**Explain the meaning of Encapsulation:**Encapsulation is a similar idea of abstraction where the goal was to separate the code to minimize bugs and errors. This time we separate code in different classes to minimize security risks, OR code breaking other code. Here we restrict direct access to some parts of code to ensure only necessary data is accessed or modified through defined methods.

**Highlight a benefit of Encapsulation:**A benefit of encapsulation is data security and integrity. When we restrict access to certain fields from different classes and only expose controlled interfaces (like our getters and setters) we mitigate unwanted interference. This is also useful to maintain modularity like we want when using abstraction as well.

**An application of Encapsulation:**An application of encapsulation is used for object oriented programming where you bundle attributes and methods/functions to operate on data into a single class.  
  
**The second page is a code snippet of an class called BankAccount that has details of Encapsulation.**  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
public class BankAccount

{

// Private variables, accessible only within the class

private string owner;

private decimal balance;

// Constructor to set the initial balance and owner

public BankAccount(string owner, decimal initialBalance)

{

this.owner = owner;

this.balance = initialBalance;

}

// Public method to deposit money into the account

public void Deposit(decimal amount)

{

if (amount > 0)

{

balance += amount;

Console.WriteLine($"Deposited ${amount}");

}

}

// Public method to withdraw money from the account

public bool Withdraw(decimal amount)

{

if (amount > 0 && balance >= amount)

{

balance -= amount;

Console.WriteLine($"Withdrew ${amount}");

return true;

}

Console.WriteLine("Withdrawal failed");

return false;

}

// Public method to get the current balance

public decimal GetBalance()

{

return balance;

}

}