RPG Specification

# Folder Structure:

## docs:

All documentation pertaining to the project.

## resources:

Any libraries or external files which are not part of the source code.

## saves:

Game saves created when the player saves the game.

## src:

Project source code.

## src/headers:

All header files required for compilation. All .cpp require a header definition except for small files such as main.cpp.

All header files must have an include guard with a relevant name. Ex. foo.h would be #define FOO\_H.

# Formatting:

Curly braces should be inline with the statement they are enclosing.

Use tabs which are two spaces in size.

# Project Structure:

**Class Equipment:**

- name : String

- requirement : int

- stat : int

+ Equipment()

+ Equipment(std::string, int, int)

+ getName() : std::string

+ getStat() : int

+ getReq() : int

# setName(std::string) : void

# setReq(int r) : void

# setStat(int s) : void

**Class Weapon extends Equipment**

**Class Shield extends Equipment**

**Class Helm extends Equipment**

**Class Chest extends Equipment**

**Class Pants extends Equipment**

**Class Creature:**

- String name

- level : int

- int maxHealth

- int health

- int attack

- int defense

+ Creature()

+ Creature(int, int, int)

+ subHealth(int) : void

+ addHealth(int) : void

# setName(std::string) : void

# setAttack(int) : void

# setDefense(int) : void

+ getAttack() : int

+ getDefense() : int

+ getName() : std::string

+ getHealth() : int

+ getMaxHealth() : int

**Class Player extends Creature:**

- int experience : int

- weapon : Weapon

- shield : Shield

- helm : Helm

- body : Body

- pants : Pants

- calculateStats() : void

+ Player()

+ Player(int, int, int, int, int)

+ getLevel() : int

+ getXp() : int

+ getWep() : Weapon

+ getShield() : Shield

+ getHelm() : Helm

+ getBody() : Body

+ getPants() : Pants

+ addXp(int) : void

+ setWep(Weapon) : Weapon

+ setShield(Shield) : Shield

+ setHelm(Helm) : Helm

+ setBody(Body) : Body

+ setPants(Pants) : Pants

**Class Monster extends Creature**

**Class Dungeon:**

- Monster m

- int level

+ Dungeon()

+ Dungeon(int)

+ ~Dungeon()

+ nextLevel() : void

+ combat(Player\*) : int

+ generateMonster(int) : void

**Class World:**

- \*p : Player

+ World();

+ ~World();

+ mainMenu() : void

+ loadedMenu() : void

+ enterDung(int) : void

+ save() : void

+ load() : void

## Other Notes:

**Monster Generation:**

Dungeon level determines stats by randomly assigning the number of stat points that would be available at that level randomly to attack, defense, and health. A minimum amount of attack defense and health.

**Leveling:**

Every level the player will be able to allocate ?? skills point(s) in health, attack, or defense