Original Design Outline/Plan

SPACE CLASS

- Abstract class
  - Pure virtual functions
  - 4 pointers
    - top
    - left
    - right
    - bottom
  - Create game w structor of linked space

GAME PLAY

- Must have theme
- Must have goal for the player
- Keep track of player location (which space)
  - Can print map or just print text
- Container for carrying items
  - Must have capacity limit
- Items
  - Player can collect throughout the game
  - Must be required as part of solution to game goal
- Must have time limit
  - Time, steps, turns, health systems that decreases with each move and has painkillers scattered throughout
- Must interact with space structures, not a collect all times game

INTERFACE / MENU

- Print goal of game at start
- No free form input
- Provide menu option for each scenario of game

NOTES

- Space class
  - Can have more pointers but not required
  - Unused point to null
- 3 derived class at least
  - Each represents different type of space
  - Special action for player
    - Ex: attack monster, open door to another space, turn on the switch, sing a song to please the king
- Game
  - Has at least 6 spaces

REFLECTION DOCS

- Test plan
- reflection
- Design description

Spaces Classes

- **Standard Derived Class**
  - Loose a life point
  - Print a helpful message
    - Keep a rotation of helpful messages
- **Heart/health**
  - Get +2 extra life point
  - Print a message
    - As many messages as health spots
- **Riddle**
  - Must solve a riddle
  - Win = get 1/3 of trifecta
  - Loose = -2 life points
  - Withdraw = -1 life point
- ??*Boss (evil)*
  - spells and swords have no effect until trifecta is applied
  - Then spell and sword have effect
  - Boss had 25 life points
    - Down to 15 after

Backpack Class

- Keep track of 3 (or 5) items in a (double) linked list
- functions
  - Remove item
  - Add item
  - get item
  - displayContents

Warrior Class

- Keep track of health points; start with 10 points
- Keep track of position
- 4 attack types
  - Dodge
    - -2 on opponent
  - sword
    - -4 on opponent
  - Trifecta (must be collected to unlock)
    - -10 on opponent
- Functions
  - Get position
  - Get health/time
  - attack
    - Menu for type of attack
- ??combine with backpack class??

Menu Class/Functions

- StartMenu
  - Intro message and such
- Between moves menu
  - Check back pack
  - Check health
- Moves menu
  - Up, down, left, right
- Input validation
  - Use only listed # inputs

Board

- 4x4 Static board
  - 3 challenges
  - 3 riddles
  - 1 boss
  - 3 health
  - 6 standard spaces
- Functions
  - Build board at start
  - Print board
    - Show where player is

Game Play

- Print intro message
  - What's the goal
  - What's the background/story
- Print board
  - Show where user is
- Loop:
  - moveMenu -> choose space
  - Print board
  - Play out contents of space
    - Standard: -1 life point
    - Heart: +2 life points
    - Riddle: get item
    - ???Boss: save the world, exit kingdom??
- Game over
  - No more health points OR you win
  - Play Again?

Writing

- riddles
  - 1
  - 2
  - 3
- Attack messages
  - Attack succeed
  - Attack fail
- Intro bit
  - backstory
  - goal: fight some stuff, collect trifecta
- Health messages
- Standard space messages

Red= health
purple=riddle
green=boss
blue=standard

	0	1	2
0	START	1, 0	2, 0
1	0, 1	1, 1	2, 1
2	0, 2	1, 2	END 2, 2

Reflection

I originally had the idea for 1 more space as well as a 2D board array. After consulting the requirements again, I realized that the board seemed to be required to be a linked list. This seemed like a somewhat easier approach and I think it was less laborious than my origin 2D array plan.

I was also going to do a bigger board, but for the sake of time, I limited it to 3 before starting coding. I think I took the right path in minimizing my plan's complexity. I found I had plenty of errors to work through with the smaller scope and I'm glad I didn't go beyond my limits to make a more complex game. I ended up with a rough time fighting a memory leak, but it turned out to be a silly error where I was asking an array to go out of bounds. I changed a 4 to a 2 and all was well.

This time, I did a better job of breaking my project into reasonable chunks that were relatively detangled from each other. I used ZenHub (a GitHub *chrome extension*) to organize the project into 4 major parts. I had the writing (the fun part for me) as the last major chunk. This allowed me to focus on game mechanics before I got lost in finding fun riddles. This decoupling of major tasks allowed me to do memory leak checks on flip at regular working intervals. I think this was key to only having 1 memory leak error to work through towards the end of the project instead of 10 or 12 (like in past projects).

Test Table

Test	Test Target	Input	Expected Output	Actual Output
No memory leak errors	Program	Regular game play	No memory leak errors	No memory leak errors
No seg faults	Program	Regular game play	No seg faults	No seg faults
Compiles on flip	Program	Regular game play	Compiles on flip	Complies on flip
BetweenMenu works properly	between Menu	Regular game play	• Reappears after every move • Doesn't appear for boss fight	• Reappears after every move • Doesn't appear for boss fight
MoveWarriorMenu Works properly	moveWarriorMenu	Regular game play	Only shows options for available spaces	Only shows options for available spaces
Riddles work properly	Riddle class	Regular game play	• Riddles can only be solved once • Riddles subtract strength from warrior when with wrong answer • Riddles subtract strength from warrior when nothing to solve	• Riddles can only be solved once • Riddles subtract strength from warrior when with wrong answer • Riddles subtract strength from warrior when nothing to solve
Gargoyle	Boss class	Regular game play	Attack damage decreases with decreases health	Attack damage decreases with decreases health
Warrior moves properly	Move	• U • D • L • R	• Warrior location is now up linked space • Warrior location is now down linked space • Warrior location is now left linked space • Warrior location is now right linked space	• Warrior location is now up linked space • Warrior location is now down linked space • Warrior location is now left linked space • Warrior location is now right linked space
Warrior attack works properly	Attack	• 0 • 1 • 2	• Trifecta attack, -6 from gargoyle • Fists of fury, -1 from gargoyle • Sword attack, -3 from gargoyle	• Trifecta attack, -6 from gargoyle • Fists of fury, -1 from gargoyle • Sword attack, -3 from gargoyle
Warrior defense works properly	Defense	Regular game play	• 50% of the time 'shield up' and no damage taken	• 50% of the time 'shield up' and no damage taken
User map prints	PrintMap	• 2	• Subtract 2 health from warrior • Map with warrior position printed	• Subtract 2 health from warrior • Map with warrior position printed
Regular map prints	printMap	• 1	• Subtract 1 health from warrior • Map without warrior position printed	• Subtract 1 health from warrior • Map without warrior position printed
Backpack works properly	Backpack class	• Solve riddle • Print contents	• Trifecta peaces added to the back • Contents printed to spell out trifecta OR Print "bag is empty message	• Trifecta peaces added to the back • Contents printed to spell out trifecta OR Print "bag is empty message
Standard space works properly	Standard space class	Regular game play	• Subtract 1 health • Output message	• Subtract 1 health • Output message
Health spaces works properly	Health space classes	Regular game play	• Add 2 health on first visit • -1 health on other visits	• Add 2 health on first visit • -1 health on other visits