**Assignment 2 Analysis**

1. *How does your design implement the four pillars of OOP (abstraction, encapsulation, inheritance and composition, and polymorphism)?*

Our abstract class, PartyMember, defines abstract methods which are implemented in both Pokemon and Egg subclasses. This demonstrates the use of abstraction within our design.

For encapsulation, our classes implement many private methods not available to the user and variables which are accessed by getters and setters. A good example of encapsulation within our design is the \_level\_up() method in the Pokemon class. This method is not called directly by the end user, rather the Pokemon “levels up” as it gains experience (using the add\_xp() public method). This method is hidden from the public interface and is integrated into the use of the class.

Our design also incorporates inheritance, as the Pokemon and Egg classes are subclasses of PartyMember, and they inherit all of PartyMember’s properties and methods.

As for composition, we decided to take our design in a slightly different direction than expected. Our PartyManager and PartyMember are in a composition relationship, because a PartyMember is created only by PartyManager, and not independently. This means one “Party” can have many PartyMembers, and without a “Party” (A PartyManager instance), you cannot create any PartyMembers (In other words, you cannot have a party member if you do not have a party to put them in).

We incorporate polymorphism in many methods throughout our project, most notably in our PartyManager class, within the “get member” methods (get\_all\_members\_by\_elemental\_type(), get\_member\_by\_type(), etc). In these methods we construct a list of all the available members from the PC storage and the party, and then access their member\_type() method to determine that member’s type (Egg or Pokemon). Accessing this method is a demonstration of polymorphism because it is implemented in both Egg and Pokemon, and depending on the class type, returns a different value.