The OOP concepts in both Python and Java operate pretty much identically with the case of Inheritance and Polymorphism as both languages offer full support for these qualities of OOP. I was easily able to create subclasses of the different accounts using the “super” keyword to pass in variables into the parent class constructor from the child class’ constructor which allowed for easy implementation of inheritance. Additionally I was able to use 2 different methods of implementing the access modifier keywords in my implementations as I opted to use “Protected” access modifier for the Python implementation (adding _ in front of the variable) to allow for every subclass of the parent class to be able to access the variables of the parent class for each subclass’ object. For the Java implementation I used the “Private” access modifier to keep each field of the class hidden from even other subclasses of the parent class which meant I had to generate getter and setter methods to get and set the class’ fields from other subclasses. As for Polymorphism I was able to override existing methods for both implementations and even utilize the existing method (if there is one) of a parent class from a child class’ overridden method using the “super” keyword to reference the parent class’ instance of the method. Although the Abstract Data Types in Python proved easier to work with as I was able to also create an “AccountDatabase” class to store all the different accounts used in the program and it would be able to store any account type whilst in Java I would have had to individually make specific ArrayList objects for each class’ testing function to store the accounts.