Wang Yuanxi A1805637 - Chan Jing kai Germin A1805312

PROJECT SPECIFICATION - MAJOR PRACTICAL

INTRODUCTION

Our project implements a rogue-like dungeon role-playing game. In the game, score is earned through progress and coins, while defeating monsters to increase the player's combat power. Items such as health and attack buffs can also be collected along the journey.

DESIGN DESCRIPTION

Assessment concepts

Memory allocation from stack and heap

- **Vectors:** We will be using vectors to create Zombies, Skeletons at increasing power as level increases.

- **Strings:** Game introduction and guide at the start of the game. Action move e.g. attacked zombie. Zombie left with 3 health. Zombie killed.

- **Objects:** Skeletons, Zombie, Apple, Sword, Potion

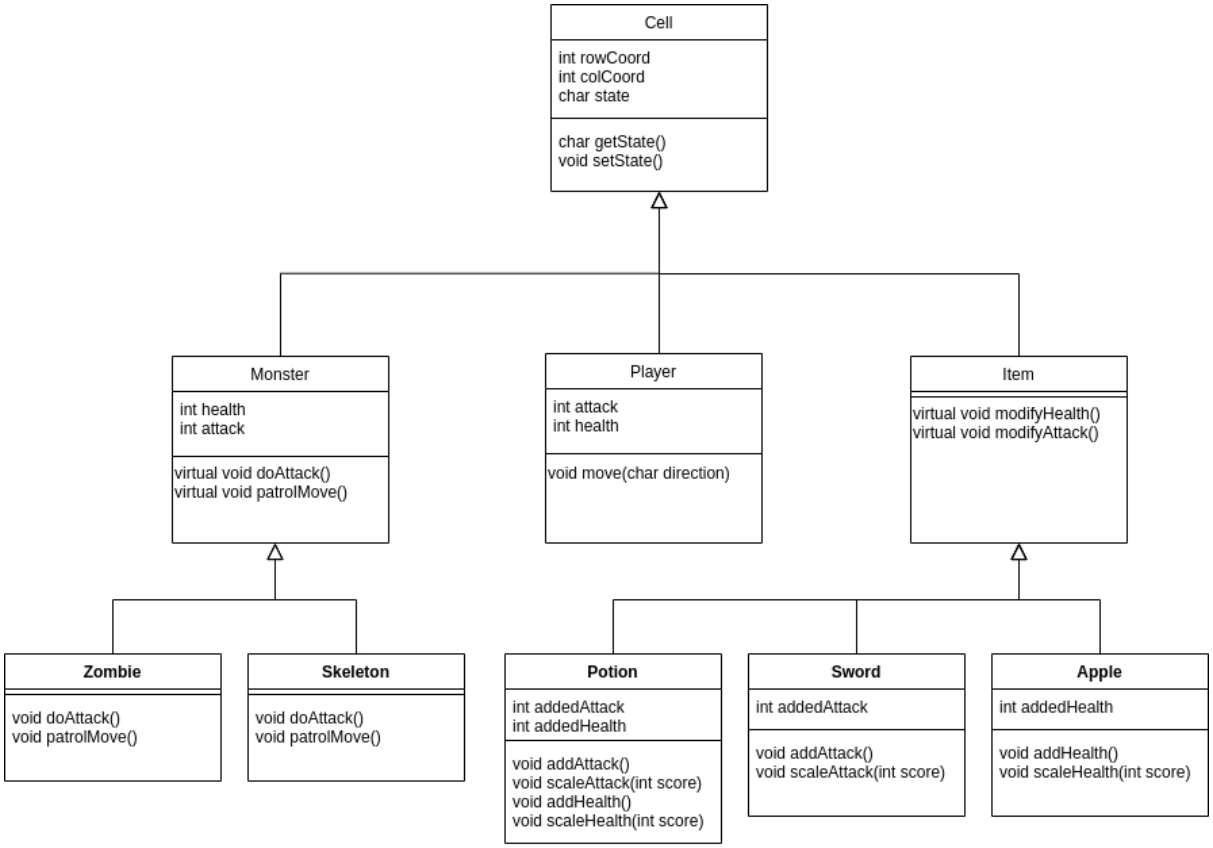
User input and output

- We have a getch() which reads user input to control the player in the map.
- Login/Logout which reads username and password to store score history as well as stats so the player does not have to start from the beginning.

Object-oriented programming and design

- Inheritance: There are different item types which increase different stats. Apple restores health. Sword gives attack. Potions give health and attack.

CLASS DIAGRAM



CLASS DESCRIPTIONS
present:
infinite dungeon crawler rpg
testing strategy:
playthrough to debug
test cases to test specific encounters(combat vs different enemies/movement) q
2 week plan:
week 10:
2 test situations(combat/death/movement)
additional features(items[inheritance + polymorphism], ability/health scaling)
week 11:
code cleanup (port to sdl?/better icons?)
additional features(mob patrol, ranged attacks, ability/health balancing, more classes?, boss fights?)

User interface

A player of this game will be able to read the game introduction. After which the program will prompt users if they are a new user or existing using.
If new, they will be asked to register an account.
If existing, they will be asked to login to their account.
Once a user has login, points and stats will be loaded from history.
If new users, health will be 25 and point is 0.

Code Style
All code in the program will be properly indented using 4 space tabs.
Classes will begin with a capital letter. Class files (.h and .cpp) will begin with capital letter.
Function Comments will be given for each function that describe what function does.
Comments will be written as the code is written.

TESTING PLAN
All our tests will run through our makefile each time we compile our code, automatically..

Unit testing
Our unit test will cover all public functions in our classes. Each class will have a corresponding test file like Zombie.cpp Skeleton.cpp which will include a main method that will exhaustively test each function with a set of inputs and test it matches an expected output.

- Player will be able to move when ‘w’ ‘a’ ‘s’ ‘d’ is pressed.

- When player touches wall, nothing happens

- When player touches zombie:
  zombie will decrease in health according to player attack
  player will decrease health according to zombies attack

- When player touches skeleton:
  Skeleton will decrease in health according to player’s attack
  Player health will decrease according to the skeleton's attack.

- Player score will increase when:
  Moving
  Collecting coins
  Defeating zombie/skeleton

Schedule Plan
Stretch Goals
Our goal is to complete most of the functional features by week 10; if we have finished testing by this point, we can expand our program to include more monster behavior such as patrols and ranged attacks, more classes for the player to have different attack/health behaviors such as archers or tanks, and more class-specific items.

Week 9
Write Makefile - Germin

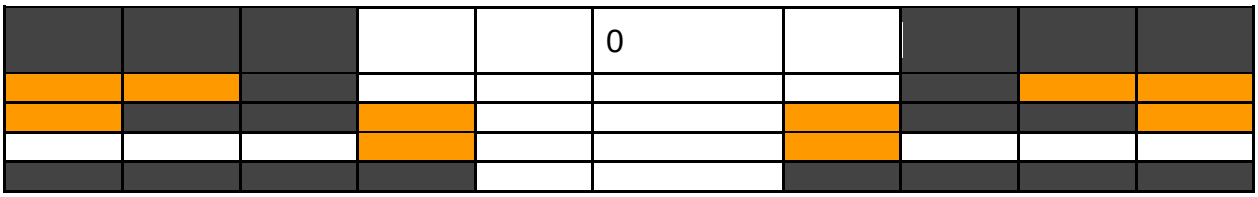
Organize code sharing with group members -YX
Finalize testing strategy - Germin
Create infinite map generation and destruction -YX
Include skeleton and zombie that changes according to attack and health - Germin

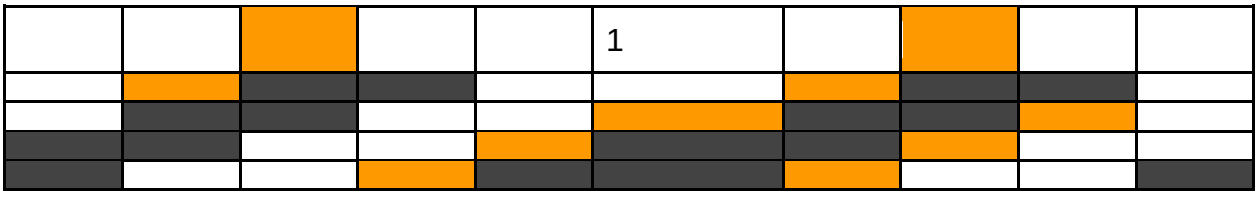
Week 10
Checklist:
Turn player into class - YX
Scale monster stats with score - YX
Code architect and arrangement and comment for ease of readability - Germin
Add items and behaviors - Germin

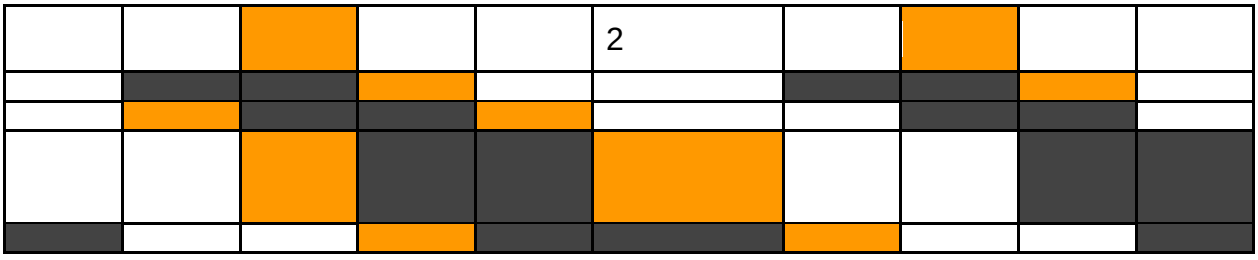
Week 11
Checklist:
Cleanup code (remove unnecessary code) - Germin YX

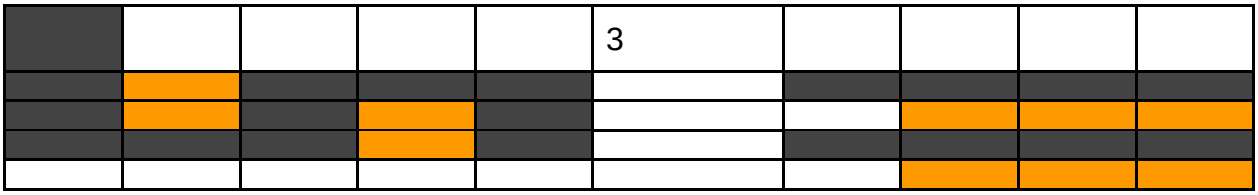
Rules for world gen presets:

Either 0,5 or 0,6 must be empty
Either 4,5 or 4,6 must be empty
Internally traversable (no need to leave 5 row area to access places)
(can only backtrack by 2 cells max)
Current world presets wall coin









4

5

6

7

S