ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

**KHOA VẬT LÝ-VẬT LÝ KỸ THUẬT**

**----------------□□----------------**

**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

**(PHY10609)**

***Đề tài:***

**PHÁT TRIỂN GAME FLAPPY ANYTHING**

**GV: ThS. Nguyễn Vương Thùy Ngân**

**Nhóm thực hiện: Cá mặp cạp đầu**

|  |  |
| --- | --- |
| **Trần Đức Lương** | **22130096** |
| **Thân Thành** | **22130167** |
| **Phan Chánh Tính** | **22130190** |

**Lớp: 22VLH1TN**

**--- TP.HCM, 12/2024 ---**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA VẬT LÝ – VẬT LÝ KỸ THUẬT**

**BỘ MÔN VẬT LÝ TIN HỌC**

**🙢🕮🙠**

**A blue and white logo

Description automatically generated**

**ĐỒ ÁN MÔN HỌC**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

*Đề tài****:***

**PHÁT TRIỂN GAME FLAPPY ANYTHING**

**GV: ThS. Nguyễn Vương Thùy Ngân**

**Nhóm thực hiện: Cá mặp cạp đầu**

|  |  |
| --- | --- |
| **Trần Đức Lương** | **22130096** |
| **Thân Thành** | **22130167** |
| **Phan Chánh Tính** | **22130190** |

# LỜI CẢM ƠN

Trong suốt quá trình học môn Lập trình Hướng đối tượng, nhóm Cá Mập Cạp Đầu (22VLTH) đã có cơ hội khám phá sâu hơn về chuyên ngành Vật lý Tin học. Bài báo cáo này chính là kết quả từ những kiến thức mà chúng em đã tích lũy được. Không chỉ dừng lại ở việc tìm hiểu lý thuyết, các thành viên trong nhóm còn học hỏi được nhiều kỹ năng bổ ích, nâng cao tinh thần làm việc nhóm, động viên lẫn nhau trong học tập và quan trọng hơn cả là rút ra những kinh nghiệm quý báu để vững bước trên con đường Đại học.

Nhóm chúng em xin gửi lời cảm ơn sâu sắc đến cô Nguyễn Vương Thùy Ngân – người đã tận tình hướng dẫn và trang bị cho chúng em những kiến thức cần thiết về Lập trình Hướng đối tượng, đồng thời chia sẻ các kỹ năng mềm quan trọng trong quá trình làm việc nhóm và thực hiện báo cáo đồ án. Nhờ sự hướng dẫn của cô, chúng em đã hiểu rõ hơn về chuyên ngành mà mình đang theo đuổi và thêm tự tin trong việc ứng dụng kiến thức vào thực tiễn.

Vì đây là lần đầu tiên nhóm chúng em thực hiện một bài báo cáo đồ án, chắc chắn sẽ không tránh khỏi những thiếu sót. Chúng em rất mong nhận được những ý kiến đóng góp quý báu từ phía cô để có thể ngày càng hoàn thiện hơn trong học tập cũng như các dự án tiếp theo.

Cuối cùng, nhóm Cá Mập Cạp Đầu xin kính chúc cô Nguyễn Vương Thùy Ngân thật nhiều sức khỏe và thành công trong công việc cũng như cuộc sống. Chúng em cũng xin chúc các bạn trong lớp 22VLTH luôn nỗ lực để đạt được những thành tựu đáng tự hào trong hành trình học tập.

Trân trọng,  
**Nhóm Cá Mập Cạp Đầu (22VLTH)**

# LỜI MỞ ĐẦU

Trong thời đại công nghệ số hiện nay, lập trình đã trở thành một kỹ năng cốt lõi, không chỉ đối với lĩnh vực công nghệ thông tin mà còn trong nhiều lĩnh vực khác như giáo dục, giải trí và khoa học. Đặc biệt, lập trình hướng đối tượng với sự mạnh mẽ và linh hoạt của nó đã trở thành một phương pháp tư duy quan trọng trong việc giải quyết các bài toán phức tạp và xây dựng các ứng dụng thực tiễn.

Nhóm chúng em đã lựa chọn phát triển trò chơi *Flappy Anything* – một phiên bản sáng tạo dựa trên trò chơi nổi tiếng *Flappy Bird*. Đây là một trò chơi điện tử đơn giản nhưng đầy sức hút, được nhóm chúng em tái hiện với những cải tiến mới, mang tính cá nhân hóa cao, cho phép người chơi tự do lựa chọn nhân vật và bản đồ. Qua dự án này, chúng em áp dụng ngôn ngữ lập trình C++ và các nguyên tắc lập trình hướng đối tượng để tạo ra một sản phẩm hoàn chỉnh và sáng tạo.

Không chỉ dừng lại ở việc củng cố kiến thức về lập trình hướng đối tượng, nhóm còn sử dụng thư viện SDL2 (*Simple DirectMedia Layer*) để xử lý đồ họa, âm thanh và sự kiện, giúp hiện thực hóa ý tưởng trò chơi một cách chuyên nghiệp. Dự án này là cơ hội để chúng em rèn luyện khả năng tổ chức mã nguồn chặt chẽ, dễ mở rộng và dễ bảo trì. Đồng thời, đây cũng là dịp để chúng em phát triển kỹ năng làm việc nhóm, nâng cao tư duy sáng tạo và khả năng giải quyết vấn đề.

Bài báo cáo này sẽ trình bày chi tiết quá trình phát triển trò chơi *Flappy Anything*, từ ý tưởng ban đầu, phân tích thiết kế, triển khai đến đánh giá kết quả. Chúng em hy vọng rằng, qua dự án này, không chỉ khẳng định sự hiểu biết về lập trình mà còn thể hiện được sự nhiệt huyết và tinh thần học hỏi không ngừng của nhóm.

Chúng em xin gửi lời cảm ơn chân thành đến cô Nguyễn Vương Thùy Ngân, giảng viên môn Lập trình Hướng đối tượng, đã tận tình hướng dẫn và cung cấp những kiến thức quý giá để chúng em hoàn thành dự án này. Đồng thời, chúng em mong nhận được những ý kiến đóng góp từ cô để có thể cải thiện kỹ năng và tiếp tục phát triển trong các dự án tương lai.

# MỤC LỤC

[LỜI CẢM ƠN 3](#_Toc185332955)

[LỜI MỞ ĐẦU 4](#_Toc185332956)

[MỤC LỤC 6](#_Toc185332957)

[DANH SÁCH CÁC HÌNH 7](#_Toc185332958)

[NỘI DUNG CHÍNH 8](#_Toc185332959)

[1. Giới thiệu về dự án 8](#_Toc185332960)

[1. Ý tưởng và mục tiêu dự án 8](#_Toc185332961)

[2. Tổng quan kiến thức liên quan 10](#_Toc185332962)

[3. Phần Phân Tích và Thiết Kế 12](#_Toc185332963)

[4. Phần Phân Tích và Thiết Kế 17](#_Toc185332964)

[5. Kết Quả và Thảo Luận 18](#_Toc185332965)

[KẾT LUẬN 21](#_Toc185332966)

[MÃ NGUỒN 22](#_Toc185332967)

# DANH SÁCH CÁC HÌNH

[Hình 1. UML của game – Mô hình hóa phương pháp triển khai game Flappy Anything 15](#_Toc185332939)

[Hình 2. Cấu trúc thư mục dự án Flappy Anything, với các thư mục chứa mã nguồn, tệp tài nguyên, và các tập tin cấu hình. 17](#_Toc185332940)

[Hình 3. Game.h 22](#_Toc185332941)

[Hình 4 Game.cpp 23](#_Toc185332942)

[Hình 5 Doge.h 24](#_Toc185332943)

[Hình 6 Doge.cpp 25](#_Toc185332944)

[Hình 7 Land.h 26](#_Toc185332945)

[Hình 8 Land.cpp 27](#_Toc185332946)

[Hình 9 Lib.h 28](#_Toc185332947)

[Hình 10 Lib.cpp 30](#_Toc185332948)

[Hình 11 main.cpp 31](#_Toc185332949)

[Hình 12 pipe.h 32](#_Toc185332950)

[Hình 13 Pipe.cpp 33](#_Toc185332951)

[Hình 14 sound.h 34](#_Toc185332952)

[Hình 15 sound.cpp 35](#_Toc185332953)

[Hình 16 Utils.h – File chứa các hằng số, đường dẫn mà nhóm định nghĩa 36](#_Toc185332954)

# NỘI DUNG CHÍNH

## Giới thiệu về dự án

Trong dự án này, nhóm chúng em phát triển trò chơi **Flappy Anything** – một phiên bản sáng tạo dựa trên trò chơi *Flappy Bird* nổi tiếng. Sản phẩm được xây dựng bằng **ngôn ngữ lập trình C++** kết hợp với **thư viện SDL2** (Simple DirectMedia Layer 2) để xử lý đồ họa, âm thanh và sự kiện.

### 1. Ý tưởng và mục tiêu dự án

#### Ý tưởng phát triển trò chơi

Dựa trên trò chơi *Flappy Bird* nổi tiếng, nhóm chúng em mong muốn phát triển một phiên bản mới mẻ và sáng tạo mang tên **Flappy Anything**. Không chỉ giữ lại lối chơi đơn giản nhưng đầy thử thách, nhóm còn tích hợp thêm nhiều yếu tố thú vị để nâng cao trải nghiệm người chơi. **Flappy Anything** cho phép người chơi tự do lựa chọn nhân vật, tùy chỉnh bản đồ và đối mặt với các thử thách độc đáo chưa từng có.

Điểm đặc biệt của **Flappy Anything** nằm ở khả năng **cá nhân hóa**:

* Người chơi có thể lựa chọn các nhân vật khác nhau như **chó Doge, mèo, chim** hoặc thậm chí tải lên **nhân vật riêng** của mình.
* Giao diện bản đồ cũng có thể thay đổi linh hoạt, từ **chế độ sáng/ tối, giáng sinh** cho đến các thiết kế độc quyền theo sở thích của người chơi.
* Các **chướng ngại vật** không còn cố định mà được thiết kế linh hoạt với độ khó ngẫu nhiên, nhanh chậm bất ngờ, và các bản đồ thay đổi sau mỗi 30 giây để làm phân tán sự chú ý người chơi để tăng độ khó cho trò chơi.

Chúng em tin rằng sự kết hợp giữa **đồ họa thú vị**, **âm thanh sống động**, và **tính năng độc đáo** sẽ mang đến một trải nghiệm vừa quen thuộc nhưng cũng đầy cuốn hút và thách thức.

#### Mục tiêu dự án

Mục tiêu chính của dự án Flappy Anything là tạo ra một trò chơi đơn giản nhưng hấp dẫn, đồng thời ứng dụng triệt để kiến thức lập trình hướng đối tượng và thư viện SDL2. Cụ thể:

* **Nâng cao kỹ năng lập trình C++ và SDL2**:
* Vận dụng các nguyên tắc lập trình hướng đối tượng (OOP) để xây dựng hệ thống mã nguồn rõ ràng, dễ bảo trì và mở rộng.
* Khai thác thư viện SDL2 để xử lý đồ họa, âm thanh và sự kiện người chơi một cách hiệu quả.
* **Tạo ra trải nghiệm game độc đáo và thú vị**:
* Cung cấp khả năng **tùy chỉnh nhân vật** và **bản đồ chơi** theo sở thích cá nhân.
* Thiết kế các chướng ngại vật với nhiều cấp độ khó và tình huống bất ngờ, giúp trò chơi luôn mới mẻ và thách thức.
* Tích hợp **hiệu ứng âm thanh và hình ảnh** sống động, tạo sự hứng thú cho người chơi.
* **Phát triển kỹ năng làm việc nhóm và sáng tạo**:
* Tăng cường khả năng làm việc nhóm, phân chia công việc hợp lý và quản lý tiến độ dự án.
* Khuyến khích sự sáng tạo thông qua việc thiết kế giao diện, phát triển tính năng mới và tối ưu hóa trò chơi.
* **Tạo tiền đề cho các dự án game phức tạp hơn**:
* Tích lũy kinh nghiệm trong phát triển game, từ khâu thiết kế, lập trình, đến thử nghiệm và hoàn thiện sản phẩm.
* Hướng đến khả năng mở rộng trò chơi trong tương lai, ví dụ như phát triển thêm **chế độ nhiều người chơi** hoặc các **sự kiện đặc biệt** trong game.

Với những ý tưởng độc đáo và mục tiêu rõ ràng, **Flappy Anything** không chỉ là một trò chơi giải trí mà còn là minh chứng cho sự sáng tạo, nỗ lực và tinh thần học hỏi của nhóm. Sản phẩm này hứa hẹn mang đến cho người chơi những giây phút thư giãn vui nhộn nhưng cũng đầy thử thách và hấp dẫn.

### Tổng quan kiến thức liên quan

#### Kiến thức về lập trình hướng đối tượng (OOP)

Lập trình hướng đối tượng đóng vai trò cốt lõi trong việc tổ chức và quản lý mã nguồn của trò chơi. Các khái niệm chính được áp dụng bao gồm:

* **Lớp và đối tượng**:
* Trò chơi được chia thành các lớp như Game, Doge, Pipe, Land, Sound, và LTexture, mỗi lớp đảm nhiệm một vai trò cụ thể.
* Ví dụ:
* Lớp Doge đại diện cho nhân vật người chơi, quản lý trạng thái và vị trí của nhân vật.
* Lớp Pipe quản lý các chướng ngại vật và xử lý va chạm.
* Lớp Sound đảm nhiệm việc phát nhạc nền và hiệu ứng âm thanh.
* **Tính kế thừa**:
* Các lớp con kế thừa từ lớp LTexture để tận dụng các phương thức chung về xử lý đồ họa, giúp mã nguồn ngắn gọn và dễ mở rộng.
* **Tính đóng gói**:
* Dữ liệu và hàm xử lý của từng lớp được đóng gói lại để tránh sự can thiệp từ bên ngoài, đảm bảo tính bảo mật và ổn định của chương trình.
* **Tính trừu tượng**:
* Nhóm đã tách biệt các chức năng thành các lớp cụ thể để dễ dàng quản lý, như lớp Sound chỉ tập trung xử lý âm thanh, hay lớp Land quản lý mặt đất.

#### Kiến thức về thư viện SDL2

**SDL2 (Simple DirectMedia Layer)** là thư viện đồ họa mã nguồn mở mạnh mẽ, hỗ trợ phát triển các trò chơi và ứng dụng đa nền tảng. Nhóm đã sử dụng SDL2 để:

* **Xử lý đồ họa**:
* Sử dụng SDL\_Texture, SDL\_Renderer, và SDL\_Rect để vẽ hình ảnh nhân vật, ống nước, mặt đất, và giao diện trò chơi lên màn hình.
* Áp dụng các kỹ thuật như **render texture với góc xoay (rotation)** và **lật hình ảnh (flip)** để tạo hiệu ứng sinh động cho nhân vật.
* **Xử lý sự kiện**:
* Sử dụng **SDL\_Event** để bắt các sự kiện từ người chơi như nhấn phím, thả phím, hoặc thoát game. Ví dụ: Nhấn phím “Space” để điều khiển nhân vật nhảy lên.
* **Xử lý âm thanh**:
* Tận dụng SDL2\_Mixer để phát các hiệu ứng âm thanh như tiếng nhân vật nhảy, va chạm chướng ngại vật, hoặc nhạc nền.
* **Cập nhật khung hình và tối ưu hóa tốc độ**:
* Nhóm sử dụng **SDL\_Delay** và **SDL\_GetTicks** để đảm bảo tốc độ khung hình ổn định và mượt mà (ví dụ: 60 FPS).

#### Xử lý va chạm và vật lý cơ bản

Để đảm bảo tính chính xác và thú vị trong trò chơi, nhóm đã áp dụng các kỹ thuật sau:

* **Kiểm tra va chạm**: Sử dụng các tọa độ của nhân vật và ống nước (thông qua SDL\_Rect) để phát hiện va chạm. Nếu hai vùng hình chữ nhật giao nhau, trò chơi sẽ kết thúc.
* **Hiệu ứng rơi tự do**: Tạo lực rơi cho nhân vật bằng cách tăng dần giá trị tọa độ Y theo thời gian, mô phỏng chuyển động trọng lực.
* **Điều khiển nhảy**: Khi người chơi nhấn phím, lực nhảy sẽ được áp dụng để nhân vật di chuyển lên trên, chống lại lực rơi.

#### Quản lý tài nguyên và tối ưu hóa mã nguồn

* **Tối ưu bộ nhớ**: Nhóm sử dụng con trỏ thông minh và quản lý bộ nhớ thủ công để tránh rò rỉ bộ nhớ trong quá trình tạo và hủy tài nguyên (texture, âm thanh).
* **Tái sử dụng mã nguồn**: Các lớp như LTexture đảm nhiệm việc nạp và hiển thị hình ảnh, giúp các lớp khác sử dụng lại mà không cần viết lại mã.
* **Phân chia module**: Chương trình được chia thành nhiều file .h và .cpp để tổ chức mã nguồn rõ ràng, thuận tiện cho bảo trì và mở rộng.

#### Tinh thần làm việc nhóm và công cụ hỗ trợ

* Trong quá trình làm việc, nhóm đã phối hợp để phân chia công việc theo các phần như xử lý đồ họa, âm thanh, logic va chạm và viết tài liệu báo cáo.
* Các công cụ như **GitHub** được sử dụng để quản lý phiên bản mã nguồn, đảm bảo tiến độ và hạn chế xung đột khi code.

### Phần Phân Tích và Thiết Kế

#### Phân Tích Yêu Cầu

Dựa trên ý tưởng phát triển trò chơi **Flappy Anything** - một phiên bản mở rộng và sáng tạo của **Flappy Bird**, các yêu cầu cơ bản của trò chơi được xác định như sau:

* **Yêu cầu chức năng**:
* Người chơi điều khiển nhân vật bay qua các ống nước bằng cách nhấn phím.
* Trò chơi kết thúc khi nhân vật va chạm với ống nước hoặc rơi xuống đất.
* Hiển thị điểm số và cho phép người chơi chơi lại khi kết thúc.
* Người chơi có thể tùy chọn nhân vật và bản đồ trước khi bắt đầu.
* **Yêu cầu phi chức năng**:
* Đồ họa mượt mà và ổn định ở tốc độ **60 FPS**.
* Âm thanh sống động bao gồm nhạc nền và hiệu ứng khi nhân vật nhảy hoặc va chạm.
* Tổ chức mã nguồn theo kiến trúc hướng đối tượng để dễ bảo trì và mở rộng.

#### Mô Hình Phân Tích

Dựa trên yêu cầu và tính năng của trò chơi, nhóm đã sử dụng các khái niệm lập trình hướng đối tượng để thiết kế các thành phần chính. Dưới đây là mô hình UML đã được xây dựng:

* **Lớp Game**:
* Chịu trách nhiệm điều khiển toàn bộ vòng đời của trò chơi (khởi tạo, chạy chính, kết thúc).
* Quản lý trạng thái của trò chơi thông qua enum input (QUIT, PLAY, NONE, PAUSE).
* **Lớp Doge (Nhân vật chính)**:
* Đại diện cho nhân vật trong trò chơi với các thuộc tính như vị trí, trạng thái và chuyển động.
* Chịu trách nhiệm cập nhật vị trí và kiểm tra va chạm.
* **Lớp Pipe (Chướng ngại vật)**:
* Sinh ra và quản lý các ống nước trên màn hình.
* Cập nhật vị trí, phát hiện va chạm với nhân vật.
* **Lớp Land (Mặt đất)**: Hiển thị và cập nhật mặt đất, đồng thời phát hiện va chạm khi nhân vật rơi xuống.
* **Lớp Sound**: Quản lý âm thanh, bao gồm nhạc nền và hiệu ứng âm thanh (nhảy, va chạm).
* **Lớp LTexture**:
* Quản lý hình ảnh và texture, hỗ trợ các phương thức để vẽ hình ảnh lên màn hình.
* Đây là lớp cơ sở được kế thừa bởi các lớp khác như Doge, Pipe, và Land.
* **Lớp Position**: Quản lý tọa độ của các đối tượng trong trò chơi như nhân vật, ống nước và mặt đất.

#### Sơ Đồ Lớp UML

Sơ đồ lớp UML được xây dựng chi tiết với các lớp, thuộc tính và phương thức cụ thể như sau (Hình 1):

* **Lớp Game**
* enum input { QUIT, PLAY, NONE, PAUSE }
* Phương thức: init(), run(), update(), render(), close()
* **Lớp Doge (Nhân vật chính)**
* Thuộc tính: angle, time, x, y, state, path, posDoge
* Phương thức: init(), update(), render(), fall(), resetTime(), checkCollision()
* **Lớp Pipe (Chướng ngại vật)**
* Thuộc tính: randMin, randMax
* Phương thức: init(), update(), render(), getWidth(), getHeight()
* **Lớp Sound**
* Thuộc tính: Mix\_Chunk\*, Mix\_Music\*
* Phương thức: init(), playJump(), playCollision(), playGroundSound(), stopGroundSound()

A diagram of a computer

Description automatically generated

Hình . UML của game – Mô hình hóa phương pháp triển khai game Flappy Anything

* **Lớp Land**
* Thuộc tính: posLand
* Phương thức: init(), update(), render()
* **Lớp LTexture** (Lớp cơ sở)
* Thuộc tính: SDL\_Texture\*, width, height
* Phương thức: load(), render(), free()

#### Sơ đồ Luồng dữ liệu (Flowchart)

Dưới đây là các bước chính trong luồng hoạt động của trò chơi **Flappy Anything**:

* + 1. **Khởi tạo trò chơi:**

Nạp các tài nguyên như hình ảnh, âm thanh và khởi tạo các đối tượng (Doge, Pipe, Land, Sound).

* + 1. **Vòng lặp chính**:
* Bắt sự kiện từ người chơi (nhấn phím, thoát).
* Cập nhật trạng thái và vị trí của các đối tượng.
* Kiểm tra va chạm giữa nhân vật và ống nước hoặc mặt đất.
* Vẽ (render) các đối tượng lên màn hình.
  + 1. **Xử lý kết thúc trò chơi**:
* Hiển thị điểm số.
* Chờ người chơi lựa chọn chơi lại hoặc thoát.
  + 1. **Đóng trò chơi**:
* Giải phóng bộ nhớ và thoát chương trình.

Nhóm đã có tạo sơ đồ luồng dữ liệu trên nền tảng Lucidchart, đường dẫn ở phần 1 trong mục Mã Nguồn.

#### Tổ Chức Mã Nguồn

Dự án được tổ chức theo cấu trúc sau để đảm bảo rõ ràng và dễ bảo trì:



Hình . Cấu trúc thư mục dự án Flappy Anything, với các thư mục chứa mã nguồn, tệp tài nguyên, và các tập tin cấu hình.

### Phần Phân Tích và Thiết Kế

#### Chuẩn Bị Môi Trường Phát Triển

Trước khi bắt đầu triển khai, nhóm đã chuẩn bị môi trường phát triển bằng cách cài đặt và cấu hình các công cụ cần thiết:

* **Cài đặt C++ và SDL2**:
* Cài đặt trình biên dịch C++ như **GCC** hoặc **MinGW** cho hệ điều hành Windows.
* Cài đặt thư viện **SDL2** và **SDL2\_Mixer** để xử lý đồ họa, âm thanh và sự kiện.
* Thiết lập các biến môi trường để đảm bảo các thư viện SDL2 có thể được tìm thấy trong quá trình biên dịch.
* **Công cụ phát triển**:
* **Code::Blocks** hoặc **Visual Studi** được sử dụng làm IDE.
* **Git** để quản lý phiên bản mã nguồn và hợp tác nhóm.

#### Triển Khai Các Lớp Cơ Bản

Dưới đây là triển khai các lớp chính trong dự án **Flappy Anything** – Tất cả các triển khai class nằm ở mục **MÃ NGUỒN** phía dưới:

* + 1. **Lớp game**:
* Giải phóng bộ nhớ và thoát chương trình. Hình 3 và 4.
  + 1. **Lớp Doge (Nhân vật chính)**:
* Lớp này đại diện cho nhân vật của người chơi, bao gồm các thuộc tính như vị trí, góc nghiêng, và phương thức để di chuyển và kiểm tra va chạm. Hình 5 và 6.
  + 1. **Lớp Pipe (Chướng ngại vật)**
* Lớp này sẽ tạo và cập nhật các ống nước, được tạo ra với các vị trí ngẫu nhiên và di chuyển từ phải sang trái. Hình 8 và 9
  + 1. **Lớp sound**:
* Lớp này sử dụng thư viện **SDL2\_Mixer** để phát âm thanh cho các hiệu ứng trong trò chơi, như nhạc nền, tiếng nhảy và va chạm. Hình 10 và 11
  + 1. **Lớp LTexture**:
* Đây là lớp cơ sở để quản lý các hình ảnh (textures). Các lớp như Doge, Pipe sẽ kế thừa lớp này để sử dụng các phương thức render và load hình ảnh. Hình 12 và 13

### Kết Quả và Thảo Luận

#### Kết quả triển khai

Sau quá trình triển khai và kiểm thử, trò chơi **Flappy Anything** đã hoàn thành với các tính năng và chức năng cơ bản hoạt động ổn định. Các kết quả đạt được trong dự án bao gồm:

Đường dẫn triển khai dự án:

https://github.com/TranDucLuong2201/CA-MAP-CAP-DAU

* **Giao Diện Người Dùng (UI)**:  
  Giao diện của trò chơi đơn giản nhưng trực quan, người chơi có thể dễ dàng điều khiển nhân vật Doge và nhìn thấy điểm số của mình trên màn hình. Các yếu tố đồ họa được hiển thị rõ ràng, không có hiện tượng nhấp nháy hay lỗi vẽ màn hình.
* **Chức Năng Chính**:  
  Các chức năng chính của trò chơi như di chuyển nhân vật, tạo ống chướng ngại vật, và cập nhật điểm số hoạt động trơn tru. Người chơi có thể nhấn phím cách để cho Doge nhảy lên, tránh các ống chướng ngại vật, và trò chơi tiếp tục cho đến khi xảy ra va chạm.
* **Xử Lý Âm Thanh**:  
  Âm thanh được tích hợp cho các hiệu ứng như nhảy, va chạm, và nhạc nền. Mọi âm thanh đều phát đúng lúc và không có trễ, mang lại trải nghiệm người chơi mượt mà.
* **Tính Năng Tùy Chọn**:  
  Trò chơi cho phép người chơi tùy chọn nhân vật và bản đồ, tạo ra sự đa dạng và tính cá nhân hóa cao, khiến trò chơi trở nên thú vị và mới mẻ hơn.
* **Tính Mở Rộng**:  
  Mã nguồn được tổ chức tốt, dễ dàng mở rộng và bảo trì. Các lớp đối tượng như **Doge**, **Pipe**, **Land** đều tuân theo nguyên tắc lập trình hướng đối tượng, giúp mã nguồn dễ dàng tái sử dụng và phát triển thêm tính năng trong tương lai, ví dụ như thêm nhiều nhân vật, chế độ chơi, hay các cấp độ khó.

#### Thảo luận

Trong quá trình triển khai trò chơi **Flappy Anything**, nhóm gặp phải một số vấn đề và thử thách, nhưng cũng đã tìm được những giải pháp phù hợp để vượt qua:

* **Xử Lý Va Chạm**:  
  Một trong những thử thách lớn là xử lý va chạm giữa nhân vật và các ống chướng ngại vật. Ban đầu, việc kiểm tra va chạm giữa hình ảnh động (Doge) và các đối tượng tĩnh (Pipe) gặp một số khó khăn, đặc biệt là khi nhân vật di chuyển với tốc độ nhanh. Tuy nhiên, chúng tôi đã sử dụng các hình chữ nhật bao quanh (bounding box) để kiểm tra va chạm, và phương pháp này đã cho kết quả chính xác.
* **Hiệu Suất**:  
  Một vấn đề khác là tối ưu hóa hiệu suất khi trò chơi chạy trên các máy tính cấu hình thấp. Khi có quá nhiều đối tượng trên màn hình (như nhiều ống chướng ngại vật), trò chơi có thể bị giật. Để khắc phục, nhóm đã tối ưu hóa cách quản lý bộ nhớ và giảm tần suất cập nhật đối tượng không cần thiết trong vòng lặp chính.
* **Cải Tiến Đồ Họa**:  
  Đồ họa của trò chơi mặc dù đã hoạt động ổn định nhưng có thể cải thiện thêm để nâng cao trải nghiệm người chơi. Đặc biệt, việc tạo ra nhiều loại ống chướng ngại vật và nền động sẽ giúp trò chơi trở nên sinh động hơn. Nhóm dự định sẽ cải thiện đồ họa trong các phiên bản tiếp theo của trò chơi.
* **Tính Năng Tùy Biến**:  
  Việc cho phép người chơi lựa chọn nhân vật và bản đồ đã tạo ra sự thú vị trong trò chơi. Tuy nhiên, một số người chơi có thể cảm thấy khó khăn trong việc điều khiển Doge ở mức độ cao khi bản đồ có độ khó lớn. Nhóm sẽ nghiên cứu thêm cách cải thiện cơ chế điều khiển và cân bằng độ khó để mang đến trải nghiệm chơi game tốt hơn.
* **Kỹ Thuật Lập Trình**:  
  Quá trình áp dụng **lập trình hướng đối tượng (OOP)** đã giúp nhóm tổ chức mã nguồn hiệu quả và dễ dàng mở rộng. Các lớp như **Doge**, **Pipe**, và **Land** đều kế thừa từ lớp cơ sở **LTexture**, giúp việc quản lý các đối tượng đồ họa trở nên gọn gàng và dễ bảo trì. Việc sử dụng SDL2 cho việc xử lý đồ họa và âm thanh cũng giúp đơn giản hóa việc phát triển trò chơi.

# KẾT LUẬN

Trong quá trình phát triển trò chơi **Flappy Anything**, chúng tôi đã áp dụng và tích hợp nhiều kiến thức lý thuyết và kỹ thuật lập trình C++ cơ bản lẫn nâng cao. Điều này bao gồm việc sử dụng các đặc điểm của lập trình hướng đối tượng như tổ chức mã nguồn thành các lớp, kế thừa, đóng gói và đa hình, kết hợp với việc khai thác sức mạnh của thư viện SDL2 để xử lý đồ họa, âm thanh và sự kiện trong trò chơi.

Chúng tôi đã tập trung giải quyết các yêu cầu cốt lõi như xây dựng gameplay mượt mà, thiết kế giao diện trực quan, xử lý va chạm chính xác, và đảm bảo tốc độ khung hình ổn định. Ngoài ra, các chức năng mở rộng như tùy chỉnh nhân vật, nhạc nền, giao diện (theme), và thay đổi bản đồ động đã được phát triển để tăng tính hấp dẫn và đa dạng cho trò chơi.

Trong kết luận, để hoàn thiện một trò chơi như **Flappy Anything**, việc kết hợp linh hoạt các kiến thức về lập trình hướng đối tượng, quản lý tài nguyên và xử lý đồ họa, âm thanh là vô cùng quan trọng. Bên cạnh đó, quy trình thiết kế hệ thống module hóa đã giúp chúng tôi dễ dàng mở rộng tính năng và cải thiện hiệu suất trò chơi. Những trải nghiệm và bài học từ quá trình phát triển này sẽ là nền tảng quý báu để chúng tôi áp dụng vào các dự án lập trình trong tương lai.

**---Hết---**

# MÃ NGUỒN

Tổng hợp mã triển khai trong game:

Game.h:



Hình . Game.h

Game.cpp:



Hình Game.cpp

**Doge.h:**

****

Hình Doge.h

**Doge.cpp:**

****

Hình Doge.cpp

Land.h:



Hình Land.h

Land.cpp:



Hình Land.cpp

Lib.h:



Hình Lib.h

Lib.cpp:



Hình Lib.cpp

Main.cpp:



Hình main.cpp

Pipe.h:



Hình pipe.h

Pipe.cpp:



Hình Pipe.cpp

Sound.h:



Hình sound.h

Sound.cpp:



Hình sound.cpp

Utils.h:



Hình Utils.h – File chứa các hằng số, đường dẫn mà nhóm định nghĩa