Phase 2 Report

(Group 11)

Overall approach to implementing the game

At the beginning of the development of the game, we first determined the role setting of the game. That is, the user's role in the system and the responsibilities that can be performed. We assume that users can manipulate red blood cells, fight enemy viruses, and collect drugs on the ground to obtain credits. If the user encounters a trap on the ground, the score will be subtracted. We also set that the enemy virus will follow the user and attack the object operated by the user. We made goals through setting and implementing SMART goals. We discussed what would be the outcome of the game, the objective that the player needs to accomplish. The game mission would be clear throughout the game and is time bounded in a session when the player starts the game. Our next step was to choose an IDE being IntelliJ as our working environment in creating code as well as learn the syntax of Java as we had limited knowledge of its structure. Lastly, we had to divide the work between members, so that we can all contribute and share with each other our work in each team meeting. This allowed us to divide and conquer so that the work wouldn't be as complex when we go through each person's work again. We also thought about having multiple classes and implement low coupling so that the design would be easy to understand and easy to maintain for future development of the code.

The adjustments and modifications to the initial design of the project

In the first stage, we have customized a series of classes belonging to our own game by referring to the introduction of functions in assignment. The most important part is the main part of the game, which is the game panel. On this panel, we successfully implemented the display panel that displayed the scores, credits and time; the wall blocking users and enemies; the generation of static and dynamic enemies; the starting point and end point of users. However, due to the more in-depth study of the game, we found that the initial project planning is unreasonable. For example, we hope to display a mini map on the game interface, which can let users know where they are. But because the game interface we set is small, users can directly observe the whole map. So, we omit this function. At the same time, in phase 1, we store functions such as reward, count score in the character class. But later, we hope to manage the calculation of scores more effectively, so we wrote separate classes for them. In this way, when character encounters enemies or rewards, we can call functions directly to make the whole code simple and clear. Finally, for a variety of reasons, given our limited time, we decided to focus most of our attention on completing the main game panel before continuing to add additional features. So, we had to give up the design of the main menu. Other than that, we are going to stick to our UML class diagram as the rest of the class diagrams relate to how we are going to implement our game. For our use case, it is very similar to the game we have coded; however, we had to make modifications and limit what the player can do in the map. We did this due to our inexperience with Java though during the next phase we are aiming to add more functionality for the player to perform to create a better game experience.

The management process of this phase and the division of roles and responsibilities

During the management process for this phase, we focused on setting short term goals, planning and deciding what each person's role and responsibility would be. The first week of phase 2, we began to discuss splitting in teams of two one doing the main menu screen and one doing the main game interface. However, after much discussion and test and error of implementing the game, we as a team decided that we should focus our attention on the main game panel. We did this as the main game interface required many functions and it was very hard to accomplish with only two people working on it. So, we left one person in charge of designing and implementation of the game room, one creating the dynamic character, another creating the static and non-static enemies. This way of splitting the work allowed us to work at our own pace independently and not feel rushed by others to finish a task. Additionally, during our biweekly meetings, we would report on our progress and assess our situation from there.

The external libraries we used

We used many external libraries like Java swing. Java swing provides the ability to create graphical user interface components. We used this to display the main screen using JFrame and added a JPanel where we added various components including the game map, player character, enemies, rewards, traps and scoreboard. We also used the java package awt rectangle. This was used for collision detection to get the bounds of each object in our game. We used this for collision detection to prevent the characters and enemies from being able to go over the boundary walls. The array list import was also used for the purpose of constructing the objects like enemies on the map.

The measures you took to enhance the quality of your code

First, we learn the game development by experienced programmers through the network, and learn the Java language by ourselves, so as to improve the quality of the code.

Second, we develop the relevant coding standard and coding pattern to require the developers to follow. This can effectively let other members know the use of each code, ready to take over later projects.

Third, we use related tools to analyze the code statically and dynamically and find the problems in the code as early as possible.

Fourth, we perform code review. Code review is required before each code submission to try to be correct at each submission, so as to save time for later modification.

The biggest challenges you faced during this phase

There are two main difficulties in phase 2: first, programming difficulties. Second, time management difficulties.

First of all, due to our lack of Java knowledge, we had to make considerable adjustments and modifications to our initial design, which took us a long time. And while coding, we came across many issues in our implementation like character movement. The problem was that the operating system was not carrying out the correct user input instructions due to bugs in our code. This was a huge problem as the game is centered around what the character performs on the map. We decided to use the key pressed and key released functions to monitor the user's behavior, and after many successions, we were eventually able to make characters movement follow user expectations. Although our code is not overly complicated, most of our code was self-taught. Through trial and error, we were able to accomplish a lot in this phase. Even after going back to the drawing board as many of our partially complete code was not working how we wanted it to. Though this was a problem for us, we hope to improve and enhance the quality of our code in the following phases.

Secondly, the challenges we faced as a group during this phase was time management. Although we held biweekly meetings, there was always one person who could not attend the meeting. This impacted our work significantly as we would have to go through with them what we had done the last meeting which delayed our progress. This consumed time especially since we all had a busy schedule and had to give in a big portion of our time to meet up in person and discuss our progress. This also affected our team's performance as one person had to put in more effort than another person to keep the project progress running forward. We had these in person meetings with the purpose of helping each other out with their problems and create less confusion in contrast if they were doing their task alone. Another challenge we faced was how we split up the group work. When we assigned each other tasks that needed to be accomplished, we all went ahead and made our own implementation of code. This resulted in confusion during meetings as one's own code was structured differently then another person. We had a hard time explaining and learning each other’s code delaying our workflow. As such, we decided that we would focus on one implementation of code and build upon from there. This greatly impacted our performance as we had to meet the deadline as well start from scratch multiple times. Another challenge we faced was disagreements. We all had our own opinion on what we should implement in our game and disagreements on what should be included. We came up with many ideas however, given the timeframe and lack of coding experience, we were very limited in our options.