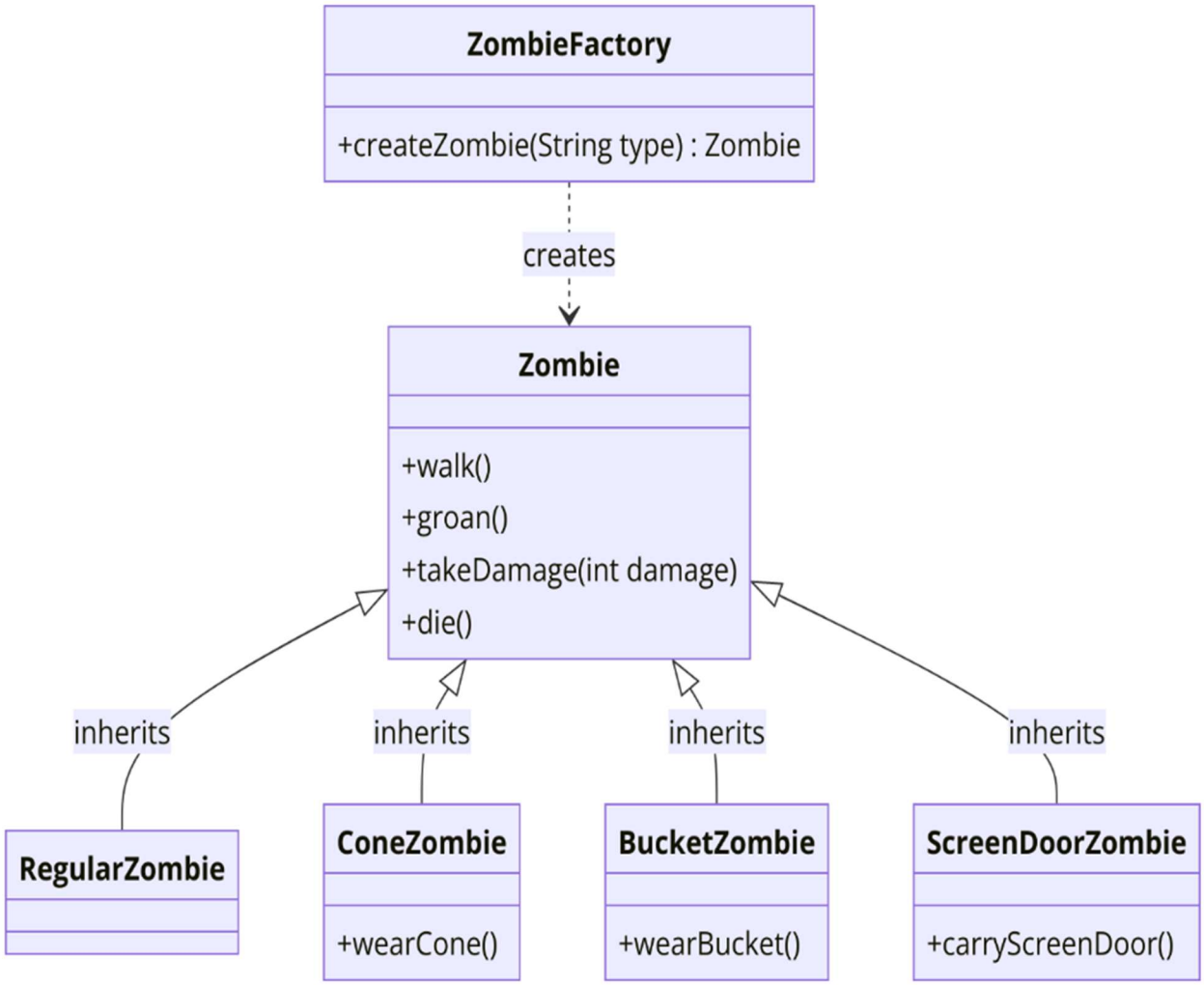
Mark Shinozaki
Cpts 487
Homework 4 – Plants vs Zombies

1. Draw a class diagram for your solution using a UML tool.
    a. Include necessary attributes and operations in your diagram
    b. Zombie -related classes MUST have takeDamage(int d) and die() operations



2. List the matchups between classes in your diagram, and the ones shown in the pattern structure:

- ZombieFactory -> Creator (Factory Method)

- Zombie -> Product (Factory Method)

- RegularZombie -> ConcreteProduct (Factory Method)

- ConeZombie -> ConcreteProduct (Factory Method)

- ScreenDoorZombie -> ConcreteProduct (Factory Method)

* I'm not exactly sure where the exact slides are but I try my best to show the match up between classes

Mark Shinozaki
Cpts 487
Homework 4 – Plants vs Zombies

4. What would change if we increased the damage of peashooter to 40, instead of 25? Does this change impact your code and/or game logic at all?

- if the damage of the peashooter is increased to 40 instead of 25, one thing that is changed is the health reduction speed: Zombies would lose health faster and that would reduce the number of rounds needed to defeat all zombies. Accessories impact, the zombies with accessories like cones or buckets, the increased damage might mean that these zombies become regular zombies more quickly because the health would drop below 50 in fewer hits. Game difficulty, the increased damage might make the game easier since zombies can be defeated more quickly.

5. What does the new feature change about your program? Does the Composite pattern still work? How this new change would impact your design and implementation. What kind of modification would be necessary?

- The zombie class and its subclass would require an additional method or an override of an existing method to account for the watermelons attack. This method would need to differentiate between the types of zombies to apply damage directly to the ScreenDoorZombie without the half damage reduction, while other zombies would take normal damage. A plant interface base class could use Attack(Zombie zombie) method and specific plants like peashooter and watermelon would implement this interface or inherit from the base class to provide their unique attack behavior.

- Composite pattern could still work if it is being used to manage the collection of game objects like zombies and plants. However since we have two types of attacks there might be a need to expand the pattern to account for these variations.

- Modifications Needed: This would require modifications to Zombie, Create a Plant hierarchy and then update the game logic to incorporate the choice of using a Peashooter or Watermelon.