**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**TIỂU LUẬN CHUYÊN NGÀNH**

**ĐỀ TÀI**

**TÌM HIỂU CÔNG NGHỆ MONOGAME VÀ  
XÂY DỰNG GAME MINH HỌA**

**SVTH: MSSV**

**Nguyễn Huỳnh Phúc 17110350**

**Phạm Văn Minh Tân 17110364**

**GVHD: ThS. Nguyễn Minh Đạo**

**Tp. Hồ Chí Minh, tháng 1 năm 2021**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**TIỂU LUẬN CHUYÊN NGÀNH**

**ĐỀ TÀI**

**TÌM HIỂU CÔNG NGHỆ MONOGAME VÀ  
XÂY DỰNG GAME MINH HỌA**

**SVTH: MSSV**

**Nguyễn Huỳnh Phúc 17110350**

**Phạm Văn Minh Tân 17110364**

**GVHD: ThS. Nguyễn Minh Đạo**

**Tp. Hồ Chí Minh, tháng 1 năm 2021**

**ĐH SƯ PHẠM KỸ THUẬT TP.HCM CỘNG HÒA XHCN VIỆT NAM**

**KHOA CÔNG NGHỆ THÔNG TIN Độc lập – Tự do – Hạnh phúc**

**\*\*\*\*\*\* \*\*\*\*\*\***

**PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

Họ và tên: MSSV:

Nguyễn Huỳnh Phúc 17110350

Phạm Văn Minh Tân 17110364

Ngành: Công nghệ thông tin

Tên đề tài: TÌM HIỂU CÔNG NGHỆ MONOGAME VÀ XÂY DỰNG GAME MINH HỌA

Giảng viên hướng dẫn: ThS. Nguyễn Minh Đạo

**NHẬN XÉT**

1. Về nội dung đề tài và khối lượng thực hiện

2. Ưu điểm

3. Khuyết điểm

4. Đề nghị cho phản biện hay không?

5. Đánh giá loại:

6. Điểm:

TP. Hồ Chí Minh, ngày … tháng … năm 2021

Giảng viên hướng dẫn

(Ký & ghi rõ họ tên)

**ĐH SƯ PHẠM KỸ THUẬT TP.HCM CỘNG HÒA XHCN VIỆT NAM**

**KHOA CÔNG NGHỆ THÔNG TIN Độc lập – Tự do – Hạnh phúc**

**\*\*\*\*\*\* \*\*\*\*\*\***

**PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN**

Họ và tên: MSSV:

Nguyễn Huỳnh Phúc 17110350

Phạm Văn Minh Tân 17110364

Ngành: Công nghệ thông tin

Tên đề tài: TÌM HIỂU CÔNG NGHỆ MONOGAME VÀ XÂY DỰNG GAME MINH HỌA

Giảng viên hướng dẫn: ThS. Nguyễn Minh Đạo

**NHẬN XÉT**

1. Về nội dung đề tài và khối lượng thực hiện

2. Ưu điểm

3. Khuyết điểm

4. Đề nghị cho phản biện hay không?

5. Đánh giá loại:

6. Điểm:

TP. Hồ Chí Minh, ngày … tháng … năm 2021

Giảng viên phản biện

(Ký & ghi rõ họ tên)

# LỜI CẢM ƠN

Lời đầu tiên nhóm xin phép được gửi lời cảm ơn sâu sắc đến ban lãnh đạo Khoa Công Nghệ Thông Tin – Trường Đại Học Sư Phạm Kỹ Thuật Thành Phố Hồ Chí Minh đã tạo điều kiện cho chúng em được học tập, nắm vững các kiến thức nền tảng, nghiên cứu, trau dồi thêm kiến thức chuyên sâu nhằm phát triển bản thân trên con đường sự nghiệp.

Nhóm cũng xin bày tỏ lòng biết ơn đến quý thầy cô Khoa Công Nghệ Thông Tin – Trường Đại Học Sư Phạm Kỹ Thuật Thành Phố Hồ Chí Minh đã tận tình giúp đỡ, hướng dẫn, chỉ bảo chúng em trong suốt quá trình học tập, nghiên cứu.

Nhóm xin gửi lời cảm ơn chân thành đến thầy Nguyễn Minh Đạo đã tạo cơ hội cho chúng em được tìm hiểu, nghiên cứu và thực hiện đề tài "**Tìm hiểu công nghệ MonoGame và xây dựng game minh họa**".

Với điều kiện thời gian cũng như kinh nghiệm bản thân có hạn, nhóm không sao tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của quý thầy cô để chúng em có điều kiện bổ sung, nâng cao chất lượng sản phẩm của mình và thực việc công việc của mình tốt hơn sau này.

Nhóm xin chân thành cảm ơn!

# MỤC LỤC

[LỜI CẢM ƠN VII](#_Toc61219454)

[MỤC LỤC VIII](#_Toc61219455)

[DANH MỤC VIẾT TẮT X](#_Toc61219456)

[DANH MỤC THUẬT NGỮ XI](#_Toc61219457)

[DANH SÁCH HÌNH XII](#_Toc61219458)

[DANH SÁCH BẢNG XIII](#_Toc61219459)

[GIỚI THIỆU ĐỀ TÀI 1](#_Toc61219460)

[1. Lý do chọn đề tài 1](#_Toc61219461)

[2. Mục tiêu của đề tài 1](#_Toc61219462)

[3. Kết cấu của tiểu luận 1](#_Toc61219463)

[CHƯƠNG 1: KHẢO SÁT ĐỀ TÀI 2](#_Toc61219464)

[1.1. Game BB-TAN 2](#_Toc61219465)

[1.2. Game Bricks Breaker Quest 3](#_Toc61219466)

[1.3. Game Physics Balls 4](#_Toc61219467)

[1.4. Game Balls Bounce 2: Bricks Challenge 5](#_Toc61219468)

[1.5. Game Bricks vs Balls Breaker 6](#_Toc61219469)

[CHƯƠNG 2: MÔ HÌNH HÓA YÊU CẦU VÀ LƯU TRỮ DỮ LIỆU 7](#_Toc61219470)

[2.1. Usecase Diagram 7](#_Toc61219471)

[2.2. Tiếp nhận và thống kê nghiệp vụ 7](#_Toc61219472)

[2.3. Mô hình hóa yêu cầu 7](#_Toc61219473)

[2.4. Lưu trữ dữ liệu 7](#_Toc61219474)

[CHƯƠNG 3: THIẾT KẾ GIAO DIỆN VÀ XỬ LÝ 8](#_Toc61219475)

[CHƯƠNG 4: LẬP TRÌNH 9](#_Toc61219476)

[4.1. MonoGame 9](#_Toc61219477)

[4.2. Sơ đồ lớp 9](#_Toc61219478)

[4.3. Chức năng chính 9](#_Toc61219479)

[CHƯƠNG 5: KIỂM THỬ PHẦN MỀM 10](#_Toc61219480)

[5.1. Tổng quan thực hiện kiểm thử 10](#_Toc61219481)

[5.1.1. Giới thiệu 10](#_Toc61219482)

[5.1.2. Unit Testing 10](#_Toc61219483)

[5.1.3. Usability Testing 11](#_Toc61219484)

[5.2. Kiểm thử hộp trắng 11](#_Toc61219485)

[5.2.1. Hàm Reflect 11](#_Toc61219486)

[5.2.2. CheckCollisionWithBall function 15](#_Toc61219487)

[5.2.3. CanShoot function 20](#_Toc61219488)

[5.3. Kiểm thử hộp đen 29](#_Toc61219489)

[CHƯƠNG 6: KẾT LUẬN 30](#_Toc61219490)

[PHỤ LỤC 31](#_Toc61219491)

[TÀI LIỆU THAM KHẢO 32](#_Toc61219492)

# DANH MỤC VIẾT TẮT

# DANH MỤC THUẬT NGỮ

# DANH SÁCH HÌNH

[Hình 1.1: Game BB-TAN 2](#_Toc61220860)

[Hình 1.2: Game Bricks Breaker Quest 3](#_Toc61220861)

[Hình 1.3: Game Physics Balls 4](#_Toc61220862)

[Hình 1.4: Game Balls Bounce 2 - Bricks Challenge 5](#_Toc61220863)

[Hình 1.5: Game Bricks vs Balls Breaker 6](#_Toc61220864)

# DANH SÁCH BẢNG

# GIỚI THIỆU ĐỀ TÀI

## 1. Lý do chọn đề tài

Qua các bài viết về MonoGame, nhóm hình thành lòng hiếu kì và muốn tìm hiểu về MonoGame. Do đó nhóm mạnh dạn chọn đề tài "**Tìm hiểu công nghệ MonoGame và xây dựng game minh họa**" để tìm hiểu lập trình được một game với MonoGame.

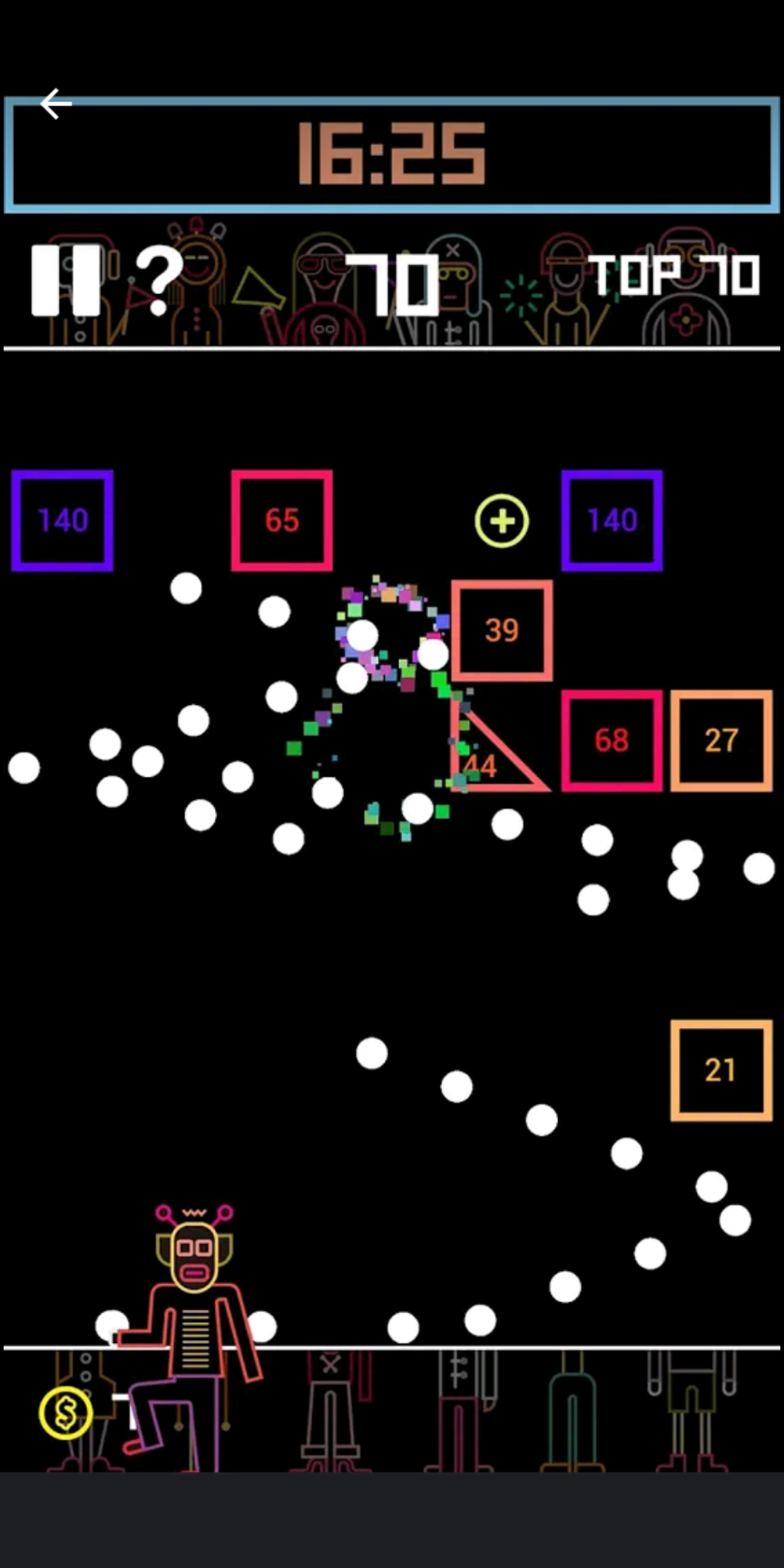
## 2. Mục tiêu của đề tài

* Xây dựng ứng dụng game với MonoGame
* Game chạy được trên đa phần các phiên bản Windows
* Game xây dựng các tính năng: chơi game, đổi bóng, xem màn hình chính, lấy bóng mới (ngẫu nhiên)
* Xây dựng giao diện dễ sử dụng
* Có hiệu ứng âm thanh, hình ảnh

## 3. Kết cấu của tiểu luận

# CHƯƠNG 1: KHẢO SÁT ĐỀ TÀI

## 1.1. Game BB-TAN



Hình 1.1: Game BB-TAN

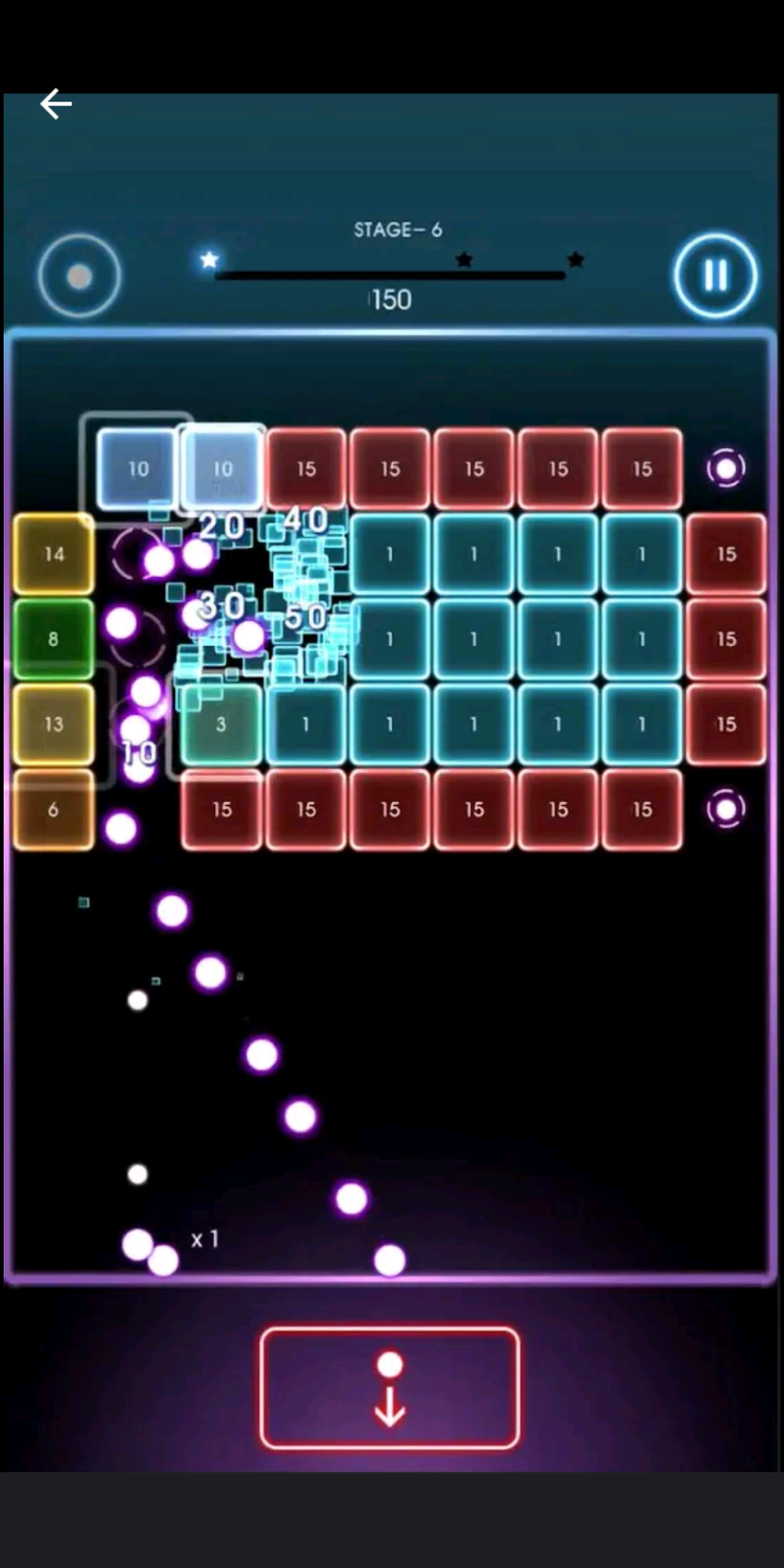
Ưu điểm:

* Màu sắc tươi sáng, đẹp mắt
* Bố cục rõ ràng hợp lí
* Hoạt ảnh mượt, hợp lí
* Hiệu ứng âm thanh bắt tai
* Dễ chơi

Nhược điểm:

* Đôi khi có độ trễ

## 1.2. Game Bricks Breaker Quest



Hình 1.2: Game Bricks Breaker Quest

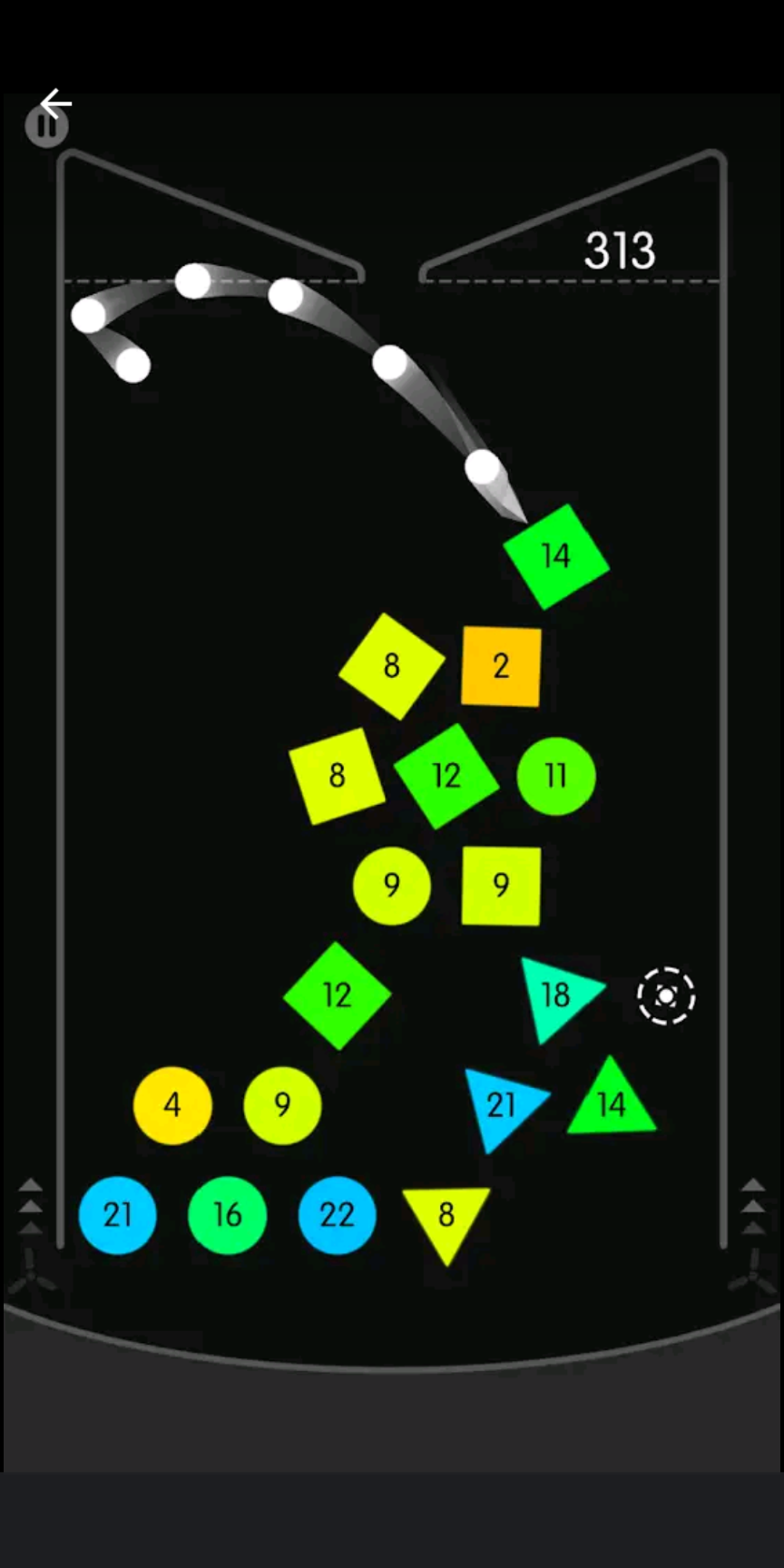
Ưu điểm:

* Màu sắc sáng tạo
* Hiệu ứng hình ảnh linh hoạt
* Hiệu ứng âm thanh sống động
* Có nhiều màn chơi

Nhược điểm:

* Độ khó tăng nhanh

## 1.3. Game Physics Balls



Hình 1.3: Game Physics Balls

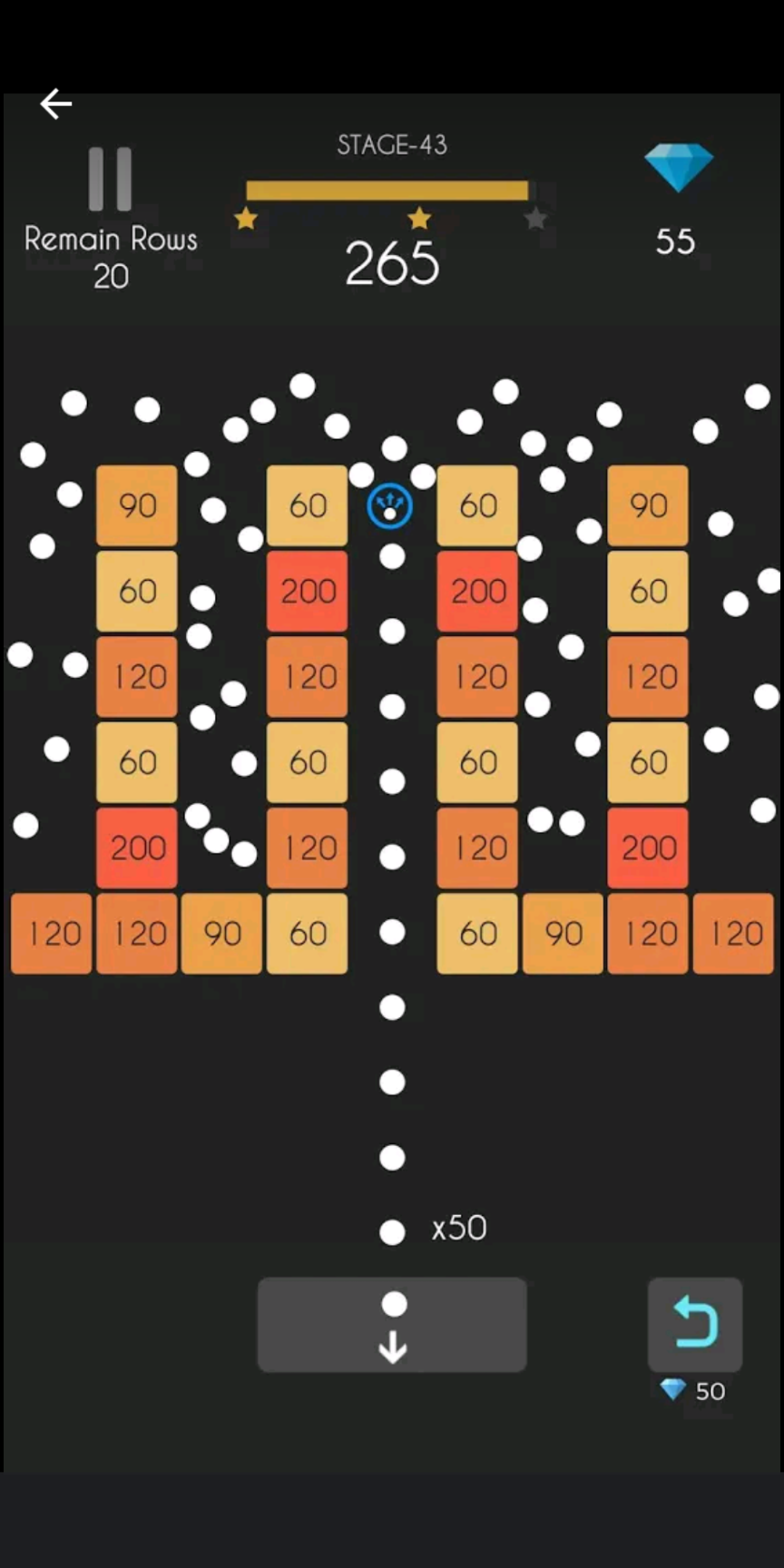
Ưu điểm:

* Lối chơi phá cách
* Đa dạng hình dạng gạch
* Hiệu ứng âm thanh nhẹ nhàng

Nhược điểm:

* Màu sắc chưa được bắt mắt
* Kén người chơi do khó dự doán hướng di chuyển

## 1.4. Game Balls Bounce 2: Bricks Challenge



Hình 1.4: Game Balls Bounce 2 - Bricks Challenge

Ưu điểm:

* Nhiều màn chơi
* Màu sắc rõ ràng, tươi sáng
* Bóng di chuyển với tốc độ cao
* Nhiều hiệu ứng hình ảnh bắt mắt

Nhược điểm:

* Nhiều thông báo

## 1.5. Game Bricks vs Balls Breaker



Hình 1.5: Game Bricks vs Balls Breaker

Ưu điểm:

* Phong cách hình vẽ độc đáo
* Màu sắc đồng nhất, tương phản tốt
* Bóng di chuyển với tốc độ cao
* Hiệu ứng độc đáo

Nhược điểm:

* Hiệu ứng âm thanh ít bắt tai

# CHƯƠNG 2: MÔ HÌNH HÓA YÊU CẦU VÀ LƯU TRỮ DỮ LIỆU

## 2.1. Usecase Diagram

## 2.2. Tiếp nhận và thống kê nghiệp vụ

## 2.3. Mô hình hóa yêu cầu

## 2.4. Lưu trữ dữ liệu

# CHƯƠNG 3: THIẾT KẾ GIAO DIỆN VÀ XỬ LÝ

# CHƯƠNG 4: LẬP TRÌNH

## 4.1. MonoGame

## 4.2. Sơ đồ lớp

## 4.3. Chức năng chính

# CHƯƠNG 5: KIỂM THỬ PHẦN MỀM

## 5.1. Tổng quan thực hiện kiểm thử

### 5.1.1. Giới thiệu

#### 5.1.1.1. Mục đích

Kiểm thử ứng dụng mô tả cách tiếp cận thử nghiệm và khuôn khổ tổng thể quá trình thử nghiệm ứng dụng BouncingGame phiên bản 1.0. Thông tin gồm:

* Chiến lược thử nghiệm: các quy tắc mà thử nghiệm sẽ dựa trên, bao gồm các yếu tố của dự án (ví dụ: ngày bắt đầu / kết thúc, mục tiêu, giả định); mô tả quy trình để thiết lập thử nghiệm hợp lệ (ví dụ: tiêu chí vào/ra, tạo các trường hợp thử nghiệm, các tác vụ cụ thể cần thực hiện, lập lịch trình, chiến lược dữ liệu).
* Chiến lược thực thi: mô tả cách thức kiểm tra sẽ được thực hiện và xử lý để xác định và báo cáo các khiếm khuyết, đồng thời sửa chữa và thực hiện các bản sửa lỗi.
* Quản lý thử nghiệm: quy trình xử lý hậu cần của thử nghiệm và tất cả các sự kiện xảy ra trong quá trình thực hiện (ví dụ: thông tin liên lạc, quy trình leo thang, rủi ro và giảm thiểu, danh sách nhóm).

#### 5.1.1.2. Tổng quan dự án

BouncingGame là một ứng dụng trò chơi mà người chơi có thể bắn bi để phá vỡ các viên gạch.

### 5.1.2. Unit Testing

Unit testing thường là các bài kiểm tra tự động do các nhà phát triển phần mềm viết và chạy để đảm bảo rằng một phần của ứng dụng (được gọi là "đơn vị") đáp ứng thiết kế của nó và hoạt động như dự định.

Trong lập trình thủ tục, một đơn vị có thể là toàn bộ mô-đun, nhưng nó thường là một chức năng hoặc thủ tục riêng lẻ. Trong lập trình hướng đối tượng, một đơn vị thường là một giao diện toàn bộ, chẳng hạn như một lớp, nhưng có thể là một phương thức riêng lẻ.

Bằng cách viết các bài kiểm tra trước tiên cho các đơn vị nhỏ nhất có thể kiểm tra, sau đó là các hành vi kết hợp giữa chúng, người ta có thể xây dựng các bài kiểm tra toàn diện cho các ứng dụng phức tạp.

Kiểm thử đơn vị được áp dụng để viết tất cả các trường hợp kiểm thử cho phần kiểm thử hộp trắng.

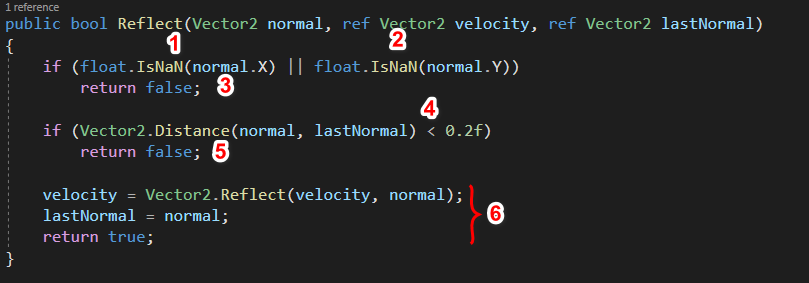
### 5.1.3. Usability Testing

Usability testing (hay kiểm tra khả năng sử dụng còn được gọi là kiểm tra trải nghiệm người dùng (UX)) là một phương pháp kiểm tra để đo lường mức độ dễ dàng và thân thiện của một ứng dụng phần mềm. Một tập hợp nhỏ người dùng cuối mục tiêu, sử dụng ứng dụng phần mềm để lộ các khiếm khuyết về khả năng sử dụng. Kiểm tra khả năng sử dụng chủ yếu tập trung vào tính dễ sử dụng của người dùng, tính linh hoạt của ứng dụng để xử lý các điều khiển và khả năng ứng dụng đáp ứng các mục tiêu của nó.

Kiểm tra khả năng sử dụng được áp dụng để viết tất cả các trường hợp kiểm thử cho phần kiểm thử hộp đen.

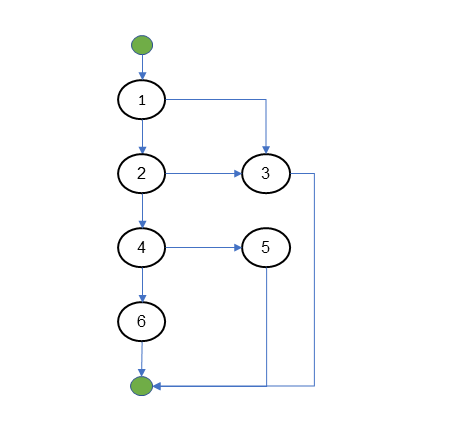
## 5.2. Kiểm thử hộp trắng

### 5.2.1. Hàm Reflect



Hình 5.1: Hàm Reflect

#### 5.2.1.1. Đồ thị dòng điều khiển



Hình 5.2: Đồ thị dòng điều khiển hàm Reflect

Có 3 node quyết định => với 3 + 1 = 4 đường thực thi tuyến tính độc lập:

1. 1->3
2. 1->2->3
3. 1->2->4->5
4. 1->2->4->6

Test cases:

1. velocity = (100, 100), lastNormal(1, 0)

Reflect((NaN,NaN), ref velocity, ref lastNormal) => Kết quả mong muốn: return false, velocity = (100, 100), lastNormal(1, 0)

2. velocity = (100, 100), lastNormal(1, 0)

Reflect((1,NaN), ref velocity, ref lastNormal) => Kết quả mong muốn: return false, velocity = (100, 100), lastNormal(1, 0)

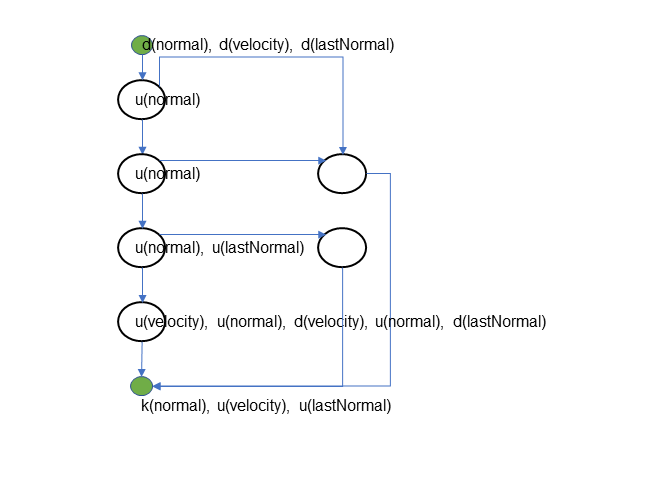
3. velocity = (100, 100), lastNormal(1, 0)

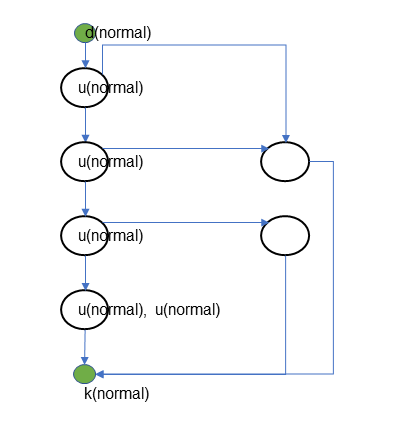
Reflect((1,0), ref velocity, ref lastNormal) => Kết quả mong muốn: return false, velocity = (100, 100), lastNormal(1, 0)

4. velocity = (100, 100), lastNormal(1, 0)

Reflect((-1,0), ref velocity, ref lastNormal) => Kết quả mong muốn: return false, velocity = (-100, 100), lastNormal(-1, 0)

#### 5.2.1.2. Đồ thị dòng dữ liệu





Kịch bản 1: ~duk

Kịch bản 2: ~duuk

Kịch bản 3: ~duuuk

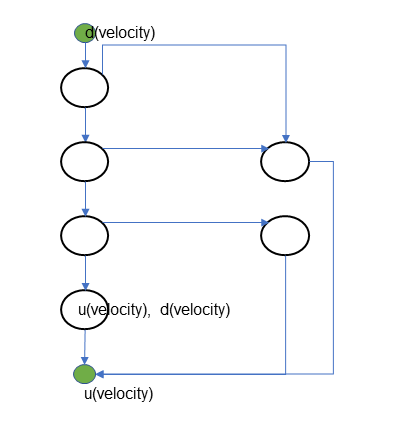
Kịch bản 4: ~duuuuk

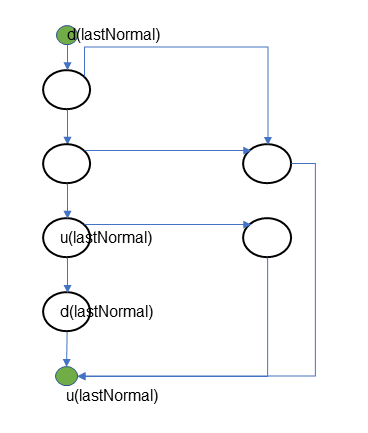
Biến tham chiếu nên không có ~

Kịch bản 1: du

Kịch bản 2: du

Kịch bản 3: du

Kịch bản 4: dudu 



Biến tham chiếu nên không có ~

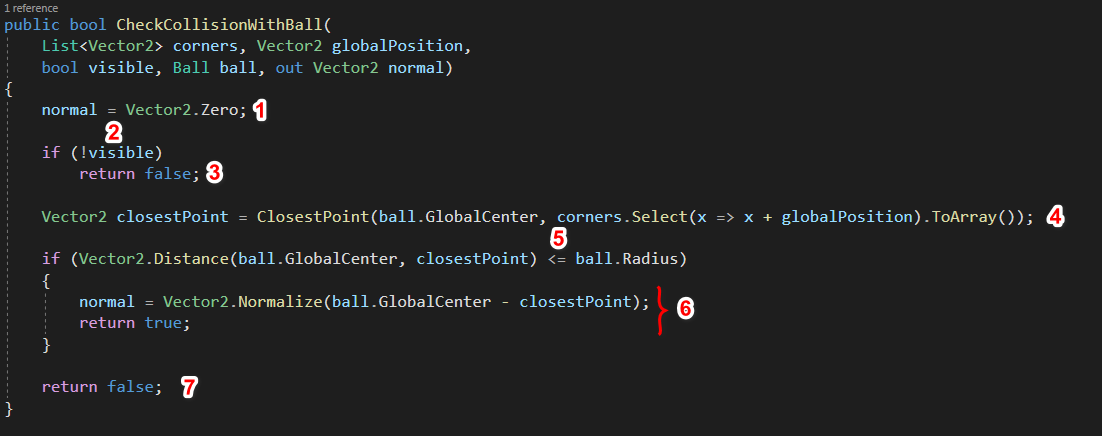
Kịch bản 1: du

Kịch bản 2: du

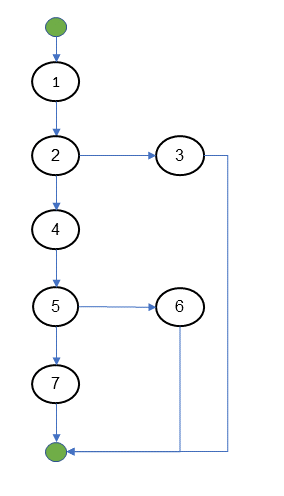
Kịch bản 3: duu

Kịch bản 4: dudu

### 5.2.2. Hàm CheckCollisionWithBall



#### 5.2.2.1. Đồ thị dòng điều khiển



Có 2 node quyết định => với 2 + 1 = 3 đường thực thi tuyến tính độc lập:

1. 1-> 2-> 3
2. 1-> 2-> 4-> 5-> 6
3. 1-> 2-> 4-> 5-> 7

Test cases:

corners = [(0,0), (0, 100), (100, 100), (100, 0)]

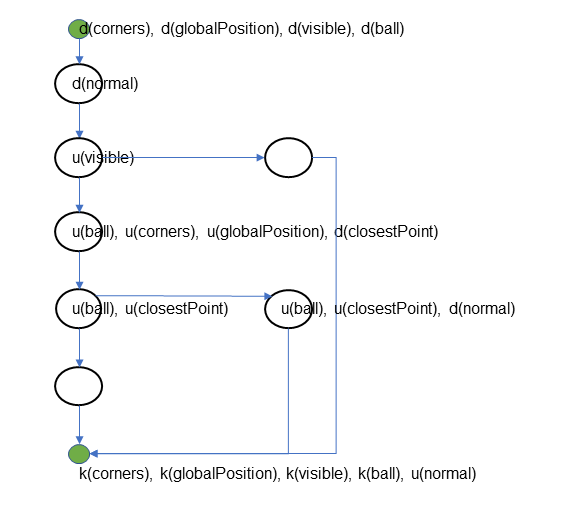
ball = {GlobalCenter = (500, 500), Radius = 20}

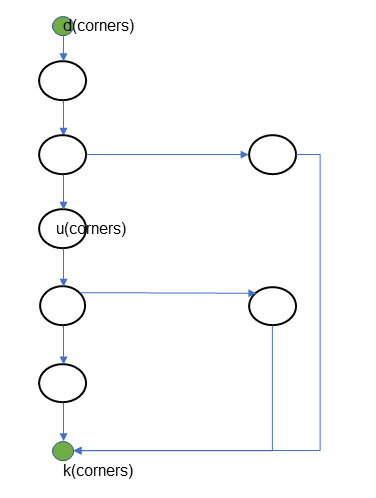
1. CheckCollisionWithBall(corners, (50,50), false, ball, out normal) => Kết quả mong muốn: return false, normal = (0,0)

2. CheckCollisionWithBall(corners, (405,450), true, ball, out normal) => Kết quả mong muốn: return true, normal = (1,0)

3. CheckCollisionWithBall(corners, (50,50), false, ball, out normal) => Kết quả mong muốn: return false, normal = (0,0)

#### 5.2.2.2. Đồ thị dòng dữ liệu



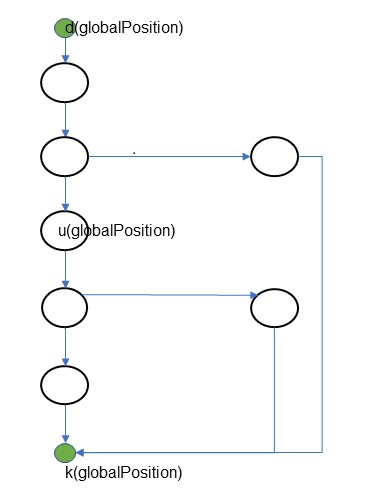


Kịch bản 1: ~dk

Kịch bản 2: ~duk

Kịch bản 3: ~duk

Kịch bản 1 có một cặp dk bất thường cần được kiểm tra lỗi.

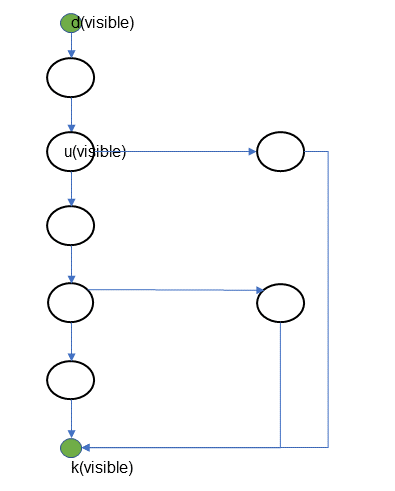


Kịch bản 1: ~dk

Kịch bản 2: ~duk

Kịch bản 3: ~duk

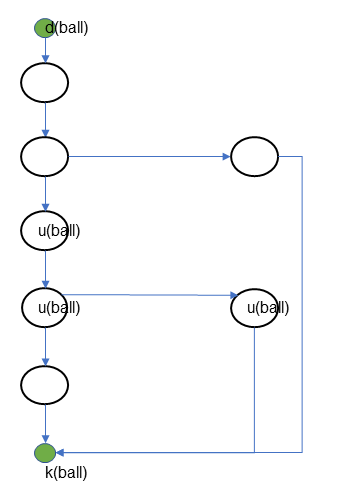
Kịch bản 1 có một cặp dk bất thường cần được kiểm tra lỗi.



Kịch bản 1: ~duk

Kịch bản 2: ~duk

Kịch bản 3: ~duk

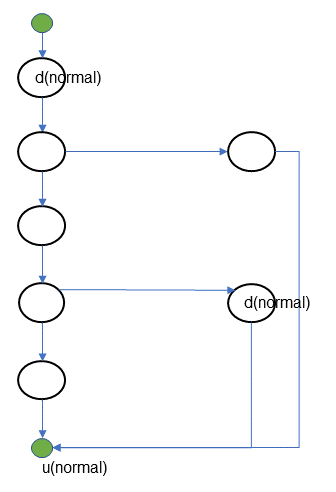


Kịch bản 1: ~dk

Kịch bản 2: ~duuuk

Kịch bản 3: ~duuk

Kịch bản 1 có một cặp dk bất thường cần được kiểm tra lỗi.



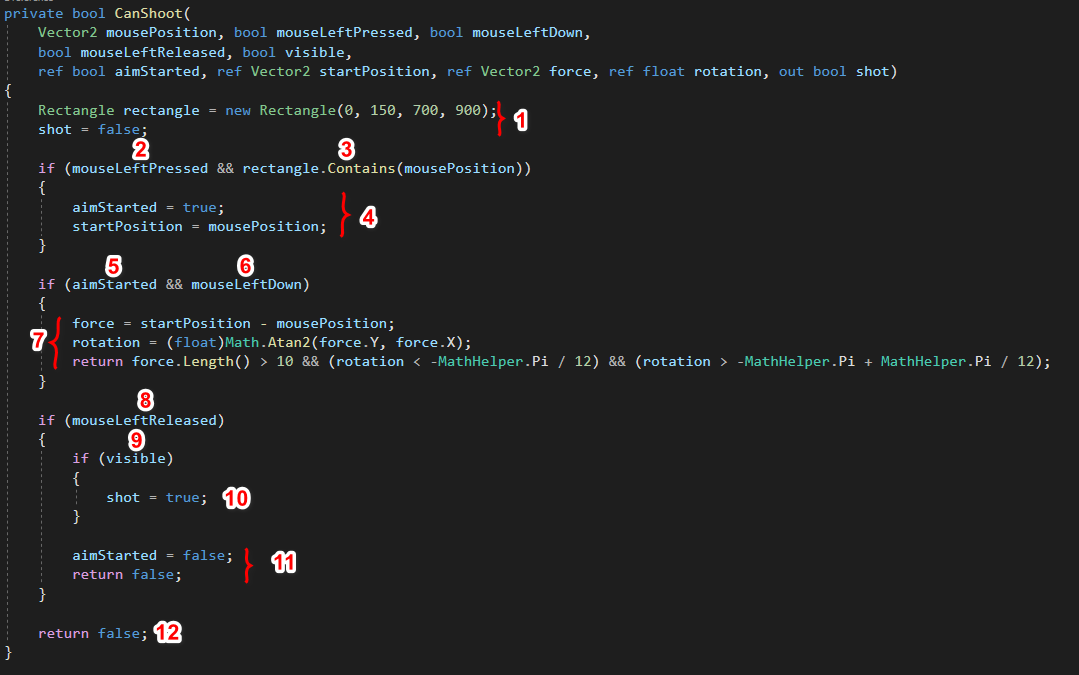
Kịch bản 1: ~du

Kịch bản 2: ~ddu

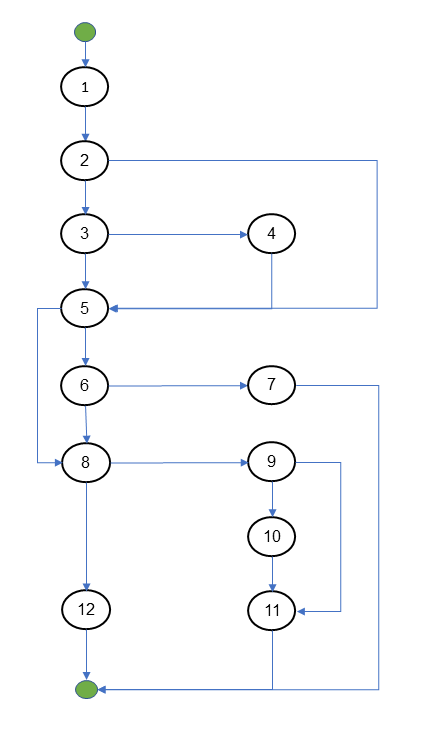
Kịch bản 3: ~du

Kịch bản 2 có một cặp dd bất thường cần được kiểm tra lỗi.

### 5.2.3. Hàm CanShoot



#### 5.2.3.1. Đồ thị dòng điều khiển



Có 6 node quyết định => với 6 + 1 = 7 đường thực thi tuyến tính độc lập:

1. 1-> 2-> 3-> 4-> 5-> 6-> 7

2. 1-> 2-> 5-> 6-> 7

3. 1-> 2-> 3-> 5-> 6-> 7

4. 1-> 2-> 5-> 8-> 12

5. 1-> 2-> 5-> 6-> 8-> 12

6. 1-> 2-> 5-> 8-> 9-> 11

7. 1-> 2-> 5-> 8-> 9-> 10-> 11

Test cases:

1. aimStarted = false, startPosition = (200,200), force(100, 100), rotation = -1.5

CanShoot((300, 300), true, true, false, true, ref aimStarted, ref startPosition, ref force, ref rotation, out shot)

Kết quả mong muốn: return false, aimStarted = true, startPosition = (300, 300), force = (0,0), rotation = 0 , shot= false

2. aimStarted = true, startPosition = (200, 200), force(100, 100), rotation = -1.5

CanShoot((300, 300), false, true, false, true, ref aimStarted, ref startPosition, ref force, ref rotation, out shot)

Kết quả mong muốn: return true, aimStarted = true, startPosition = (200, 200), force = (100,100), rotation = 0.7853981633974483 , shot= false

3. Not occur

4. aimStarted = false, startPosition = (200,200), force(100, 100), rotation = -1.5

CanShoot((300, 300), false, false, false, true, ref aimStarted, ref startPosition, ref force, ref rotation, out shot)

Kết quả mong muốn: return false, aimStarted = true, startPosition = (200, 200), force = (100,100), rotation = -1.5 , shot= false

5. Not occur

6. aimStarted = false, startPosition = (200,200), force(100, 100), rotation = -1.5

CanShoot((300, 300), false, false, true, false, ref aimStarted, ref startPosition, ref force, ref rotation, out shot)

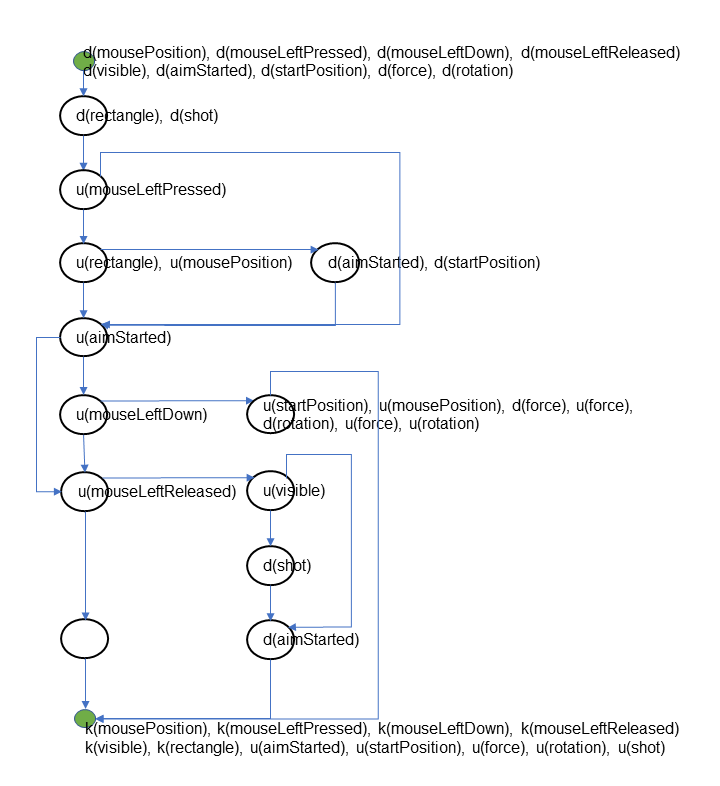
Kết quả mong muốn: return false, aimStarted = false, startPosition = (200, 200), force = (100,100), rotation = -1.5 , shot= false

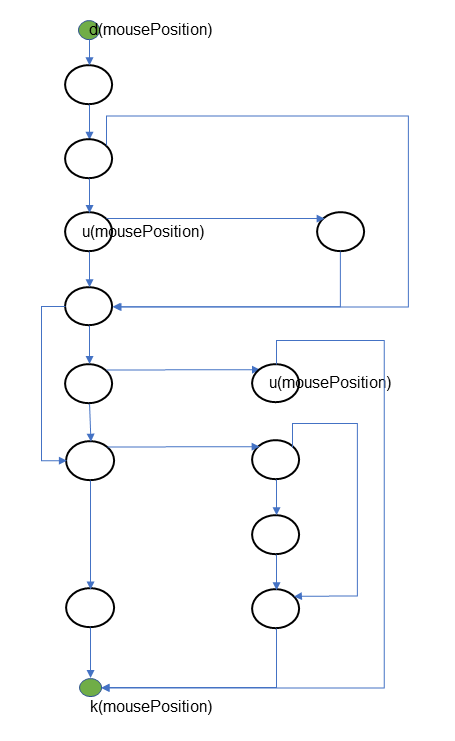
7. aimStarted = false, startPosition = (200,200), force(100, 100), rotation = -1.5

CanShoot((300, 300), false, false, true, true, ref aimStarted, ref startPosition, ref force, ref rotation, out shot)

Kết quả mong muốn: return false, aimStarted = false, startPosition = (200, 200), force = (100,100), rotation = -1.5 , shot= true

#### 5.2.3.2. Đồ thị dòng dữ liệu





Kịch bản 1: ~duuk

Kịch bản 2: ~duk

Kịch bản 3: ~duuk

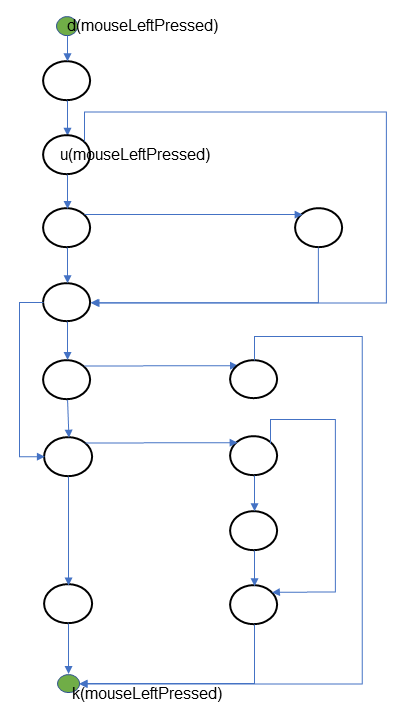
Kịch bản 4: ~dk

Kịch bản 5: ~dk

Kịch bản 6: ~dk

Kịch bản 7: ~dk

Kịch bản 4-7 có một cặp dk bất thường cần được kiểm tra lỗi.



Kịch bản 1: ~duk

Kịch bản 2: ~duk

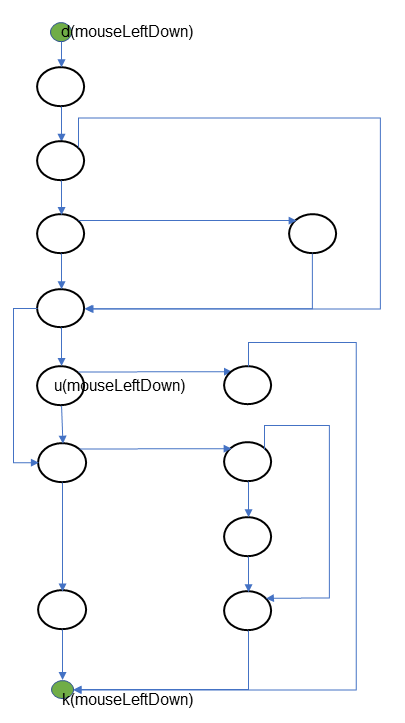
Kịch bản 3: ~duk

Kịch bản 4: ~duk

Kịch bản 5: ~duk

Kịch bản 6: ~duk

Kịch bản 7: ~duk



Kịch bản 1: ~duk

Kịch bản 2: ~duk

Kịch bản 3: ~duk

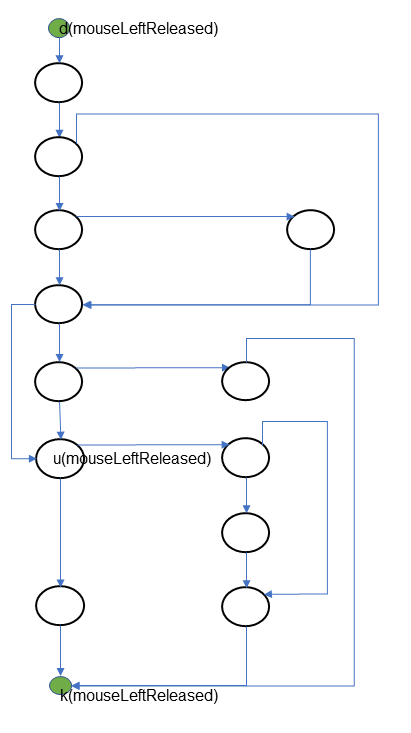
Kịch bản 4: ~dk

Kịch bản 5: ~duk

Kịch bản 6: ~dk

Kịch bản 7: ~dk

Kịch bản 4,6,7 có một cặp dk bất thường cần được kiểm tra lỗi.



Kịch bản 1: ~dk

Kịch bản 2: ~dk

Kịch bản 3: ~dk

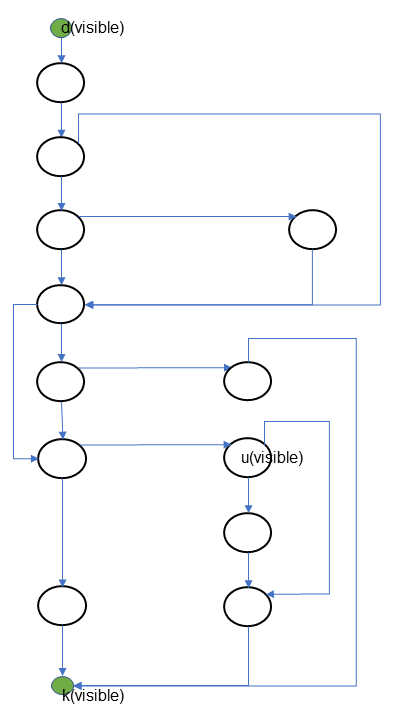
Kịch bản 4: ~duk

Kịch bản 5: ~duk

Kịch bản 6: ~duk

Kịch bản 7: ~duk

Kịch bản 1-3 có một cặp dk bất thường cần được kiểm tra lỗi.



Kịch bản 1: ~dk

Kịch bản 2: ~dk

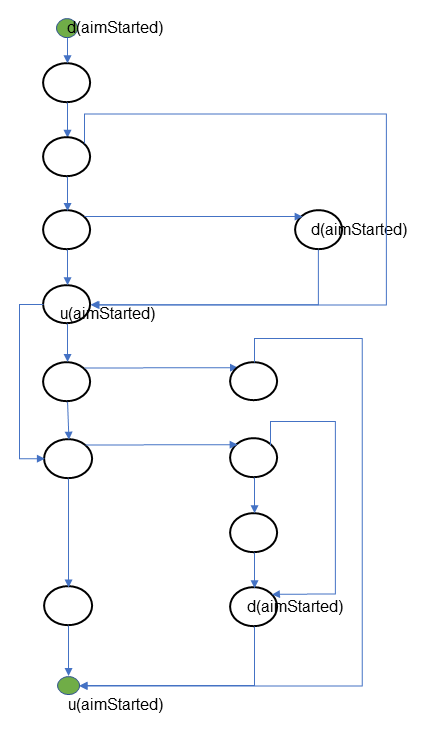
Kịch bản 3: ~dk

Kịch bản 4: ~dk

Kịch bản 5: ~dk

Kịch bản 6: ~duk

Kịch bản 7: ~duk



Biến tham chiếu nên không có ~

Kịch bản 1: dduu

Kịch bản 2: duu

Kịch bản 3: duu

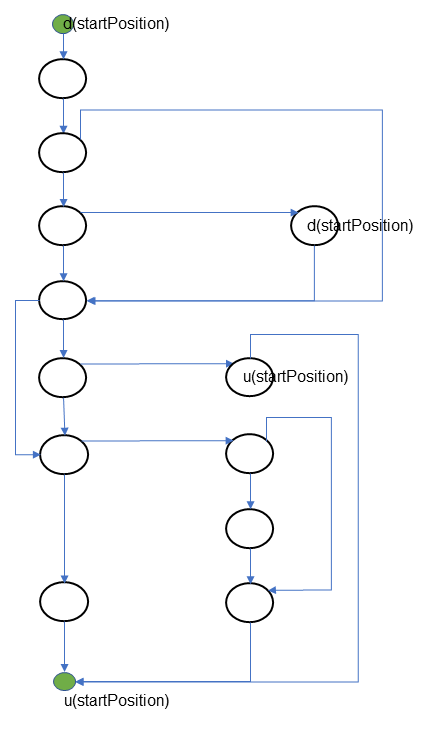
Kịch bản 4: duu

Kịch bản 5: duu

Kịch bản 6: dudu

Kịch bản 7: dudu

Kịch bản 1 có một cặp dd bất thường cần được kiểm tra lỗi.



Biến tham chiếu nên không có ~

Kịch bản 1: dduu

Kịch bản 2: duu

Kịch bản 3: duu

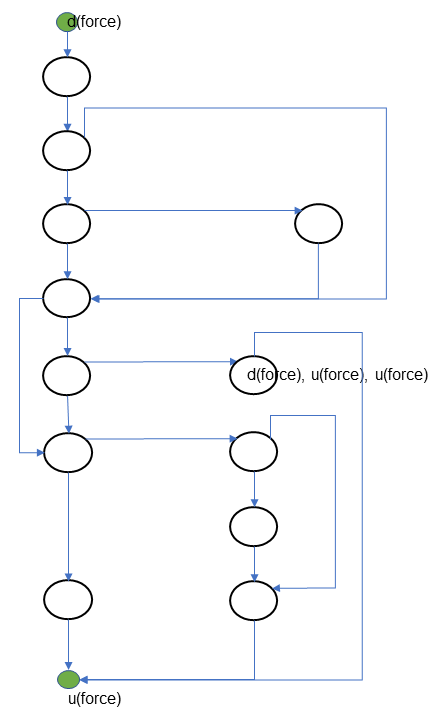
Kịch bản 4: du

Kịch bản 5: du

Kịch bản 6: du

Kịch bản 7: du

Kịch bản 1 có một cặp dd bất thường cần được kiểm tra lỗi.

Biến tham chiếu nên không có ~

Kịch bản 1: dduuu

Kịch bản 2: dduuu

Kịch bản 3: dduuu

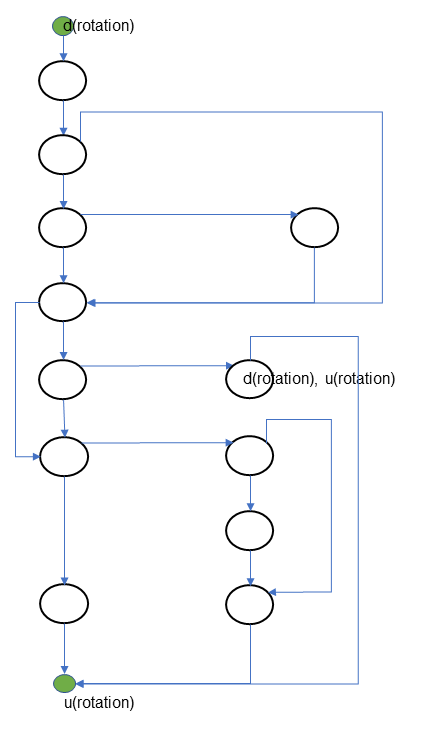
Kịch bản 4: du

Kịch bản 5: du

Kịch bản 6: du

Kịch bản 7: du

Kịch bản 1-3 có một cặp dd bất thường cần được kiểm tra lỗi.

Biến tham chiếu nên không có ~

Kịch bản 1: dduu

Kịch bản 2: dduu

Kịch bản 3: dduu

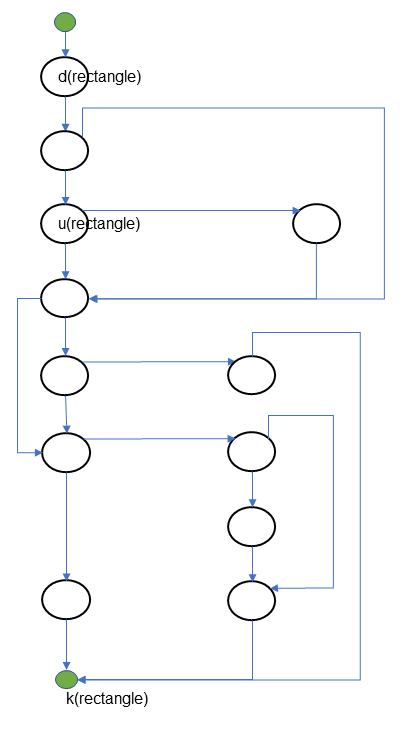
Kịch bản 4: du

Kịch bản 5: du

Kịch bản 6: du

Kịch bản 7: du

Kịch bản 1-3 có một cặp dd bất thường cần được kiểm tra lỗi.



Kịch bản 1: ~duk

Kịch bản 2: ~dk

Kịch bản 3: ~duk

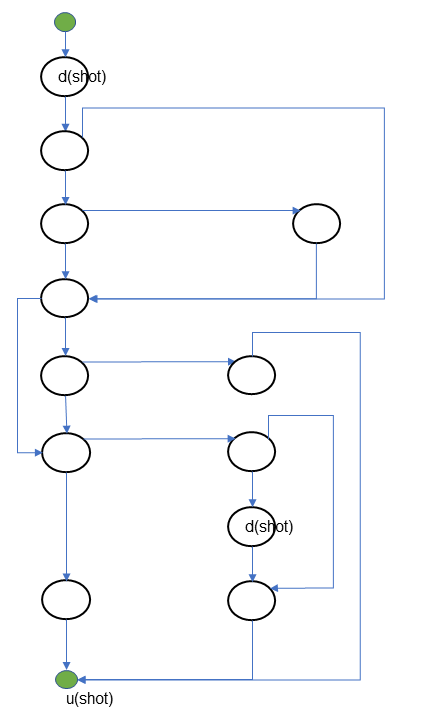
Kịch bản 4: ~dk

Kịch bản 5: ~dk

Kịch bản 6: ~dk

Kịch bản 7: ~dk

Kịch bảns 2 và 4-7 có một cặp dk bất thường cần được kiểm tra lỗi.

Kịch bản 1: ~du

Kịch bản 2: ~du

Kịch bản 3: ~du

Kịch bản 4: ~du

Kịch bản 5: ~du

Kịch bản 6: ~du

Kịch bản 7: ~ddu

Kịch bản 7 có một cặp dd bất thường cần được kiểm tra lỗi.

## 5.3. Kiểm thử hộp đen

# CHƯƠNG 6: KẾT LUẬN

# PHỤ LỤC

# TÀI LIỆU THAM KHẢO