lập trình giải quyết vấn đề:
- Vấn đề thứ Nhất:

```
private void btnStart_Click(object sender, EventArgs e)
{
    this.Hide();
    if (Setting.mapType == 1)
    {
        Setting.soundMenu.Stop();
        FormChinhDe formChinhDe = new FormChinhDe();
        formChinhDe.timer2.Interval = 200;
        formChinhDe.ShowDialog();
    }
    else if (Setting.mapType == 2)
    {
        Setting.soundMenu.Stop();
        FormChinhVua formChinhVua = new FormChinhVua();
        formChinhVua.timer3.Interval = 150;
        formChinhVua.ShowDialog();
    }
    else if (Setting.mapType == 3)
    {
        Setting.soundMenu.Stop();
        FormChinhKho formChinhKho = new FormChinhKho();
        formChinhKho.timer1.Interval = 100;
        formChinhKho.ShowDialog();
    }
    Setting.soundMenu.PlayLooping();
    this.Show();
}
```

Hình 4.1.11.2 Code chơi nhạc trong vòng lặp và dừng lại khi nhấn nút.

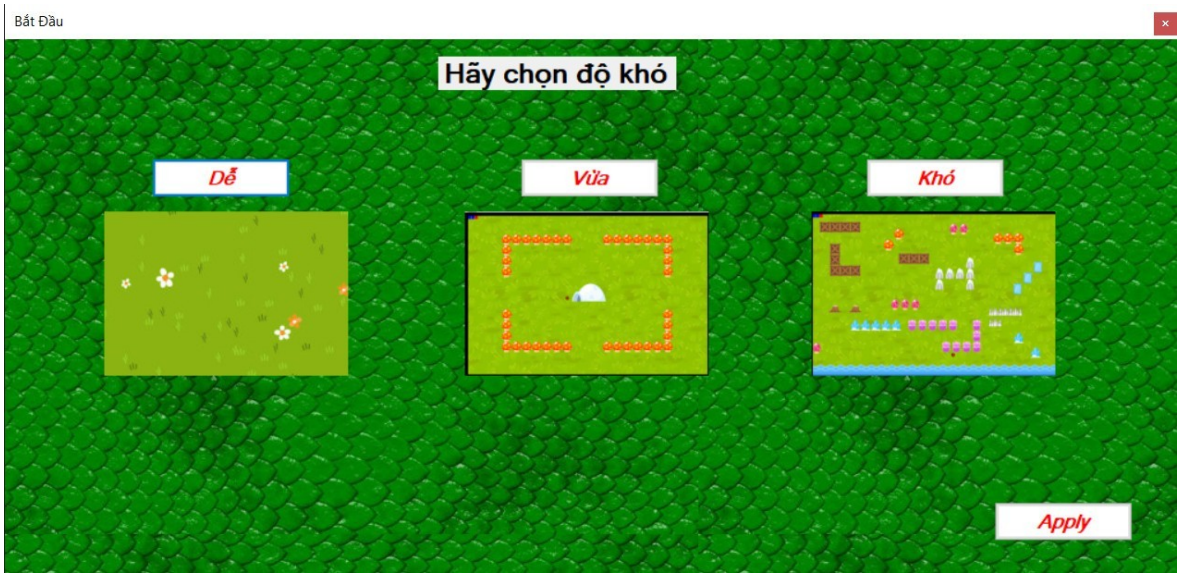
- Vấn đề thứ Hai:

Hình 4.1.11.34. Code chơi nhạc khi vào form GAMEOVER.

4.2. Xây dựng giao diện game

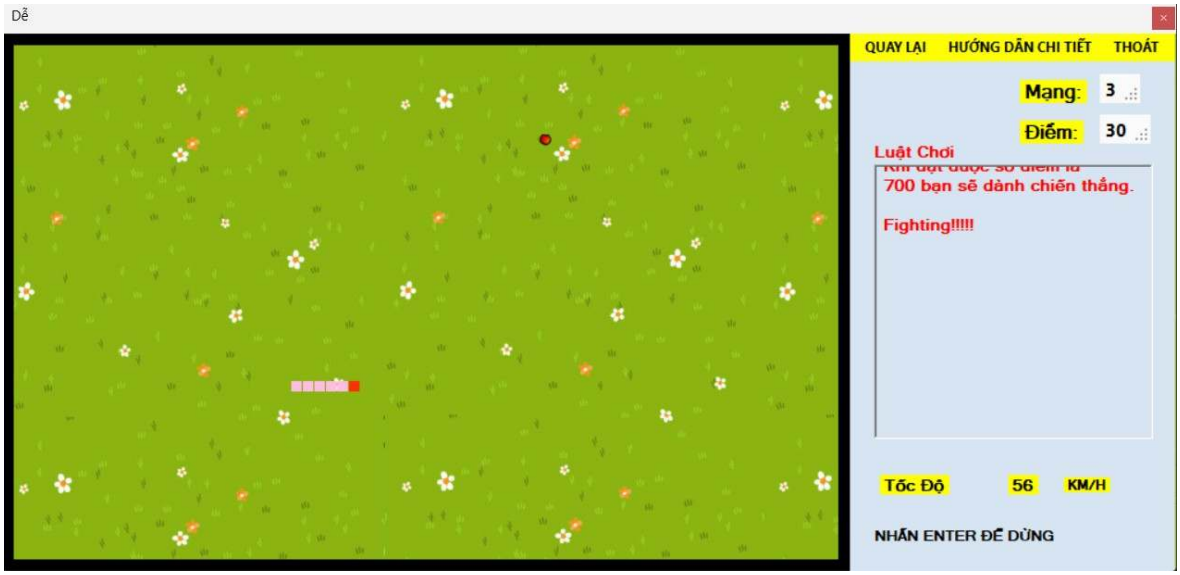
Trò chơi gồm các form

- Form thứ nhất có tên là **SetMap.cs** với chức năng là dùng để chọn mức chơi. Có 3 mức để lựa chọn Dễ, Vừa, Khó với 3 mức tốc độ timer khác nhau.

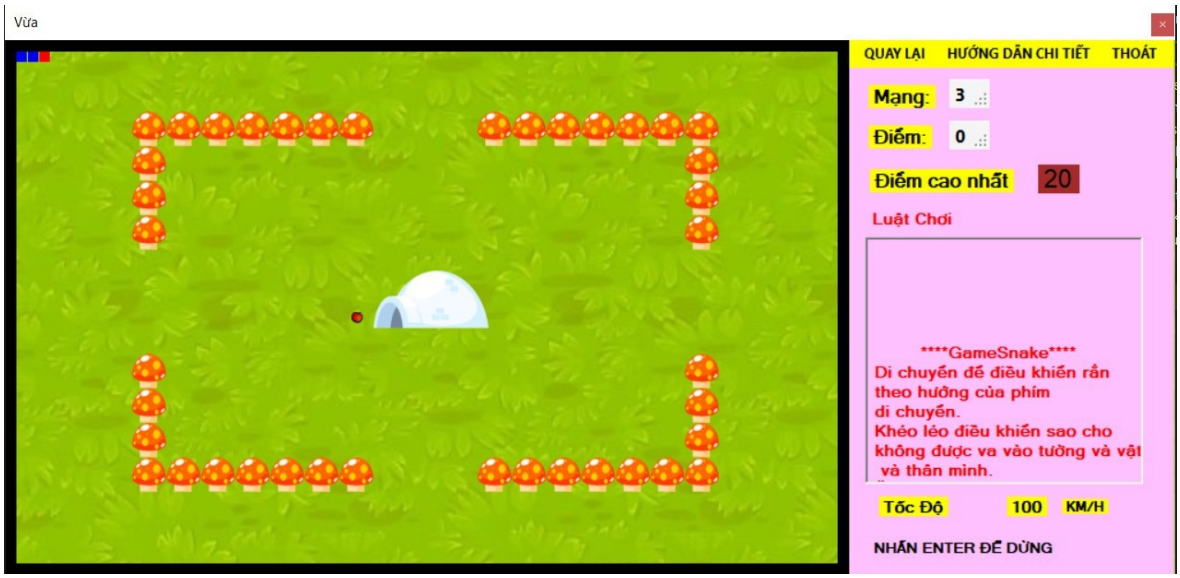


Hình 4.2.35. Giao diện SetMap.

- Form thứ hai, ba, bốn là **FormChinhVua**, **FormChinhKho** , **FormChinhDe** là form chính dùng để vẽ các đối tượng, các lớp và giúp người chơi chơi trên form.Ở tại các form này, khi nhấn nút QUAY LẠI thì người chơi sẽ trở về Menu Form, nút HƯỚNG DẪN CHI TIẾT sẽ mở lên form Instuction hướng dẫn cách chơi, còn có nút THOÁT có chức năng thoát khỏi trò chơi. Ngoài ra, tại các form này khi bận việc hay mỏi tay gì đó, người chơi có thể nhấn nút Enter để tạm dùng trò chơi.



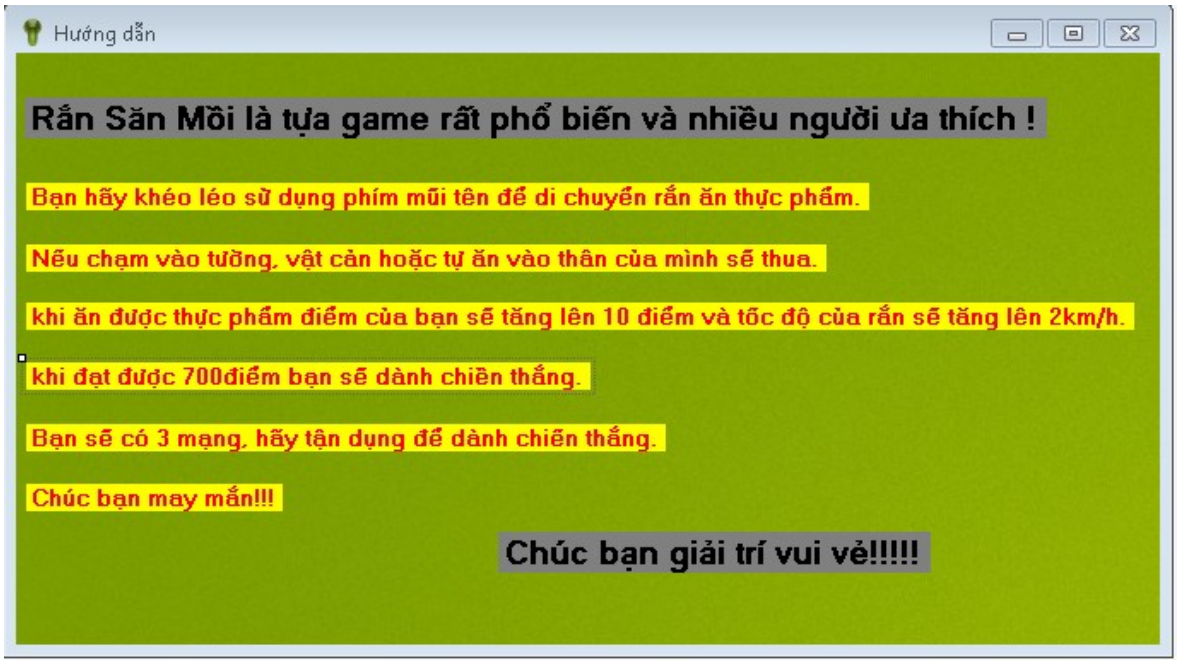
Hình 4.2.36. Giao diện FormChinhDe.



Hình 4.2.37. Giao diện FormChinhVua.

Hình 4.2.38. Giao diện FormChinhKho.

- Form thứ năm là **Instruction.cs**, dùng để hướng dẫn người chơi khi click vào "Hướng dẫn" ở FormChinh***.



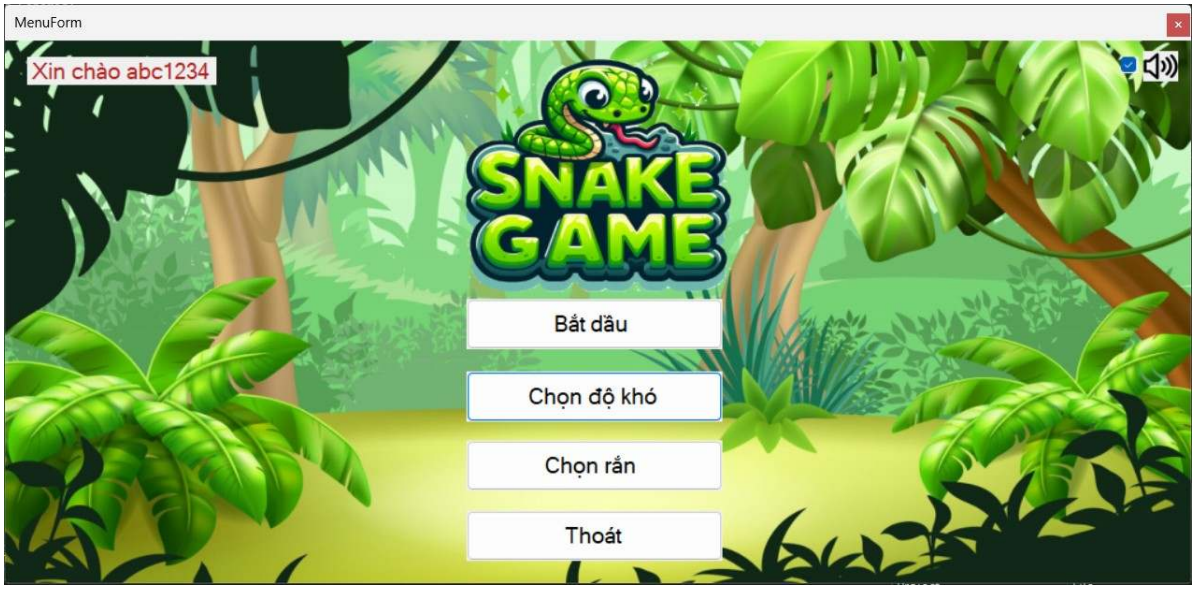
Hình 4.2.39. Giao diện Instruction.cs

- Form thứ sáu là GAMEOVER.cs, form này hiện ra khi người chơi thua cuộc.

Hình 4.2.40. Giao diện GameOver

Lưu ý: ta dùng "CreateParams" để vô hiệu hóa nút thoát mặc định trong các form.

- Form thứ bảy là form **MenuForm.cs**, hiện ra khi người chơi đăng nhập thành công vào game. Có 5 lựa chọn để "Thoát", Xem "Điểm", "Bắt đầu" màn chơi, Chọn chế độ khó, Chọn rắn để mở các form có nội dung tương đương.

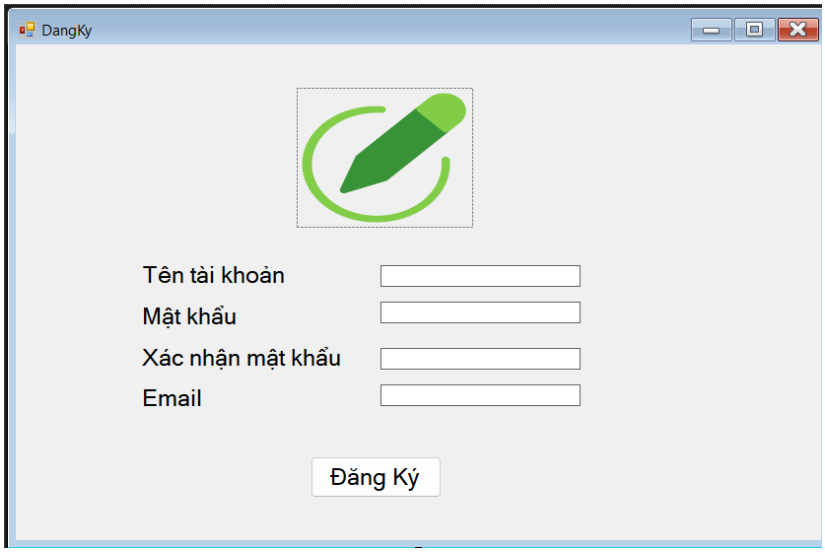


Hình 4.2.41 Giao diện MenuForm.

- Form thứ tám là form ĐangNhap.cs, hiện ra khi người chơi khởi động chương trình, có 2 lựa chọn là Quên Mật Khẩu và Đăng Ký dẫn đến các form có chức năng tương đương.

Hình 4.2.42. Giao diện đăng nhập

- Form thứ chín là form **DangKy.cs**, hiện ra khi người chơi nhấn Đăng Ký ở form DangNhap.cs.

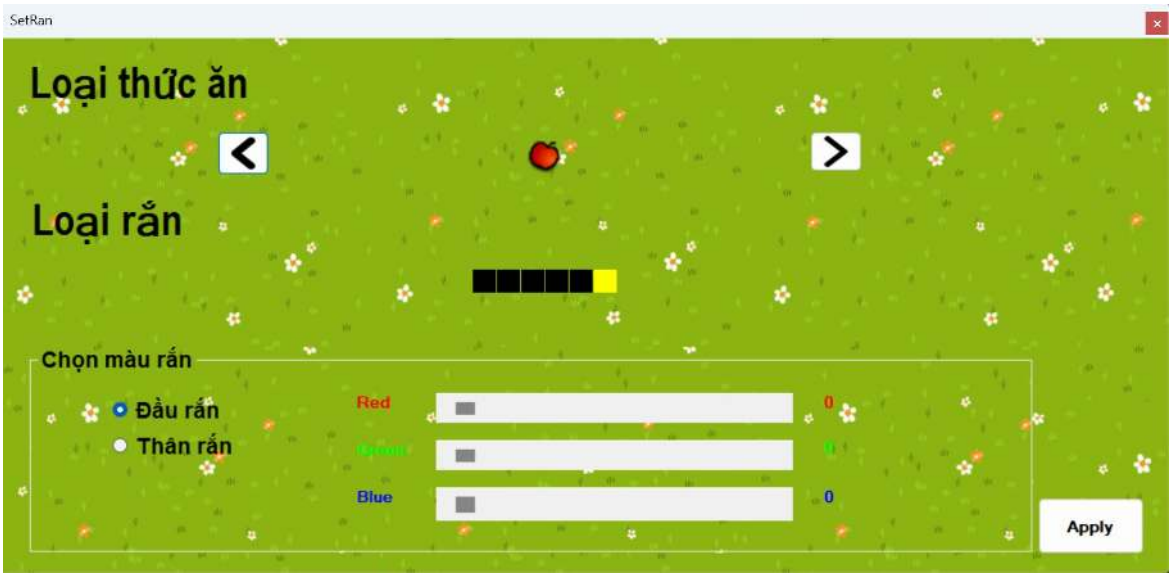


Hình 4.2.43. Giao diện đăng kí

- Form thứ chín là form **QuenMatkhau.cs**, hiện ra khi người chơi nhấn Quên Mật Khẩu ở form DangNhap.cs.

Hình 4.2.44. Giao diện QuenMatKhau

- Form thứ mười là form **SetRan.cs**, với chức năng là dùng để chọn thức ăn và màu rắn, hiện ra khi nhấn vào nút Chọn rắn.



Hình 4.2.45. Giao diện SetRan.

Chương 5 KẾT LUẬN

5.1. Các kết quả đạt được của đồ án

- Qua quá trình thực hiện đồ án này, chúng em/tôi đã đạt được nhiều kết quả

quan trọng và học được nhiều kiến thức giá trị.

- **Kỹ năng lập trình**: Chúng em/tôi đã cải thiện kỹ năng lập trình của mình, đặc biệt là với ngôn ngữ C# và nền tảng Windows Form. Việc lập trình một game từ đầu đến cuối đã giúp chúng tôi hiểu rõ hơn về quy trình phát triển phần mềm.
- **Hiểu biết về cơ sở dữ liệu**: Việc tích hợp cơ sở dữ liệu vào game đã giúp chúng tôi nắm bắt được cách lưu trữ và truy xuất dữ liệu một cách hiệu quả.
- **Thiết kế giao diện người dùng**: Chúng em/tôi đã học được cách tạo ra một giao diện người dùng thân thiện và dễ sử dụng, từ việc thiết kế menu cho đến việc hiển thị kết quả.
- **Giải quyết vấn đề**: Cuối cùng, chúng em/tôi đã học được cách giải quyết vấn đề một cách sáng tạo. Khi gặp khó khăn, chúng em/tôi đã tìm hiểu, thử nghiệm và áp dụng các giải pháp để vượt qua.
- **Làm việc nhóm**: Đồ án này cũng đã giúp chúng em/tôi nâng cao kỹ năng làm việc nhóm. Chúng em/tôi đã học cách phân chia công việc, giao tiếp hiệu quả và hỗ trợ lẫn nhau để hoàn thành mục tiêu chung.
- Chúng em/tôi tin rằng những kinh nghiệm và kiến thức mà chúng em/tôi đã học từ việc thực hiện đồ án này sẽ là nền tảng quý giá cho những dự án tiếp theo của chúng tôi.

5.2. Ưu điểm của đồ án

- **Đa chức năng**: Game "Snake" của chúng em bao gồm nhiều chế độ chơi, menu tương tác và hiển thị kết quả cuối cùng, tạo ra một trải nghiệm chơi game phong phú và đa dạng.
- **Tích hợp cơ sở dữ liệu**: Việc tích hợp cơ sở dữ liệu để lưu trữ thông tin tài khoản người chơi giúp tăng tính tương tác và cá nhân hóa trải nghiệm chơi game.
- **Thân thiện với người dùng**: Giao diện người dùng được thiết kế một cách cẩn thận để dễ sử dụng, từ việc điều hướng menu cho đến việc hiển thị kết

quả.

5.3. Hạn chế của đồ án

- **Chỉ hỗ trợ chế độ chơi đơn người**: Hiện tại, game của chúng em chỉ hỗ trợ chế độ chơi đơn người. Chúng em đang cố gắng phát triển chế độ chơi trực tuyến nhiều người chơi.
- **Cần cải tiến hiệu suất**: Mặc dù game hoạt động mượt mà, nhưng chúng em nhận ra rằng vẫn còn một số cơ hội để cải thiện hiệu suất và tối ưu hóa code.

5.4. Hướng phát triển của đồ án

- Trong tương lai, chúng em/tôi dự định phát triển game "Snake" của mình có thêm nhiều map chơi hay là các loại rắn mà các người có thể tự tạo, từ một trò chơi đơn người thành một trò chơi trực tuyến nhiều người chơi. Điều này không chỉ tăng cường tính tương tác và cạnh tranh, mà còn mang lại trải nghiệm chơi game phong phú và đa dạng hơn cho người chơi.
- Việc chuyển đổi từ một trò chơi đơn người sang trò chơi trực tuyến nhiều người chơi sẽ đòi hỏi chúng tôi phải nghiên cứu và áp dụng các công nghệ mạng, cũng như cải tiến cơ sở dữ liệu để đảm bảo việc lưu trữ và truy xuất dữ liệu một cách hiệu quả trong môi trường trực tuyến.
- Chúng em/tôi tin rằng việc thực hiện những cải tiến này sẽ không chỉ giúp chúng tôi nâng cao kỹ năng lập trình và hiểu biết về công nghệ, mà còn giúp game "Snake" của chúng tôi trở nên thú vị và hấp dẫn hơn.

Tài liệu tham khảo

[1] https://stackoverflow.com

[2] Slide bài giảng Lập trình trực quan của cô Nguyễn Thị Xuân Hương

[3] TopDev.vn

[4] CodeLearn.io

[5] Nguồn âm thanh của game: www.zedge.net

[6] Nguồn hình nền game ở form chính và vật cản: www.shutterstock.com, www.gameart2d.com