**Ideas on how to implement REQs 3-4**

New package entities/enemies? Under game package?

Just for organisation purposes

**REQUIREMENT 3 ( Enemies )**

*GENERALLY FOR ENEMIES*

General behaviours and features will be as follows :

- Wanders around and tries to attack player within a certain range/distance, due to "HOSTILE\_TO\_ENEMY" Status of the player

- When the player attacks them or they attack the player, behaviour switches to make them follow the player

- All enemies cannot enter areas designated as the Floor ( \_ ), there has to be a checking system to see if the Ground is Floor or not. Shouldn't allow Goomba actor in if so.

- can be done by implementing canActorEnter into Floor class to not allow Goomba objects or Koopa objects ( make it return False for enemy actors )

- When enemies are unconcious, they are to be removed from the map. This can be directed towards Goombas only as Koopas have to turn dormant ( D )

- Possible implementation would be to modify the isConscious() method inherited from the Actor class and have it remove the Actor from the map before the return statement if True, and not if False

- Could also have implementation be different for the Koopa class to make logic turn Koopa dormant

- In need of spawning mechanic to spawn each, can connect with Zubin's work

- Possible spawning in Application Class??

**REQUIREMENT 3.1**

- Goomba class

- 10% suicide chance

- 10 damage, 50% chance

- 20 hp

What the class will do :

- extends Actor

- Needs an intrinsic weapon kick, 10 dmg 50% chance

- Some spawn from sprout, others dont. Some goombas spawn from sprout 10%, works with Zubin's Sprout Implementations

- 10% suicide chance

- Needs to not fully populate map

WAYS TO IMPLEMENT 10% suicide chance

- New method toSuicide() in Goomba class that is called on Goomba objects at the end of every turn. Returns True/False based on random int generator between 1 and 100, <= 10 means suicide by removing actor from map, > 10 no suicide

- New logic can be put into isConscious to have a random number generator to determine removal of Goomba. ( Highly likely to have clashing though )

- New suicide action class which works similarly to the toSuicide() method. Could have Goomba object as attribute and have methods to return true or false to determine removal

- New suicide/Goomba handler class which could contain ArrayList of all instantiated Goomba objects, similarly has methods to check if Goomba should be removed if it is conscious

- Handler class can also have methods to control Goomba population

- When Goomba ArrayList size > maxPopulation number, have a random number generated in range of ArrayList size and remove Goomba object on that index from the map

- Cannot have entire map as goomba, maybe have an int attribute representing goomba count to stop spawns

- Could implement a new Utils class to hold all different number generators for all rates and percentages

**NEW CLASSES NEEDED FOR REQ 3.1**

**-** SUICIDEHANDLER / SUICIDEACTION

**REQUIREMENT 3.2**

- Create a new Koopa class

- Start actor with 100 hp

- Actor does not get removed when unconscious, goes dormant

- Player needs wrench to destroy shell

- Destroyed shell drops mushroom

- Attack damage not given,

- Options can be to : 1. let it deal about the same damage as Goombas and have similar hit rate ( Medium )

2. Have higher damage and lower hit rate ( Seems most turtle-like )

3. Low damage and high hit rate

What the class will do :

- extends Actor

- 15% spawn chance per Tree ( T ) per turn, connect with REQ1

- Addition of new behaviour class or Status ( Capability ) for dormant

- Will change koopa display character to D

- Doesn't let koopa move

- Can possibly remove AttackAction from capabilitySet?

- If not we can change message displayed from AttackAction when attacking Koopa with dormant status to say " no effect "

- Dormant must not follow, attack or wander

- Changing status to dormant can be achieved through use of isConscious method, make logic to change status when False for example

- Cannot attack Dormant koopas without wrench

- Have to add new action class, destroyShell either that .... or some new logic / code in AttackAction class ( possible but unlikely )

- Allows player to destroy shell if player has wrench in inventory

- Removing Koopa Object/Actor from the map

- Destruction of shell will drop SuperMushroom object, can use factory method for example, to spawn SuperMushroom instance

**NEW CLASSES NEEDED FOR REQ3.2**

- KOOPA CLASS

- WRENCH CLASS

- DESTROYSHELL CLASS

- DORMANT BEHAVIOUR CLASS

- SUPERMUSHROOM CLASS

- REQUIRES VANESSA'S WORK AS WELL

- TRADING, COIN, WALLET AND TOAD TO BUY THE WRENCH ITEM

**REQUIREMENT 4 ( Magical Items )**

GENERAL BEHAVIOURS AND FEATURES FOR MAGICAL ITEMS

- Need a built-in method for consuming item or a consumingItem action class

- Built-in method can be coded in Player actor class or Item class itself

- Have to remove item from inventory at end of consumption code

- Can be picked up and stored in inventory by playyer

- Can be bought from Toad via Trading

- Both magical items to be spawned and instantiated with Player

**REQUIREMENT 4.1**

- Create a new superMushroom class

- increase max HP of player by 50

- changes display character to ( from m to M )

- extends items

- implements capable

- 100% jump rate

What the class will do :

- could have an attribute of player actor class

- to make use of built-in player method to change maxHP, increaseMaxHp()

- same thing to change display character, setDisplayChar()

- have an attribute as a flag to determine whether item consumed or not

- Consume() method can contain code to change maxhealth and displayChar, removes itself from player inventory toward end of code. Either remove item from inventory or drop item. Dropped items could be removed from map by cleaner checking isConsumed() which returns true or false.

- Set player class to have a different status, can use TALL or new SUPERMUSHROOM for implementation of new player abilities

- Could also use capable, e.g player.addCapability( Status.TALL )

IMPLEMENTATION FOR SUPERMUSHROOM BEHAVIOURS

- While status is TALL/SUPERMSHRM, jumpAction() class ( ZUBIN ) could have a checker for player status. 100% jumprate if TALL/SPRMSHRM

- Could be implemented with new superJumpAction() class as well but not very efficient / suitable

- attackAction() to have new logic to check whether target actor.hascapability(Status.TALL/SUPERMUSHROOM) or check status of player. Will change display character if TALL and remove capability, continue as normal if else

**NEW CLASSES NEEDED FOR REQ4.1**

- SUPERMUSHROOM CLASS

- JUMPACTION CLASS

- CONSUMEACTION CLASS ( possible but not necessary )

**REQUIREMENT 4.2**

- Create a new powerStar class

- heal player by 200 points ( not increase MAXHP !! )

- Immunity

- Insta-kill attacks

- No need to jump

- Lasts 10 turns when activated, stays on map for 10 turns

- extends Items

- implements Capable

What the class will do :

- create new action classes and add them to the allowable actions list of the POWERSTAR item for all required attributes and features

- uses in-built methods from items, tick() and some form of counter to check if up to 10 turns have passed, displays description of how many turns left until the star buff runs out, another display method to say "Mario is INVINCIBLE"

- A tick() method for when star is on floor, inventory and when consumed

- or just one tick() method that checks where item is and performs logic after

- needs to remove Status.STARBUFF once consumed, either via tick or another method to keep track of the PowerStar

ATTACKING AND IMMUNITY

- attackAction also might need new if statement to check if STARBUFF status is present in target of attack ( Player ).

- no dmg dealt if so

- possible new action class instaKill() that is used when status STARBUFF is active

- attackAction to check if STARBUFF status is present in attacker ( Player ).

- dmg dealt = HP of enemy is so

HOW TO IMPLEMENT DESTROYING GROUND

- new action class destroyGround(), should work similarly to the jump action but changes all Trees and Walls into ground and drops coins. Makes use of wallet class and coin class ( VANESSA ). Can be added to allowableActions of item

- another possible implementation would be to see if canActorEnter is false, then a destroyGround action could be automatically carried out

- yet another possible implementation is refactoring canActorEnter to add logic to check for STARBUFF status in player, lets destroyGround() action occur after or changes ground via built-in codes

- to be used on Trees and Walls classes

- needs a factory method or constructor to create coin objects once ground is destroyed for player to pick up

**NEW CLASSES NEEDED FOR REQ4.2**

**-** STARATTACKACTION CLASS

-DESTROYGROUNDACTION CLASS

- COIN CLASS ( vanessa )

- WALLET CLASS

- TOAD/TRADING

**-** Possible TICKCOUNTER CLASS

-

- new class powerStar

- needs attributes

- ...

- extends items

- uses in-built methods from items, tick() and some form of counter to check if up to 10 turns have passed

- code a new display() or description method to display how many ticks passed or how many turns left until star buff runs out, can have another one for Mario is INVINCIBLE or use same method to do two things

- another possible implementation is to add another if statement for player actor to see if STARBUFF status active

- new action class destroyGround(), possible to reuse old code canActorEnter, to see if false, then call destroyGround() action

- can also refactor canActorEnter to see if player has STARBUFF status, lets destroyGround() action occur after

- use on Trees and Walls classes

- can also use ActionList with Status and more types of Actions like NoEffectAttackAction when Status.STARBUFF

- add to actions list

- needs coin Class from Vanessa's implementations

- possibly Wallet Class to add balance

- destroyGround() most like drops coins onto ground through some method or itself. Could just automatically add to wallet straight but required implementation most likely wants drop.

- stored in player Inventory (under actor class)

- instantiated with Player, spawns next to player

- connects to Toad class, or any trading class (Vanessa)

NOTES AND THOUGHTS

Give specific capabilities of each Item based on Status buff from Enum.

canPlayerDestroy(), return player.hasCapability( enum.STATUS )

Add new actions

ATTACK ACTIons to be linked with attack behaviour??

- New attack actions can be linked to attack behaviour??

Add new behaviours?

How are actions added? How are they read?

Actions added based on what

Actions added where

After u attack or they attack, follows player. Dead removed from map, cannot go in Floor

- update floor with canActorEnter() logic to not allow enemies

- possibly enemy status?

- check status for enemy then player cannot enter\*

- possible spawning method for enemies?

- spawning action?

- isUnconscious() method? returns True and removes Actor from map right before return statement

- doesnt remove from map if returns False

- Slightly changed for koopa class

- called on every goomba at end of every turn

- returns true or false based of random int generator between 1 and 100, <= 10 suicide by removing actor from map, > 10 not

- some goombas spawn from sprout 10%, works with Zubin's Sprout Implementations

- could also have arraylist attribute containing goomba objects, size() cannot pass a limit (similar as above)

for magical items, either in-built method() for consuming item or consumingItem() action

isConsumed() method, check whether to remove from inventory. Remove from inventory at end of consume code

- extends items

- use in-built player method to change max hp, increaseMaxHp()

- in-built player method to change displayChar, setDisplayChar()

- attribute to flag if consumed??

- consume() method to make changes to maxhealth and displayChar when called, removes itself from player inventory after by removing from player arrayList or dropping item? dropped items could also have isConsumed boolean for a cleaner to check all items on floor to destroy/remove

- set player to have SUPERMUSHROOM status or reuse TALL for implementation or player.addcapability

- while this status is active, jumpaction() has 100% success rate (jump action should have if statement to check if TALL/SUPERMUSHROOM)

- attackAction() to have new if statement to check whether target actor.hascapability(Status.TALL/SUPERMUSHROOM)

- changes display character is true, continues with action if false

- new jumpAction class

- checks superMushroom status, for jumping, normal jump with random number generator if no status