

**〜〜〜〜〜〜〜〜**

***Computer Science & Engineering Department***

**INTERNATIONAL UNIVERSITY**

**Ho Chi Minh City, Vietnam**

***A GAME PROJECT’ S***

***DOCUMENTATION***

**PACMAN**

*Subject: DATA STRUCTURE AND ALGORITHMS*

*Instructor: TRAN THANH TUNG*

*Department: Computer Science and Engineering*

*ACADEMIC YEAR: SEMESTER II, 2018-2019*

**〜〜〜〜〜〜〜〜**

**〜〜〜〜〜〜〜〜**

**I. INDEX**

|  |  |
| --- | --- |
| **Part** | **Page** |
| **Index** | **2** |
| **Group Member** | **3** |
| **Introduction** | **4** |
| **Game rule** | **5** |
| **Features** | **6- 13** |
| **Game Design** | **14- 18** |
| **References** | **19** |
| **Github Link** | **19** |

**II. GROUP MEMBER**

|  |  |  |
| --- | --- | --- |
| NAME | ID | ROLE |
| TRẦN HOÀNG PHIÊN | ITITIU140 | * Assist Quynh on implementing the BFS. * Do collision detection with tiles * Do the installation part |
| DIỆP PHƯƠNG QUỲNH | ITITIU17041 | * Build and code the algorithm for the ghost to run (implementing the BFS into game) * Do the animation * Write user guide |
| TRẦN TÔ QUẾ PHƯƠNG | ITITIU17001 | * Load world into game * Design the presentation slides * Writing documentation |
| HỒ ĐẶNG PHƯƠNG NGỌC | ITITIU17058 | * Add entities into world * Do collision detection between entities (for Pacman to eat coin and ghost to eat Pacman) * Do the animation (for ghost to transform in special case) |

**III. INTRODUCTION**

Pacman is an inevitable part of everyone’s childhood; most of us cannot growing up without having experienced Pacman. Pacman is so easy and popular that its impact is expanding internationally.

The rule for Pacman is quite simple and beginners without any experiences on games can also take part in. The player (Pacman in the game for specification) is provided a map (or a maze as well) with full of coins for Pacman to collect. In the maze there may also be some obstacles such as rocks or wall that Pacman cannot pass through so that Pacman must decides the best effective way for him to get all the coins without getting stuck inside the maze or being eaten by some ghosts (Pacman’ s enemies.) About the ghost, they are provided some priorities. They can eat Pacman within just a second of intersection (or you can set Pacman’s lives depending on your taste and the level provided by the game as well.)Some versions there can be in a glance that Pacman can eat the ghosts as well: in the case that Pacman have collected some special food( or weapons) such as diamonds, wood, and so on that he can have small amount of time ( for about five to ten seconds) to weaken the ghosts( in this period the ghosts turn into blue or yellow) and Pacman can wipe out the ghosts within this period. Out of the given time, the ghosts go back to their normal state and continue chasing Pacman and trying to kill Pacman.

It comes as a piece of cake to master this game; however, it is not that simple. In the game we have built, we have implementing an algorithm that might make the game more interesting and raise the complexity a little bit higher that is slightly out of the game’ s comfort zone. We use JAVA as the main language of this built in game and implement the knowledge of **Bread First Search (BFS)** to have this game completed and brought into practice, which will be explained profoundly in the adjacent part.

**IV. THE GAME RULE**

The rules are quite similar to the former versions of Pacman: Pacman will be set inside a maze (which is called a world in our game built); in the world there are walls that separate the world into many segments and link to each other through some small holes or narrow aisles; Pacman must pass through those paths to collect the coins appearing on the screen so that he can earns as many points as possible; each coins values 10 points. There are some special diamonds in the maze which value 100 points allowing the player to eat the ghost- which gives 100 bonus points to the player score if he has collected this diamond (within 3 seconds). The player will be noticed of the 3 seconds through the color of the ghost (the ghost change to blue within this period). After this period, the ghosts turn back to their initial state. The player wins when the points reaches 1500.

The player should not pump into any ghost in any circumstances; any contact with any ghost is enough for him to be killed immediately. He just has THREE LIVES to lose. Running out of the three lives, the player will end up with GAME OVER.

The attractive feature of this game is placed on the ghosts. The ghosts are set animation to appear and disappear one after another and repeatedly. They just appear completely in blue condition in the period of the first 3 seconds the player earns the diamond. All of the ghost are implemented the random move around and inside the maze; especially there are three of them are implemented the BFS to find the shortest path to Pacman’s position (to chase Pacman).

**V. FEATURES**

**A. GHOSTS MOVE**

There are four ghosts in the game:

* Blinky (red): move randomly
* Inky (blue): chase and move randomly
* Pinky (pink): chase and move randomly
* Clyde (yellow): chase and move randomly

**1. Random move:**

The ghosts have four selection of directions to select : UP, DOWN, LEFT and RIGHT.

For the UP direction:

A screenshot of a social media post

Description automatically generated

In the case of UP direction, we check for up, left, right and down next move respectively if the next move will get collision with the wall or not; if there is any collision with wall for the considering move, the program will change to next considerable move in the mentioned order above.

We will do it the same way for DOWN, LEFT AND RIGHT directions.

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

**2. Move implementing BREAD FIRST SEARCH (BFS)**

**A screenshot of a computer

Description automatically generated**

**IDEA**

The idea of this implementation is for the ghost to find the shortest path to Pacman position (this is for the ghost to chase Pacman); within the steps to Pacman is under or equal to a set constant, the ghost will be implemented this algorithm to chase Pacman; otherwise, it will run randomly as above random move.

**EXPLANATION**

*EXAMPLE*

From the picture above, it is shown that there are three ways to Pacman position for the

ghost (named inky) to enter and kill Pacman: the first, second and third take seven, five and thirteen steps respectively to the position next to Pacman that the ghost have the possibility to catch Pacman. It is set to automatically choose the steps of five for inky only (the thirteen- step way is omitted first since it is more than seven steps, the seven one is larger than five one so five would be the chosen pathway.)

*EXPLANATION*

Consider the path from the Ghost to Pacman as a graph.

* Every blacktile is a vertex.
* Set all vertices visited=0.
* Because ghost move randomly, we have to tick new position of the ghost to update

the path so we add it into a queue.

* Tick position of the ghost visited[x][y]=1.
* Into the while loop:
  + Set Coordinate s= queue.poll();
  + If s.y>0 and upper tile = blacktile and visited of the tile = 0 , add its coordinate into queue and set its visited = visited[s.x][s.y]+1.
  + Similar for 3 other directions.
* findPath() method check position of pacman.
* Run BFS for the ghost to check the distance in 4 direction up,down,right,left of Pacman: Inky check for number 7, clyde check for number 5, pinky check for number 10.
* If pacman run into tiles which had been marked, check 4 direction up,down, right,left of pacman. Which direction has the number of tile position of pacman -1 , the ghost chase pacman by that path.
* Else it moves randomly as blinky.

B. PACMAN MOVE

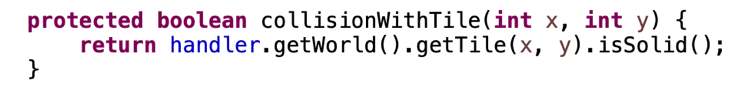
Pacman moves depend on the input of the player

* UP arrow key for moving up
* DOWN arrow key for moving down
* LEFT arrow key for moving left
* RIGHT arrow key for moving right

The player is just allowed to move in the black areas, which means that there is no wall so that Pacman can pass (Pacman cannot go through walls).

C. CREATURE’ S MOVE WITHOUT COLLIDE WITH WALL

It is logic that Pacman and Ghosts cannot move through walls. In the world there are a lot of walls and Pacman or Ghosts can just only move on the black path, which means there is no walls in the path.

This has been conducted thanks to a check collision with tile method inside the Creature class. Creature here comprises of Player (Pacman) and Ghosts. 

At first when we design the world, it has been decided that tile walls return true for isSolid() and the blackTiles returns false. For that reason, the move can just be computed if the tile at the position to move is not wall.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**D. PACMAN EATS COINS AND DIAMONDS/ GHOSTS EAT PACMAN**

**IDEA**

Basically, in this we check for collision between entities: Pacman and coins (or diamonds), and Ghost and Pacman. If there is any collision being noticed, the lives of the entity being eaten will decrease by ; if the live is zero means that that entity has die and is cleared in the Entity Manager list, which also means that it will disappear on the game board.

**EXPLANATION**

**A close up of a sign

Description automatically generatedA close up of a sign

Description automatically generated**

*EXAMPLE*

From the above example, we can see that at the first picture the coins inside the column path that Pacman is standing are still exist. After that in the second picture they disappear on that path since Pacman has gone through and eat all the coins in the area.

*EXPLANATION*

Each entity has its own boundary box, which is a rectangle smaller than the entity real size (for easy move between tiles since we assigned entity real size equals to the tiles’ s size) formed together with the entity to check if there is any intersection of that with another boundary box of another entity, which means that there is a collision between two entities.

A screenshot of a social media post

Description automatically generated

* Method getCollisionBounds() to get the coordinates, height and width of that box of each entity.
* Method beEaten(int amt) will take in an amount of health (which is entity’s lives) to be decreases by. If health is zero, the entity dies.
* Method die() is called when the entity has died before being wiped out in the entity list.

**IN THE PLAYER CLASS:**

*In the case of Pacman eating Coins (or Diamonds)*

When the boundary box of Pacman intersects with the box of a coin (or a diamond), that entity will decrease its health by 1; since coins or diamonds are just static entities and do not have any lives so after being eaten, it call the method die() right away generating signal for the entity manager to clear out that coin or diamond in the entity list.

The score will be updated by 10 if there is a coin being eaten and 100 in the case of a diamond.

*In the case of Ghosts eating Pacman*

When the boundary box of Pacman intersects with the box of a Ghost, Pacman will loses one live and being set at another position by calling newPos() method since it can be eaten again if being re- generated at the same position that it has been killed. Losing all three lives will bring the player to game over state.

***SPECIAL ANIMATION***

There is a special idea for the game design: we want that Pacman can also have the ability to fight back the ghost some time; so special diamonds in the world comes as a great tools to fulfill this idea. When Pacman collects one diamonds, which means that there is an intersection between Pacman collision bound rectangle and that of the diamond, the ghost is set animation to become blue ( a blue image of frightened ghost is rendered instead of its normal frame) within 3 seconds:

The three second is set by using Timer of javax. swing. Timer. After that period, the ghost turns back to normal. There is a boolean variable in the class Game called frighten ( default is false). If diamond is eaten, frighten becomes true; this is the condition for Pacman to eat back the ghost and earns 100 points for each ghost being eaten. After 3 seconds, frighten is set back to false; now that the condition is no longer satisfied, Pacman cannot eat the ghost anymore.

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

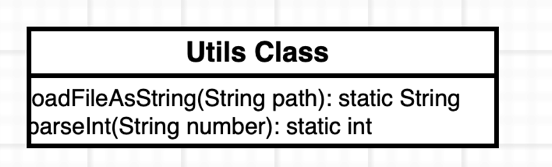
A screenshot of a cell phone

Description automatically generated

**VI. GAME DESIGN**

A screenshot of a cell phone

Description automatically generated



A screenshot of text

Description automatically generated

A screenshot of a computer

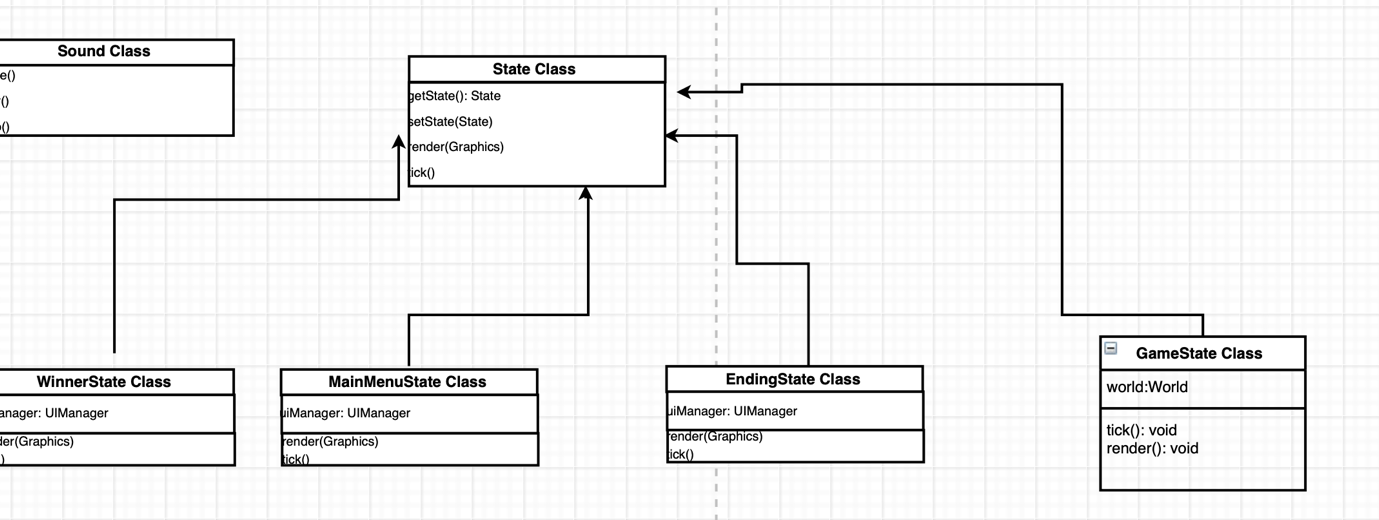
Description automatically generated

* The Tile Class is the super class of BlackTile class and Wall class, which are drawn on the game board.
* The Utils Class enables the loading world fron text file into the game. In the text file, the first row hold the width and height of the game board, the second row consists of the initial position of the player, which we won’t use anymore in the real game because we have changed into another option and haven’t fixed that in the text file, but it doesn’t affect the game flow; and the below 0 and 1 is a way to mark our game board with 0 is blackTile and 1 is wall.
* The World Class holds attributes and methods to create and manage our game; our main world organizes two important components: the Entity (which is manages by the entity manager from the EntityManager Class) and the Tiles.

A close up of text on a white background

Description automatically generatedWe categorizes our entities into two kinds: static entity (StaticEntity Class) and Entity (Creature Class), both inherits from Entity Class. In the StaticEntity, there are two more children classes called Diamond Class and Coin Class representing the coins and diamonds placed inside the world as food for Pacman. The Creature Class has four more children classes including Player Class representing Pacman, Ghost1, Ghost2,Ghost3, Ghost4 Class representing the four different ghosts in the game since four ghosts have slightly different type of movement.

Besides, we have EntityManager Class to manage the entities in the world; this class is just a managing- class holds the list of all entities in the world.



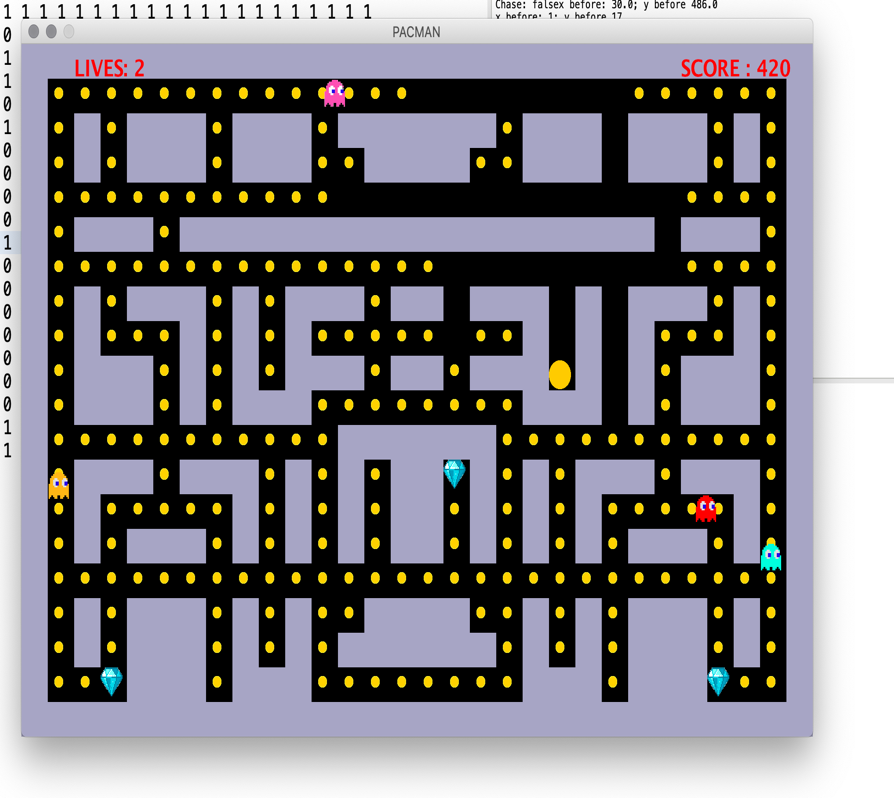
We divide our game into states for easier organization because we can change between one and another stage; the flow game would be from main menu to the game state, and then if win to winner state, else to ending stage; from both we can go back to menu state to have the game started again. The four classes are specification of the class State Class.

A screenshot of a cell phone

Description automatically generated

These are classes assist in creating and organizing UI experience in the States. For example, the button pressing leading to another state and the images in each state.

A picture containing object

Description automatically generatedA picture containing object

Description automatically generated



The four states of the game in order: menu state, game state, winning state and ending state.

A screenshot of text

Description automatically generatedA screenshot of a cell phone

Description automatically generated

These classes holds method and attribute to take in the input from the keyboard and from mouse ( for clicking button or moving Pacman by up, down, left or right arrow key)

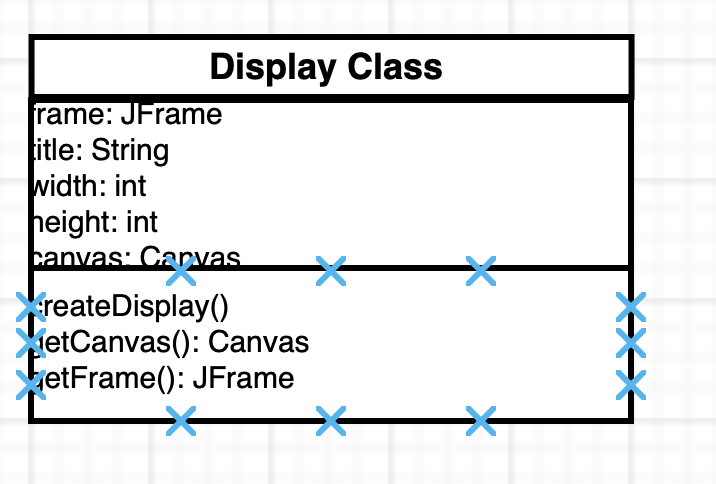
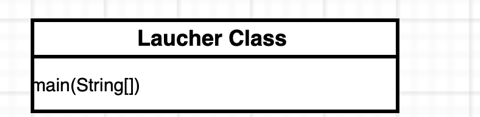
A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated  
These classes help controlling the graphic inside the game: Assets Class will holds all the pictures used inside the game, ImageLoader helps to load image and Animation will help to do the animation.



Launcher class holds the main and the whole game will run due to the called method start() inside this main.

VII. REFERENCES

1. The You Tube series guiding to create simple 2D game

<https://www.youtube.com/playlist?list=PLah6faXAgguMnTBs3JnEJY0shAc18XYQZ>

2.

<https://en.wikipedia.org/wiki/Pac-Man>

3.

<https://stackoverflow.com/questions/29141501/how-to-implement-bfs-algorithm-for-pacman>

4.

<https://www.hackerrank.com/challenges/pacman-bfs>

VIII. GITHUB LINK

<https://github.com/ititiu14078/PACMAN_OFFICIAL>