CMPT 276 - Phase 4 Report

Group 22:

Michael Plunkett (mplunket@sfu.ca)
Dina Zeng (dhz1@sfu.ca)
Sameer Hossain (sameerh@sfu.ca)
Salman Rafiei (salman_rafiei@sfu.ca)

Overview of our game

Main menu, difficulties, and levels
Level design + UI
Items: Door, key, potion, spikes
Enemies: bats, skeleton, slimes

How our plan changed over time

Original plan:

- Dungeon Manager Escapes is a 2D maze game set in a dungeon in a fantasy setting. You play as a distinguished priest who has been fired from the church due to budget cuts. You now find yourself employed by a black company “LIVE LLC.” to manage a “building”. However, on your first day of work you see that said “building” is a dungeon, and you really regret not reading through the contract. Now that you have read the terms and conditions you found a loophole in the contract stating that the contract is valid so long as you’re in the dungeon, therefore you must now escape.
- You must venture through several levels, collecting keys to unlock the door to the next floor while avoiding traps and mobs that move randomly, patrol and chase the player.
- There are health potions that the player must collect in order to stay alive and keys that unlock the next level.

Changes we made:

- Our final project stayed more or less true to our original concept. We structured our code similar to our original UML diagram with a couple of changes to make the development process easier such as adding more classes and renaming
- The UI is different from our original mockup as it was very primitive and we did not have a concrete vision of how our game was going to look.
- Originally there were supposed to be 2 enemy types, 1 to chase the player and the other patrols. We started off with a basic enemy type that just wanders aimlessly as we didn’t know how to implement pathfinding hence we ended up with 3 enemy types.
- We implemented 2 entity classes, 1 for moving characters (player & enemies) and 1 for inanimate objects (keys, spikes, doors). It made more sense to have the punishment(spikes) be an item that the player interacts with than being part of the enemy class.

- The door also extends the item class rather than being part of the tiles/map components as it is an interactable object rather than a map tile.

What we learned

Git:

- Managing a git repository, creating README, using .gitignore
- Basic skills & becoming proficient with committing, pushing, pulling, and using branches
- Resolving merge conflicts
- Creating meaningful branches for specific tasks
- Understanding git's history graph was incredibly useful

Java

- Design patterns and high level object oriented design
- Importance of encapsulation and singleton classes which made code cleaner and easier to read, had we done so in the beginning our classes would be less coupled
- Maven (build automation) made it easier to build the project in different environments without worrying about the dependencies, processes, etc. since we were using 2 different IDEs(IntelliJ & VS Code)
- Javadocs (documentation) which makes it easier to understand the implementation of functions
- JUnit (Unit and Integration tests)

Teamwork/communication: these things are important…

- Managing communications via discord with different channels for different things
- Holding bi-weekly online SCRUM meetings
- Holding in-person sprint sessions
- Sharing documents and staying up to date with each others work
- Delegating workload

Include link to our video:

https://www.youtube.com/watch?v=FEo5Omps2wY

References:

Game structure:

1. Ryisnow. (n.d.). How to Make a 2D Game in Java. YouTube. Retrieved April 9, 2023, from https://www.youtube.com/playlist?list=PL_QPQmz5C6WUF-pOQDsbsKbaBZqXj4qSq

Game assets:

2. Pixel_Poem. (n.d.). *PIXEL_POEM DUNGEON TILESET*. Retrieved April 9, 2023, from https://pixel-poem.itch.io/dungeon-assetpuck

3. 患者長ひっく(patient head scratch). (n.d.). マルモニカ【無料・商用利用可】- *x0y0pxfreefont - booth*. x0y0pxFreeFont. Retrieved April 9, 2023, from https://00ff.booth.pm/items/2958237

4.