1. One to many association: Line 38. Holder 15 has two accounts.

2. Data Type: My for loop iterator "i" is an integer, which is a primitive data type. My enumeration for "AccountType" is a user-defined data type.

3. Inheritance: IndividualHolder and CorporateHolder inherit from abstract class AccountHolder.

4. AccountHolder has private balance and account types, which is why I have to access them using methods.

5. AccountHolder has "balance" and "account type" properties (enumeration), which is why I have to access them using methods.

6. Line 60 has me using "GetAccountTypeOf" to find out if the accounts owned by a particular AccountHolder are "Checking" or "Savings." Not sure what else an end point could be in this context.

7. IndividualHolder and CorporateHolder both inherit from the AccountHolder class, which is an example of a dependency as well.

8. "AccountType" in the "Account" class is enumerated. In order to parse things easier, I had the "GetAccountType" method return a string ("Checking" for Checking and "Savings" for Savings).

9. The "CheckSSN" method is derived from the SSN. Since this program stores that SSN, at least one organization technically does have the SSN, so it warns the user.

10. Line 2 has a comment that explains why there are so many new instances of classes.