Exercise 1.2 SEM

1. Composition is a relationship between two classes. One of the two classes A is the owner of the other class B. When class A doesn't exist, class B can't exist as well.
Aggregation is a relationship between two classes. One of the two classes A is the owner of the other class B. These two classes are independent of each other. So as a result of that they class B can exist even if class A doesn't exist.

The classes in our project that use aggregation are the World, WorldObject, EndlesRunner and TextureCache classes. The World contains various WorldObjects that belong in the World, but these two classes exist independently.
The EndlesRunner contains one TextureCache that belongs to the EndlesRunner. It's here also the case that these two classes exist independently
The classes in our project that use composition are World, CollisionChecker and Spawner classes. The CollisionChecker and Spawner classes are part of the World class. They can't exist independently, because without the World the CollisionChecker can't check the collisions of the World and the Spawner can't spawn objects into the World.

2. There are no parameterized classes in our source code.
Parameterized classes are used when your classes make use of inheritance and you need to be able to store the superclass or subclasses in a new certain storage class, like a LinkedList. This way you don't need to keep track of every object what kind of class it is, but the parameterized class will take care of this for you. In these cases it is better to make use of a parameterized class.

3. The only class hierarchy that exists out of more than one class is the WorlObject class hierarchy. This was done in order to be able to make different WorldObjects like the FloorTile, Obstacle and the AnimatedWorldObject classes. This way all these objects could be easily added to the World in an ArrayList of WorldObjects. Also the Player was an AnimatedWorldObject in order to make it easier for objects in the World that would ahve an animation to add that to the World.
No hierarchies should be removed, because these hierarchies are important for the game to be able to have and add objects.

UML Class Diagram