Individual Assignment

## Due 3/26

Congratulations, your game “Riders of Fortune” was a success. The Magic Gaming company wants a sequel and you listen to the fans and decide to help Yarra continue his adventure.

For version 2.0 “Dungeon Escape” you will need to make the create the following:

### Game Modes

When you start the game, you will have the following menu:

Welcome to Riders of Fortune 2.0: Dungeon Escape. Choose to:

1. Start a New Game
2. Load a Saved Game

If the player chooses 1, the game will start as normal. If player chooses 2 it will load the current game state from saveGame.txt

### Main character

You will use Object Orientation to create a player object. The player will now contain the following traits.

* HP: Health Points. This amount will start with 100 (max) and will decrease when attacked. If it reaches 0, the player will die. It can also be restored.
* XP: Experience Points: This amount will start at 0 and grow as needed.
* BoardPosition: Where the player is on the board.
* Equipped Weapon: The Equipped Weapon.
* Alignment: GOOD or EVIL. Default to GOOD.
* WeaponCollection: The weapons that have been acquired.
* ItemCollection: The items that have been acquired.

The player object will also have the following methods or actions:

* WeaponAttack() : This will execute an attack on an enemy with a weapon and will return the damage.
* Run(): This will allow the player to escape from a battle.
* UseItem() : This will allow the player to consume an item and its effects.
* ChangeAlignment(): Changes the player’s alignment.

### The Game Board

You will now create a map which allows the player to move continuously, as follows:

S \* \* \* W \* \* C Places Legend:

\* \* S = Player Start

\* \* C = Cathedral

E G G = Graveyard

\* \* W = Well of reflection

\* \* E = Dungeon Exit

\* \*

G \* \* \* W \* \* C

The player starts in S and can continue to move forward to different spaces on the board. When the player gets to a corner of the board, they must continue to move forward vertically or horizontally in a positive direction. The player can overlap the board as many times as they want, but they must move in a positive direction.

As in project 1, you will provide the player with a menu:

1. Travel to another space on the board
2. Dismount and explore the current space
3. Save your game

When a player lands on a place, the following effects will occur to the player:

Cathedral: If your alignment is good, you will gain 20% HP, else you will lose 20% HP (round up to the next integer, do not pass max limit).

Graveyard: If your alignment is evil, you will gain 20% HP, else you will lose 20% HP (round up to the next integer, do not pass max limit).

Well of Reflection: Increases your XP by 2 and your alignment is changed.

Dungeon Exit: This will allow you to exit the dungeon if your XP is 20 or higher and you have found a magic relic.

Landing on any other places will result in one of the following random events:

1. An enemy appears and a battle takes place.
2. You find an item
3. You find a weapon

### Saving the Game

If the player chooses to save the game, the state of the game will be written to a binary file called saveGame

This file will only need to hold one saved state a time, so saving will overwrite the existing save.

### Enemy Battles

The battle will now last multiple rounds, so they will continue until either the enemy dies or the player dies.

The player can now choose to ATTACK, RUN or USE ITEM

**Attack:**

If the user chooses to attack, Display the weapons that the user currently has and ask the player to choose a weapon. Once the user chooses the weapon use that weapons modifier for that attack.

An attack from the enemy will result in HP damage.

Defeating an enemy will increase the XP of the player by 30% (Use the integer value and round up).

**Run:**

If the player chooses to run, choose a random item that will be dropped by the player

**Use item:**

The player can choose to use a potion. If they have any in their collection and the effects will be applied.

### Regarding Enemies

The enemies themselves will have HP which is random value that increases as the player’s XP increases and will be immune to a weapon type.

Example:

Player XP Enemy HP

1. Random value (1-5)
2. Random value (1-10)
3. Random value (1-15)

Etc…

When you fight an enemy, randomly pick a weapon type that the enemy is immune to.

### Weapons (Struct Implementation)

Create a Weapon Struct to hold a Weapon which has the attributes: Weapon Type and Attack modifier.

You will randomly find one of these weapons types:

* Crossbow
* Flail
* Whip
* Mace
* Staff
* Broad Sword

An each will have a random attack modifier from 1 to 20. When the weapon is found, it is added to the vector.

Keep these in a vector, display a list of the weapons type to the user and ask them to select a weapon. Retrieve the weapon and use its modifier in your attack.

### Items (Dynamic Allocation)

Your item bag will start out with 5 slots, so use an array of 5 to create the item bag. As you land on an item space, you will

You will randomly obtain one of these items:

1. Potion 1: Increases HP by 5 when used.
2. Potion 2: Increases HP by 10 when used.
3. Magic Relic: Allowed you to pass through the dragon entrance (E)

If you reach your item bag limit, you will have to create a new bag with space for 5 more items and move all the items to the new bag.

HINT: If you find more items than its current size, you will dynamically allocate a new array to hold the n+ 5 items. Continue to do this every time you an additional reach your arrays limit. Review Chapter 9 for ideas.

**End Game:**

Use the same End Game scenario from assignment 1. The differences will be the following:

1. You exit the dungeon when you reach an XP of 20 or higher, land on the dungeon exit and have magical relic item.
2. If the player has all the requirements 1, they will exit the dungeon and have completed the game.

S \* \* \* W \* \* C Places Legend:

\* \* S = Player Start

\* \* C = Cathedral

E G G = Graveyard

\* \* W = Well of reflection

\* \* E = Dungeon Exit

\* \*

G \* \* \* W \* \* C