CA project submission for Software Development Fundamentals (semester 2)

Role Playing Console Game

Terran VS Zerg

Balazs Barcza - SB18002

# Section one: The project

Name of the Game : Terran VS Zerg

The game is base on the famous franchise StarCraft. All of the characters and story background is protected by copyright. This game is only study for purposes.



Terran VS Zerg is role play text base console game.

Definition:

The role-playing games relied heavily on either group consensus or the judgement of a single player. RPG system also affects the game environment, which can take any of several forms. Generic role-playing game systems, such as Basic Role-Playing, GURPS, and Fate, are not tied to a specific storytelling genre or campaign setting and can be used as a framework to play many different types of RPG. Many role-playing game systems involve the generation of random numbers by which success or failure of an action is determined. This can be done using dice (probably the most common method) or cards.

Information about my game:

Terran VS Zerg is text base console game. I have used Java program language. My game includes lots of class and it focuses on the [abstraction and inheritance and polymorphism](https://moodle.cct.ie/mod/resource/view.php?id=37690).

“The main difference is polymorphism is a specific result of inheritance. Polymorphism is where the method to be invoked is determined at runtime based on the type of the object. This is a situation that results when you have one class inheriting from another and overriding a particular method.”

The story is about two races. One is the Terran and other one is the Zerg They are fighting over the Earth. The player can choose between the two race.

Terran:

The terrans (or humans) are a young species with psionic potential. The terrans of the Koprulu sector descend from the survivors of a disastrous 23rd century colonization mission from Earth. Compared to the protoss and zerg, the terrans are highly factionalized and endure frequent wars amongst themselves in addition to the more recent conflicts with their alien neighbors. Nevertheless, terrans stand as one of the three dominant species of the galaxy.



Zerg:

The Zerg are a race entirely unlike the Terrans. They are composed of many different species integrated into the Swarm via Zerg infestation. These creatures are rapidly and selectively evolved into deadly and efficient killers to further the driving Zerg imperative of achieving absolute domination. The Zerg make no use of technology to create their weapons, armor, or starships. Instead, these functions are efficiently fulfilled through biological adaptation and planned mutation of the Zerg strains. Even Zerg buildings are in fact specialized organs within the living, growing organism of a Zerg nest.



After picking race , the player can choose main hero. There are three different characters per race. All of the characters are randomly created so every time restarts the game and it will be different.

Terran

* Marine m1;
* Reaper r1;
* Ghost g1;

Zerg

* Hydralisk h1;
* Zergling z1;
* Queen q1;

Main Boss

* Kerrigan k1; // Zerg main boss
* TychusFindlay t1; // Terran main boss

All units must have :

* name
* health
* skill
* toughness
* level
* inventory

After choosing player can move around on the map 3x3. Every location has a unique story. Player needs to fight three enemies and needs to have 3 victories to be able to fight with the main boss.

In the main fight, the gamer will get extra two characters. This will be team fight so player can change characters (if necessary). Player has to be careful because any loos of characters (dead) means end of the game.

The game has a combat system. I have used the dices to decide who will make attack or defence etc.

Combat system:

* attack dice base
* defend dice base
* magic paper rock scissors simple game

“Rock–paper–scissors is a hand game usually played between two people, in which each player simultaneously forms one of three shapes with an outstretched hand. These shapes are "rock", "paper", and "scissors".”

* change characters !! This optional only available on the main boss fight !!
* run a way exit from the game

Goal of the game:

If all player units are defeated the player loses (end game).

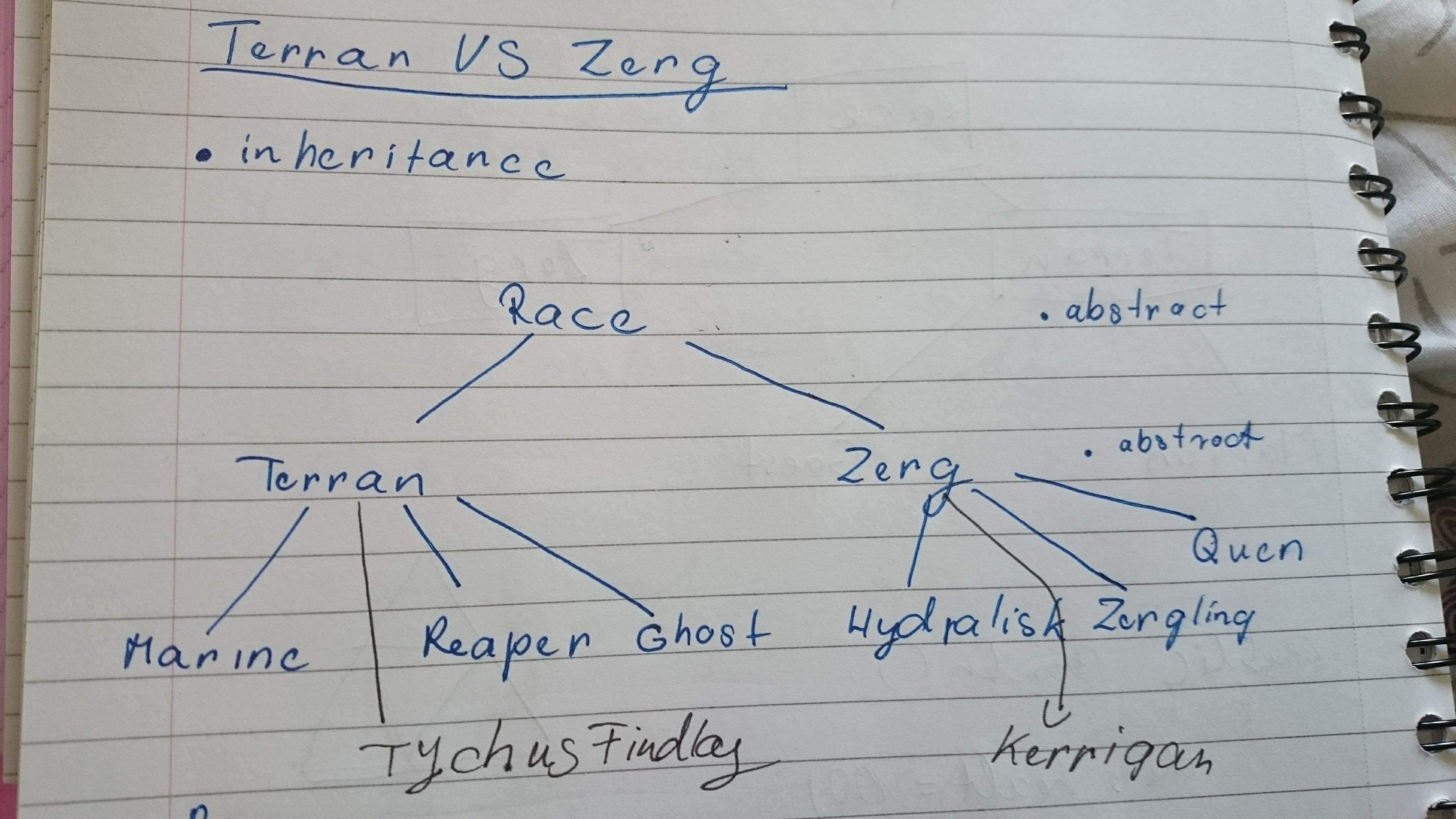
If the enemy boss is defeated, player wins (end game).

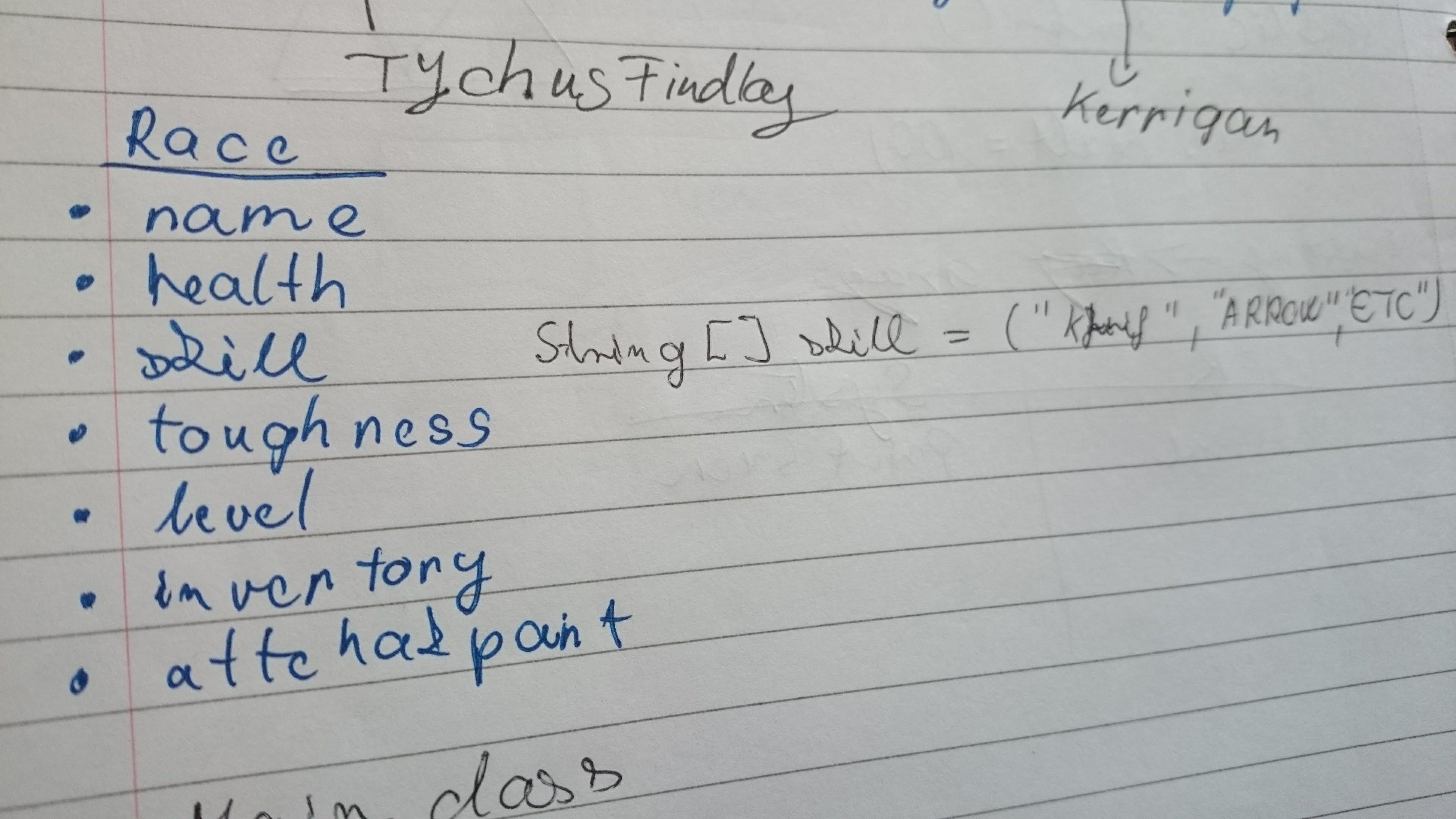
# Section 2: Analysis

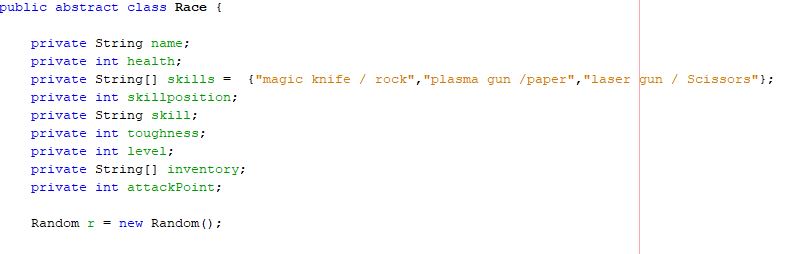
|  |  |  |
| --- | --- | --- |
| **User** | **System** | **Code / objects required** |
| Welcome message | Print to screen | Print statement |
| Enter your name | Read from keyboard / save value | Player object – set name field |
| Menu | Print to screen | Print menu |
| If the user choose 1 : info | Print to screen | Print info page |
| if the user choose 2 : start game | Print to screen | Start the game loop |
| If the user choose 3 : Exit | Print to screen | System.out(0); |
| Please select your race | Print to screen - options | Menu object / set choice in Player object |
| Please select your hero | Print to screen - options | Menu object / set choice in Player object |
|  | System create objects | Marine m1 = new Marine(); ETC |
| Please move the character | Print out the map and direction | Call map class |
| Character moved | Print out the story | Call the story class |
| Character meet with enemy | Start the fight method | Fight.fightWithEnemy(); |
| Please chose fight optional | Print attack / defend / exit | Call method  attack();  defend(); |
| 1: attack | Print out the attack result | Dice setup healthpoints |
| 2: defend | Print out the defend result | Dice setup healthpoints |
| 3: Exit | Exit from the game | System.out(0); |
| The gamer win | Print out hero infomations | fight.a[player.getHeroCharacter()].info(); |
| Victory | Increase the win number | win++; |
| Lost the fight  optional exit game or restart | Exit from the loop | (a[heroCharacterF].getHealth()<0){System.out.println("GAME OVER");} |
| Continue move the character  find other enemy | Print out the map and direction  make fight | Map class  story class  fight class |
| 3 victory get message | Find the main Boss | Map class  story class |
| Fighting with MAIN BOSS | Get extra unit  print out the fight method | fight.fightWithEnemy(); |
| 1: attack | Print out the attack result | Dice setup healthpoints |
| 2: defend | Print out the defend result | Dice setup healthpoints |
| 3: change characters | Print out units name | charactercChangeMethod(); |
| 3: Exit | Exit from the game | System.out(0); |
| The gamer won the game  restart the game or Exit | Print out VICTORY | switch (menunumber) |
| Victory | Increase the win number | win++; |

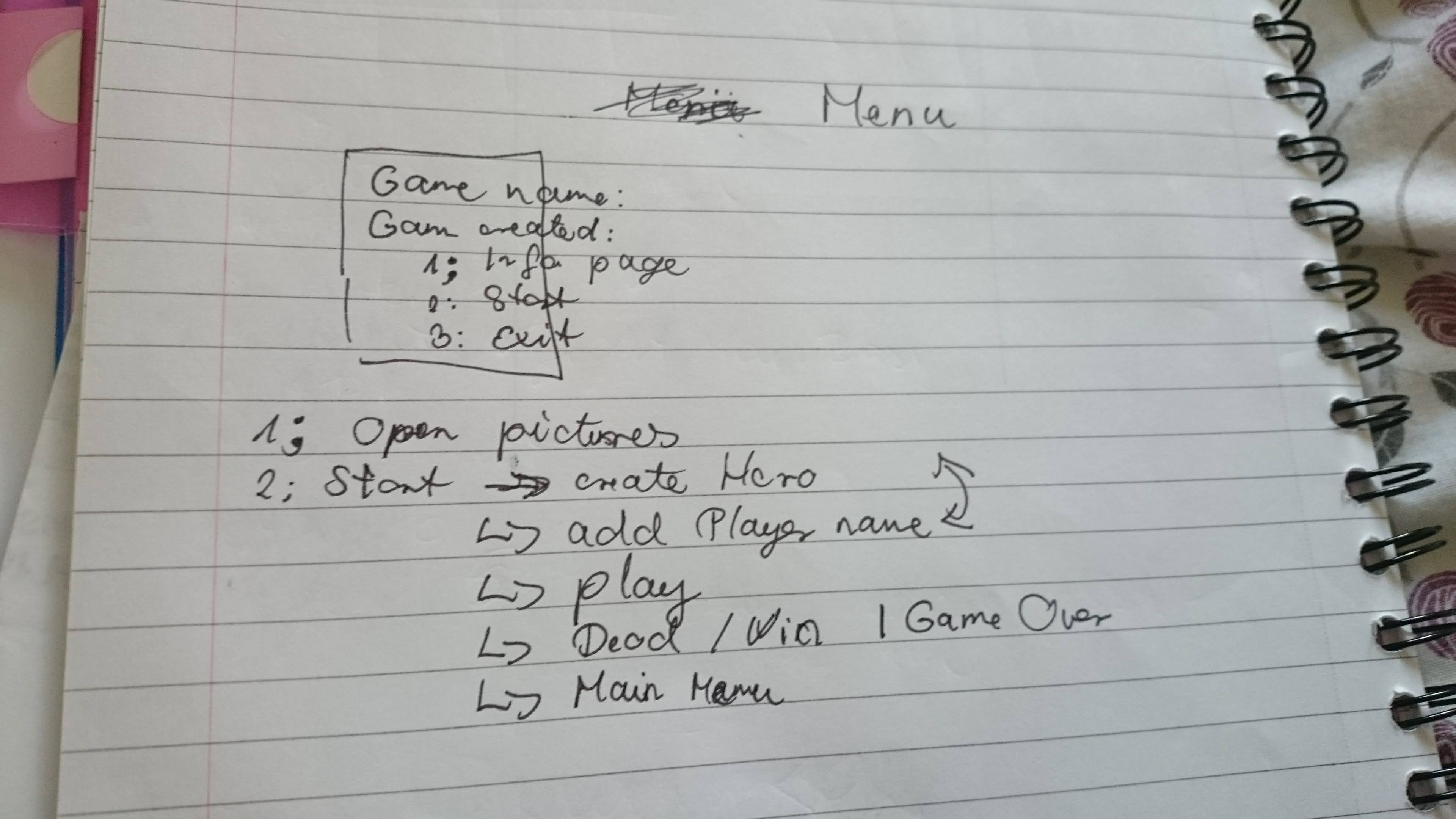
# Section 3: Development

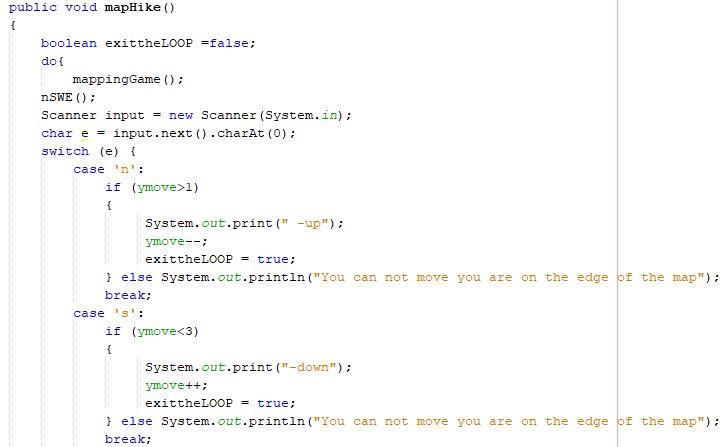
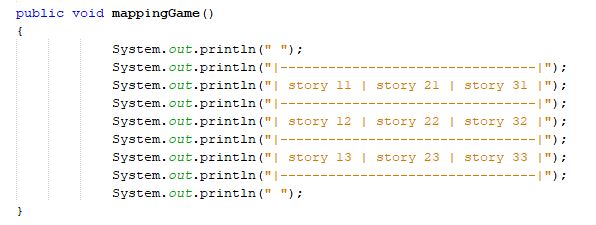
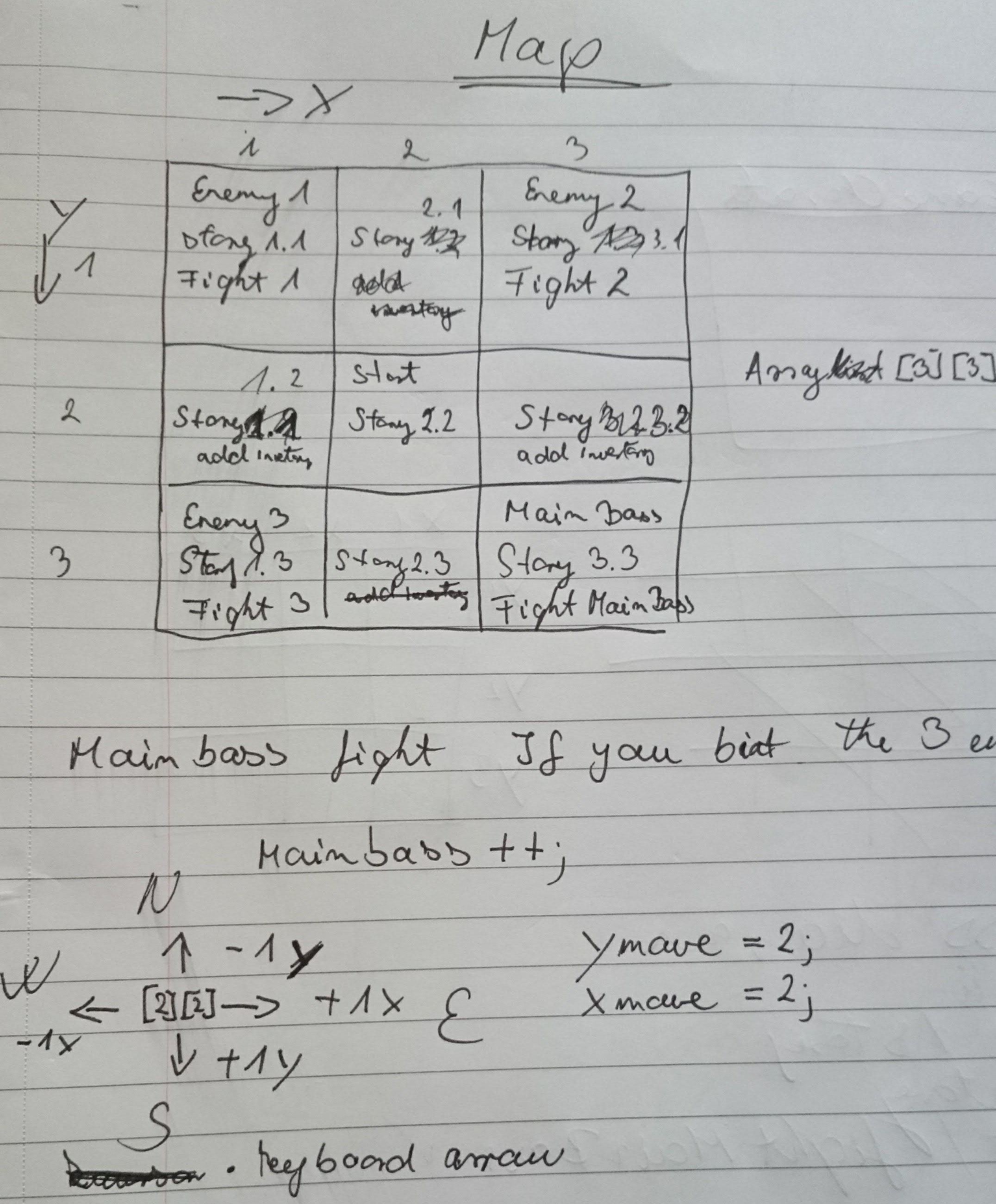
Class layout / data structures used / logic flow of program. Why you decided to structure your program the way you have (diagrams / screen shots).

**Abstract classes**

**Race abstract class**

**Race abstract class program**

**Game menu layout**

**Map class design**

 public void fightWithEnemy()

{

defend = 5;

magicCounter = 3;

System.out.println("====");

System.out.println("main character");

a[heroCharacterF].info();

if (heroCharacterF < 4) {

do {

enemy = r.nextInt(3) + 4;

System.out.println(a[enemy].getHealth());

} while (a[enemy].getHealth() < 0);

System.out.println("====");

System.out.println("Your ENEMY character");

a[enemy].info();

}

if (heroCharacterF > 3) {

do {

enemy = r.nextInt(3) + 1;

System.out.println(a[enemy].getHealth());

} while (a[enemy].getHealth() < 0);

System.out.println("====");

System.out.println("Your ENEMY character");

a[enemy].info();

}

do{

// different attack

do{

System.out.println("Your choose Attack :1 Defend :2 Magic :3 Run Away :4 ");

Scanner h = new Scanner(System.in);

if (h.hasNextInt())

{

attachChoose=h.nextInt();

}

switch(attachChoose)

{

case 1: // attack

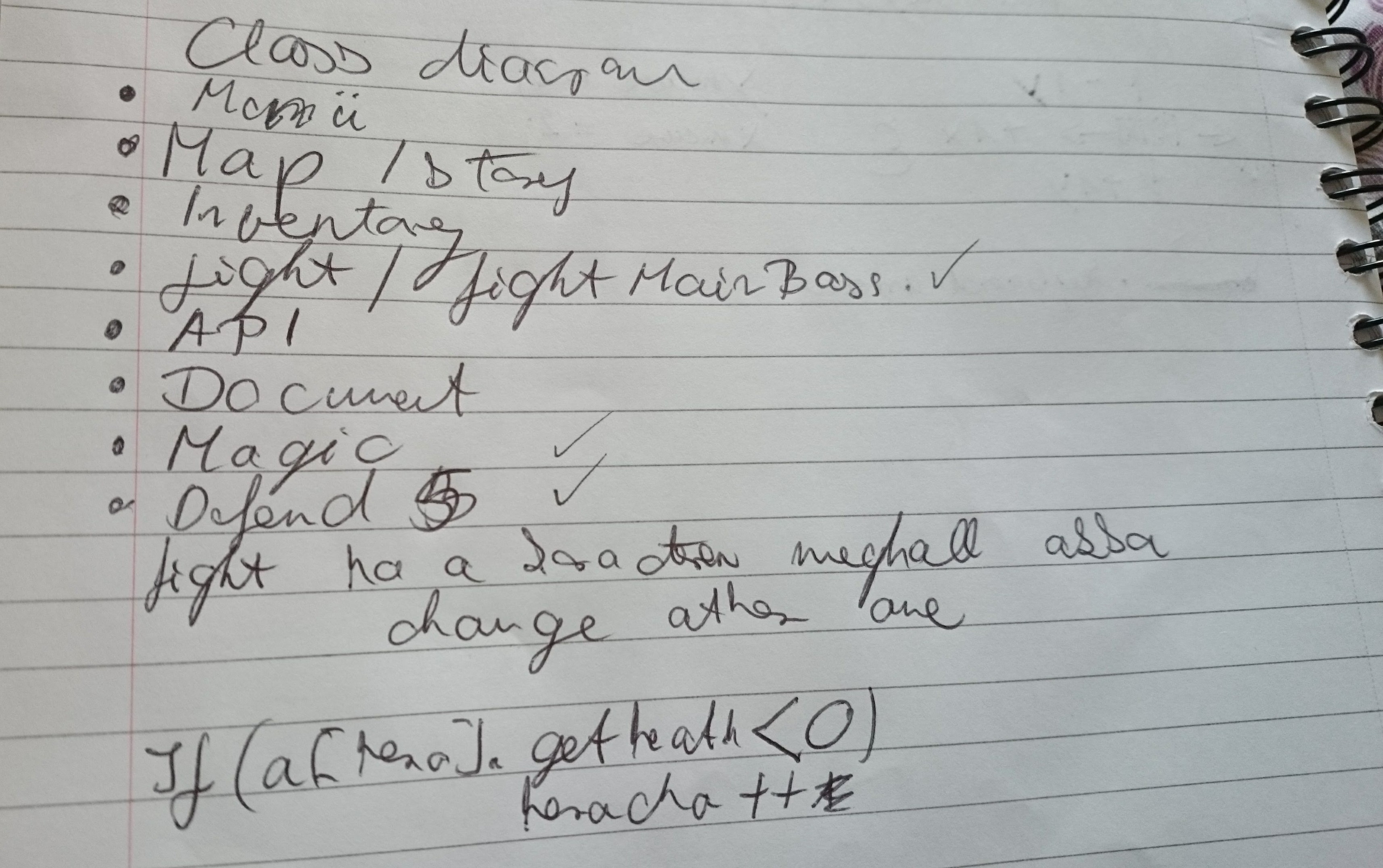
attack();

break;

case 2: //defend

defend();

**fight.fightWithEnemy(); method**

**Program debugs list**

# Section 4: Reflection

I my opinion It was difficult to me to create a lot of classes and I had to pass the objects between the classes. It was important to make good design on the paper and this layout had to be transferred to the program. If I had more time I would like to extend the map and maybe to change the combat system, other way give more AI support. I could use a jframe to make more visual display.

I have learned about the inheritance and the polymorphism. Those help to make more complex program and give opportunity to work on the team (abstract class). After I finished the java program I understood importance of collaboration in real life.