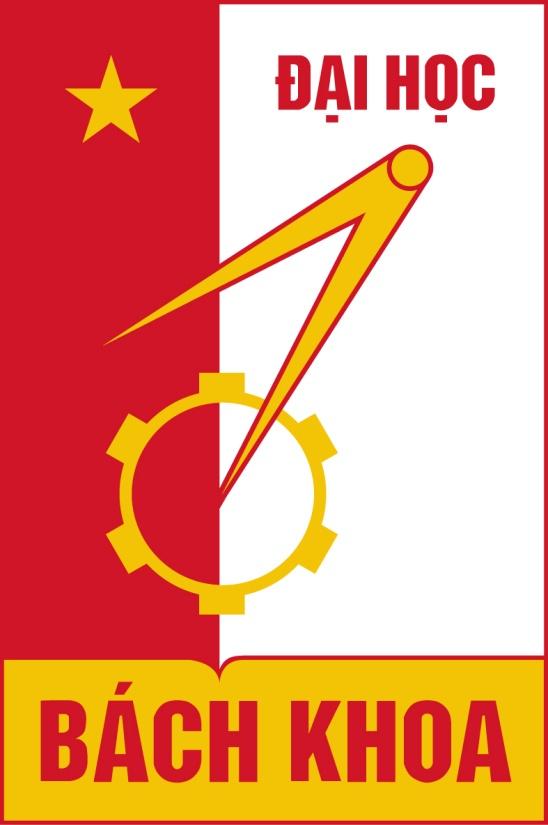
**Hanoi University of Science and Technology**

**School of Information & Communication Technology**



**Object-Oriented-Programming**

**Mini-Project Report**

Students: Pham Duc Thanh 20210795

Nguyen Ba Thiem 20214931

Doan Trong Tan 20194831

Le Thanh Thang 20194451

Instructor: Nguyễn Thị Thu Trang

Hanoi, 07/2023

**ACKNOWLEDGEMENTS**

We would like to express our deepest appreciation to Prof. Nguyen Thi Thu Trang who gave us a golden opportunity to work on this project. This work would not have been possible without your tremendous support and guidance.

## Assignment of members:

General contribution:

1. Pham Duc Thanh: 40%
2. Nguyen Ba Thiem: 40%
3. Doan Trong Tan: 10%
4. Le Thanh Thang:10%

* Work assignments :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Member** | **Model** | **View** | **Controller** | **Others related works** |
| Pham Duc Thanh | * Board(40%):board, cell. * Test for board package and player package | * Create Help.fxml * Fix Home.fxml and Play.fxml to suitable with Help.fxml | - HelpScreen-Controller  - Fix HomeContr-oller and PlayController | * Use case and several class diagram * Writing report |
| Nguyen Ba Thiem | * Board(60%): Half-circle, pickable, square * Player(100%) | * Create Home.fxml and Play.fxml | - Home-Controller  - PlayController | * Writing slide * Create project background and gameplay images |
| Doan Trong Tan | * Gem package, application |  |  |  |
| Le Thanh Thang |  | * Commit images |  |  |

* Idea:

+, Ideas for the (aesthetic) design of user interfaces:

[[Game Ô ăn quan - Dân gian - Game Vui](https://gamevui.vn/o-an-quan/game)]

+, Game rules:

[<https://hocvienboardgame.vn/huong-dan-tro-choi-o-an-quan/>]

## Description:

## Mini-project description:

* Describe in detail mini-project requirement:
* On the main screen:

+ Start: start the game. For convenient, you do not have to create different difficulties

+ Exit: exit the program. Be sure to ask users if they really want to quit the game

+ Help: Show guide for playing the game

* In the game:

+ Game board: The game board consists of 10 squares, divided into 2 rows, and 2 half- circles on the 2 ends of the board. Initially, each square has 5 small gems, and each half- circle has 1 big gem. Each small gem equals 1 point, and each big gem equals 5 points.

+ For each turn, the application must show clearly whose turn it is. A player will select a square and a direction to spread the gems. He got points when after finishing spreading, there is one empty square followed by a square with gems. The score the got for that turn is equal to the number of gems in that followed square (see the gameplay for more details about streaks)

+ The game ends when there is no gem in both half-circles. The application must notify who is the winner and the score of each player.

+ For simplicity, you do not have to build a bot to play with human

* Use case diagram: How the users interact to the software with use cases:
* Users can start a new game, choose play mode or exit the game.
* User can also see the guide of the game and view information about the game
* User can insert their name into the game.

A diagram of a person

Description automatically generated

## Design:

* A general class diagram: Class diagram may be with packages, including all classes without attributes/operations:
* A diagram of a computer program

  Description automatically generated
* Detail class diagrams for each package or several packages, with detail attributes/operations for each class:

A screenshot of a computer program

Description automatically generated

A screenshot of a cell code

Description automatically generated

A screenshot of a cell

Description automatically generated

A close-up of a computer screen

Description automatically generated

A close-up of a sign

Description automatically generated

A yellow rectangular sign with black text

Description automatically generated

A diagram of a cell

Description automatically generated

* Explanation of the design: Describe the relationships between classes, the implementations of some important methods
* Class relationship:

+ Inheritance

* Class smallGem and bigGem inherit Gem
* Multi-level inheritance: Class Halfcircle and Square inherit Cell, and Square implement interface Pickable
* Player1 and Player2 inherit Player

+ Association:

* Classes Cell and Board have aggregation relationship
* Gem is a component of Cell (one-to-many association)
* Classes Gameplay associates with board (one-to-one)
* Classes Gameplay associates with Player (one-to-many)

+ Polymorphism:

* It can be utilized in both the bigGem and smallGem as objects of the Gem class, but with distinct values assigned to their attribute
* It is also used in square and halfcircle as only square implements pickable.
* Important methods:

+ Method spreadGems() in class Player: to control gem’s movements in the game

* Firstly, check if current box has stones to move
* If current box is empty, then check if we can earn stones in the next box
* If current box is not empty, check if we can continue to move:
* If current box is mandarin box(instance of Halfcircle), end turn
* Else if current box is square box, continue to move
* If direction =1 (or equals to 0), checking the previous to see whether player can move gems then moving gems to the given direction.
* Method switchTurn() in class Player:

+ Firstly, check status of the game via the return value of checkStatus() method

+ If status = 0, i.e two mandarin boxes are both empty, check for the winner. Winner is the player has more stones in his/her side (in both square boxes and score box)

+ If setTurn= 1, i.e out of stones in current player's side and he/she have enough stones to spread out, spread stones and the game resume

+ If status = -1, i.e out of stones in current player's side but he/she does not have enough stones to spread out, this player lose, end game