PHASE 2 - REPORT

1. Implementation approach
    a. Factory method - allow extension of game to create variation
    b. Structure of the classes
    c. Primarily focus on core algorithm of the game
    d. Score board implemented with string reading - keep top 5 in array list
2. state and justify the adjustment / modifications to initial design (class diagram / use case
    a. Adjust and update UML diagram to include abstract factory class to allow inheritance and variation of game (such as enchanted room and spells)
    b. Remove log in information and remote access from use cases
    c. Retain Cell and Play classes, remove the rest of classes
    d. Add cellmanage class, which helps to mange each cell we are using
    e. Add Entity class. It represents basic Entity in the game. Also, it serve for other entity includes property;
    f. Add CollisionChecker class. It serves for if 2 entities collide, then game over.
    g. Add GamePanel class. It serves for executing the game logic.
    h. Add KeyCatcher class. It serves for receiving signal from the keyboard where the play pressed
    i. Add OBJ_key class. It serves for representing a key object in the game
    j. Add SuperObject class. The basic class for various object in the game, includes images, name, collison status etc;
    k. Upload some 2d diagram which represent objects in the game like player,enemy etc
3. explain management process of this phase and the division of roles and responsibilities.

    Phase 2 code breakdown / outline

    Choosing libraries (find useful library that may be helpful for the game implementation and summarize methods and functionality of library) - Janet W

    Raw codes and comments -Yizhou Lu

    Creating folders / modules / separation of classes into appropriate files for organization-haoxin w

    Update UML diagram according to abstract factory - Linli

4. list external libraries used (GUI) - justify the reason(s) for library

    a. ArrayList - keep score board
    b. AWT - Organize Image
    c. SWING - Graphic control and Panel Control
    d. ImageIO - Read and write in image
    e. IOException - checked exception that indicates problems
5. describe measures to enhance the quality of your code
    a. Measure the code's testability. It refers to assessing how easily and effectively the code can be tested. Testability influences the ability to verify the correctness of the code through testing processes. When measure the testability of code, you are essentially evaluating how well the code. accommodates the creation and execution of tests.
    b. Measure the complexity of the code. The purpose of measuring code complexity is to gain insights into how difficult the code may be to understand, maintain, and debug.
    c. Measure the reuseability of code. It aims to maximize the utility of code by making it adaptable and applicable across different scenarios.

6. discuss the biggest challenges you faced during this phase.
    Understanding the purpose of these libraries we are using in order to use them more appropriately.

<u>THEMEs:</u>

1. CrazyFarm

Libraries Use:
java.io.IOException(Gives checked exception that indicates a problem);

javax.imageio.ImageIO(Read and Write in Image data);

java.lang.Thread.sleep(When a processor is finished, let it sleep and give cpu to another processor);

java.awt.*(Use the package Awt);

- java.awt.image.BufferedImage;(Java awt image is going to create and modify image, bufferedImage is an image with accessible buffer data)
- java.awt.event.KeyEvent(Generate event);
- java.awt.event.KeyListener(Listener interface to receive keyboard event);
- java.awt.Color(Control Image Color);
- java.awt.Dimension(Control Image Dimension);
- javax.swing.JPanel(Organize components and layouts);
- java.awt.Graphics(Generate Graphic);
- java.awt.Graphics2D(Gives control of 2 dimension of the graph);
- java.awt.Rectangle(Specified x and y coordinates' spaces);
-

javax.swing.*(Use the package Swing);

- javax.swing.JPanel(Organize components and layouts);
- javax.swing.JFrame(Foundation of creating Graphic Java Application);