

ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-----*



BÁO CÁO MINI-PROJECT

HỌC PHẦN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

MÃ HỌC PHẦN: IT3103

MÃ LỚP: 143577

NHÓM 27 - TOPIC 8: Ô ĂN QUAN

<i>Họ Và Tên</i>	<i>MSSV</i>
Dương Minh Phúc(TL)	20194648
Phạm Hữu Phúc	20215119
Trần Đăng Phúc	20215120
Phạm Hiếu Phương	20215121

Mục Lục

1.Phân công công việc.....	2
2.Mô tả project	2
2.1.Yêu cầu của ứng dụng trò chơi Ô Ăn Quan	2
2.2.Use case diagram	3
3.Thiết kế.....	3
3.1.General class diagram	3
3.2.Detail class diagram	4
3.3.Phân tích thiết kế	4
3.3.1.Mối quan hệ giữa các class:	4
3.3.2.Chi tiết hoạt động của các phương thức.....	5

1.Phân công công việc

Họ Và Tên	Công việc
Dương Minh Phúc	
Phạm Hữu Phúc	
Trần Đăng Phúc	
Phạm Hiếu Phương	

2.Mô tả project

2.1.Yêu cầu của ứng dụng trò chơi Ô Ăn Quan

- On the main screen:

+ Start: bắt đầu trò chơi

+ Exit: thoát trò chơi. Xác nhận lại với người dùng trước khi thoát game

+ Help: đưa ra hướng dẫn cách chơi trò chơi

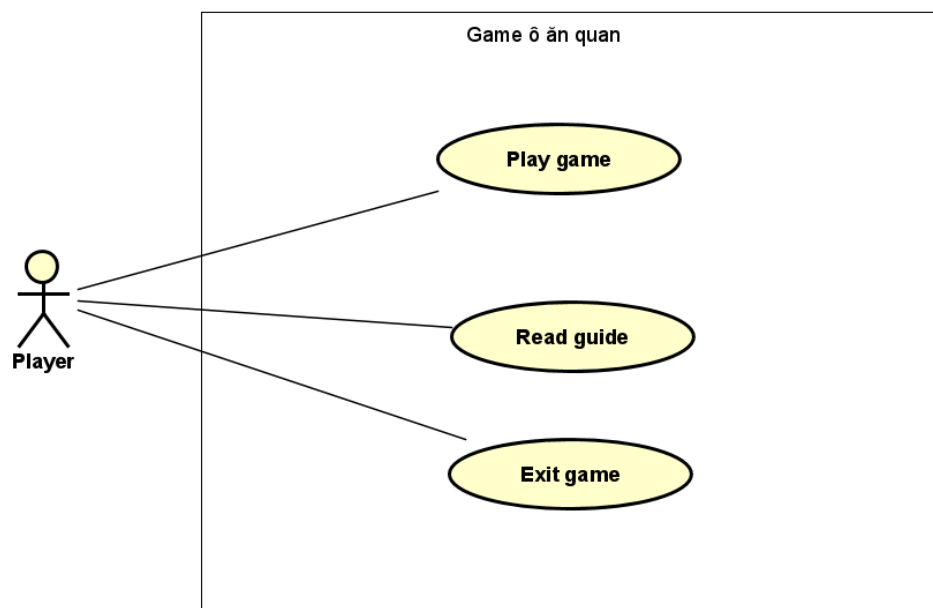
- Gameplay

+ Gameboard: bàn chơi gồm có 10 hình vuông – ô dân (chia thành 2 hàng) và 2 hình bán nguyệt – ô quan ở 2 đầu của bàn. Ban đầu, mỗi hình vuông gồm có 5 viên sỏi nhỏ– quân dân, và mỗi hình bán nguyệt có 1 viên sỏi lớn – quân quan. Mỗi viên sỏi nhỏ có giá trị 1 điểm, mỗi viên sỏi lớn có giá trị 5 điểm.

+ Trò chơi chỉ ra hiện tại là lượt chơi của người chơi nào. Mỗi lượt, người chơi sẽ chọn 1 hình vuông và 1 hướng (phải hoặc trái) để rải sỏi. Người chơi sẽ ăn điểm nếu như sau khi rải, liền sau đó là một ô trống rồi mới đến một ô có chứa quân. Hoặc nếu liền sau ô có quân đã ăn có thêm một ô trống nữa rồi đến một ô có quân thì người chơi có quyền ăn tiếp cả quân ở ô này. Điểm số được quy đổi bằng tùy vào số sỏi nhỏ và sỏi lớn ăn được.

+ Trò chơi kết thúc khi không còn sỏi ở cả 2 hình bán nguyệt. Trò chơi sẽ thông báo người chơi nào chiến thắng và số điểm của mỗi người.

2.2. Use case diagram



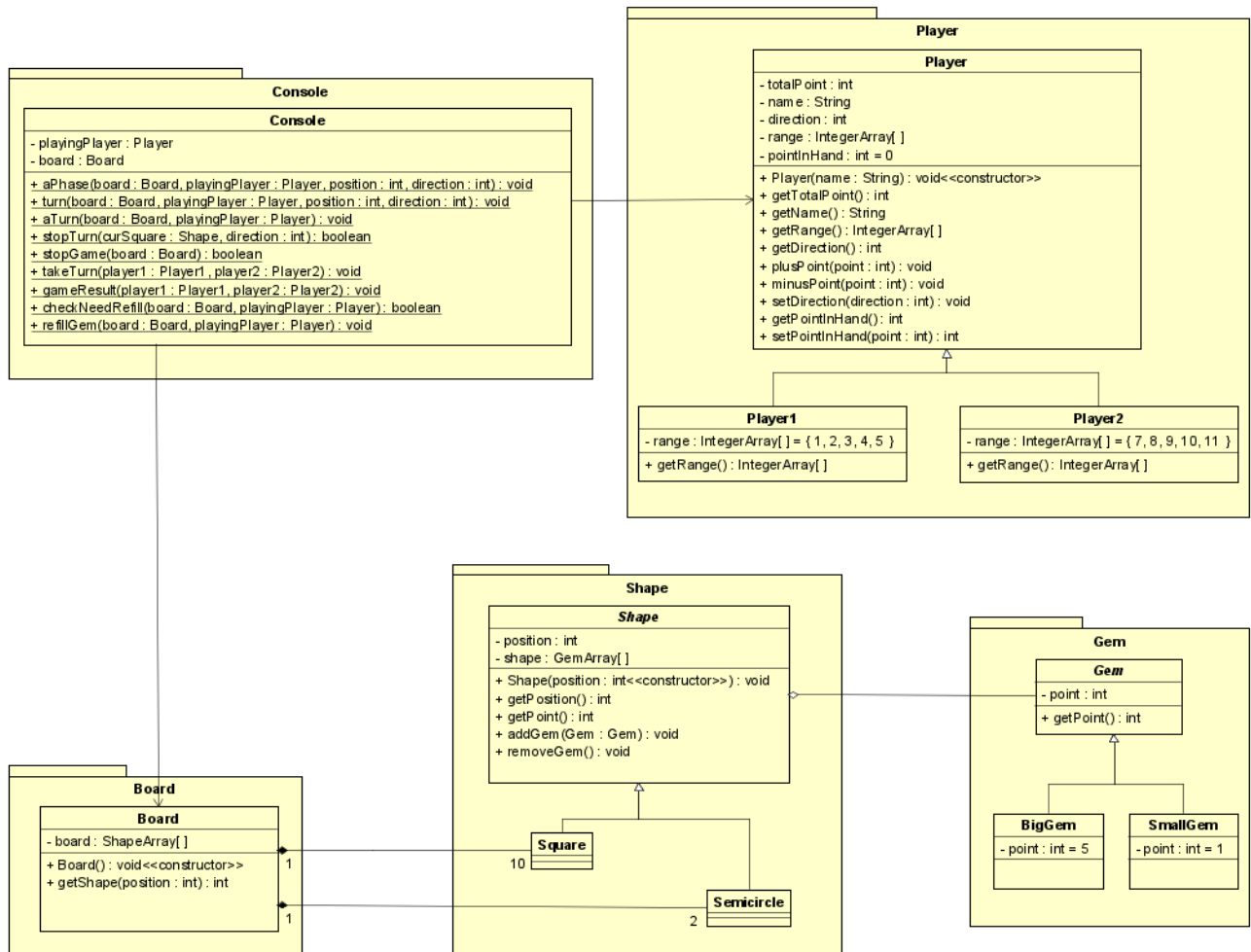
Tương tác giữa người chơi và ứng dụng:

- Player chơi trò chơi
- Player đọc hướng dẫn cách chơi
- Player thoát trò chơi

3. Thiết kế

3.1. General class diagram

3.2.Detail class diagram



3.3. Phân tích thiết kế

3.3.1. Mối quan hệ giữa các class:

- Class Gem, BigGem, SmallGem: Class Gem là 1 abstract class được sử dụng để tạo ra 2 class BigGem và SmallGem kế thừa thuộc tính point(int), và phương thức getPoint() của lớp Gem.
- Class Shape, Square, SemiCircle: Class Shape là 1 abstract class được sử dụng để tạo ra 2 class Square và SemiCircle kế thừa thuộc tính position và shape của class Shape.
- Class Player, Player1, Player2: class Player1, Player2 là 2 class con kế thừa từ class Player
- Class Gem, Shape: mối quan hệ Aggreation, Gem là 1 phần của Shape tuy nhiên 2 lớp có thể tồn tại độc lập
- Class Square , Board: mối quan hệ Composition, Square là 1 phần của Board, khi Board mất đi thì Square cũng mất và khi Board sinh ra thì Square cũng được sinh ra. 1 Board có 10 Square
- Class SemiCircle, Board: mối quan hệ Composition, SemiCircle là 1 phần của Board, khi Board mất đi thì SemiCircle cũng mất và khi Board sinh ra thì SemiCircle cũng được sinh ra. 1 Board có 2 SemiCircle.

3.3.2.Chi tiết hoạt động của các phương thức

Package	Class	Method
Board	Board - board: ShapeArray[] ->mảng lưu trữ vị trí của các hình trong bảng	+Board(): void<<constructor>> +getShape(position: int) : int ->trả về vị trí của hình
Console	Console -playingPlayer: Player -board: Board	+ aPhase (board: Board, playingPlayer: Player, position: int, direction: int): void -> + turn(board: Board, playingPlayer: Player, position: int, direction: int): void + aTurn(board: Board, playingPlayer: Player): void ->thực hiện 1 lượt chơi, người chơi chọn 1 ô trong bảng và thực hiện rải đá.Trước khi rải, kiểm tra tính hợp lệ của ô đã chọn +stopTurn(curSquare: Shape, direction: int): Boolean ->Kiểm tra kết thúc lượt, nếu số ngọc trong tay người chơi là 0 và ô tiếp theo rỗng thì sẽ kết thúc + stopGame(board: Board): Boolean ->Kiểm tra kết thúc trò chơi, nếu cả hai ô SemiCircle không có viên ngọc lớn thì sẽ kết thúc + takeTurn(player1: Player, player2: Player2): void ->Chuyển lượt chơi + gameResult(player1: Player1, player2: Player2): void ->trả về kết quả trò chơi dựa trên số điểm của 2 người chơi + checkNeedRefill (board: Board, playingPlayer: Player): Boolean ->Kiểm tra xem người chơi có cần nạp ngọc vào bảng, nếu tất cả ô trong phạm vi chọn rỗng thì trả về "true". + refillGem (board: Board, playingPlayer: Player): void ->nạp thêm ngọc vào tất cả các ô mà người chơi có thể chọn, sau đó giảm số điểm của người chơi đi 5 điểm
Gem	Class Gem -point: int	+getPoint(): int ->trả về số điểm của sỏi
	Class BigGem -point: int = 5	None
	Class SmallGem -point: int = 1	None
Player	Class Player -totalPoint: int ->tổng điểm của người chơi -name: String	+ Player(name: String): void<<constructor>> + getTotalPoint(): int ->trả về điểm của Player + getName(): String

	->tên người chơi -direction: int ->hướng di chuyển(1: ngược chiều kim đồng hồ,-1: thuận chiều) - range: Integer Array[] ->Danh sách ô mà người chơi có thể chọn - pointinHand: int = 0 ->	->trả về tên của Player + getRange(): Integer Array[] ->trả về danh sách của vị trí mà Player có thể rải đá đến + getDirection(): int ->trả về 1 nếu ngược chiều kim đồng hồ và -1 nếu cùng chiều kim đồng hồ + plusPoint(point: int): void ->cộng điểm cho player + minusPoint(point: int): void ->trừ điểm player + setDirection (direction: int): void -> + getPointinHand(): int ->trả về số điểm hiện tại của Player + setPointinHand(point: int): int ->thiết lập điểm cho Player
	Class Player1 -range: IntegerArray[] = {1,2,3,4,5} ->danh sách các vị trí người chơi 1 có thể chọn	+getRange(): IntegerArray[] ->trả về danh sách người chơi 1 có thể chọn
	Class Player2 -range:IntegerArray[] = {7,8,9,10,11} ->danh sách các vị trí người chơi2 có thể chọn	+getRange(): IntegerArray[] ->trả về danh sách người chơi 2 có thể chọn
Shape	Class Shape -position: int ->biểu diễn vị trí của hình trong bảng -shape:GemArray[] ->mảng chứa các đối tượng Gem	+ Shape(position: int<<constructor>>): void ->khởi tạo một hình với vị trí được chỉ định + getPosition(): int ->trả về vị trí của hình + getPoint(): int ->tính tổng số điểm của tất cả các viên ngọc trong hình + addGem (Gem: Gem): void ->Thêm 1 viên ngọc vào danh sách + removeGem(): void ->loại bỏ tất cả các viên ngọc khỏi danh sách của hình +toString(): ->chuyển đối tượng "Shape" thành một chuỗi.
	Class Square	
	Class SemiCircle	