* **My understanding of Encapsulation**  
  Encapsulation is a fundamental principle of object-oriented programming that involves bundling the data (attributes) and methods (functions) that operate on that data into a single unit known as a class. This approach restricts direct access to some of an object's components and can prevent the accidental modification of data. Encapsulation promotes a clear separation between the internal workings of a class and how it is used externally.
* One of the key benefits of encapsulation is that it enhances data security. By controlling access to an object's data, a class can safeguard its internal state and ensure that it can only be modified through well-defined methods. This leads to fewer bugs and makes the code easier to maintain.
* Encapsulation can be applied in various scenarios, such as when creating user-defined classes. For instance, in a banking application, a BankAccount class can encapsulate the account balance and provide methods for depositing and withdrawing funds, ensuring that the balance cannot be modified directly from outside the class.
* In the C# program, encapsulation is evident in the Word, Reference, and Scripture classes. Each of these classes has private member variables, which are inaccessible from outside the class. For example, the Word class has a private variable \_isHidden that tracks whether a word is hidden. This variable can only be modified through the class's methods, like Hide() or GetDisplayText(). By encapsulating these details, we maintain control over how the data is accessed and modified, leading to more robust and error-free code.