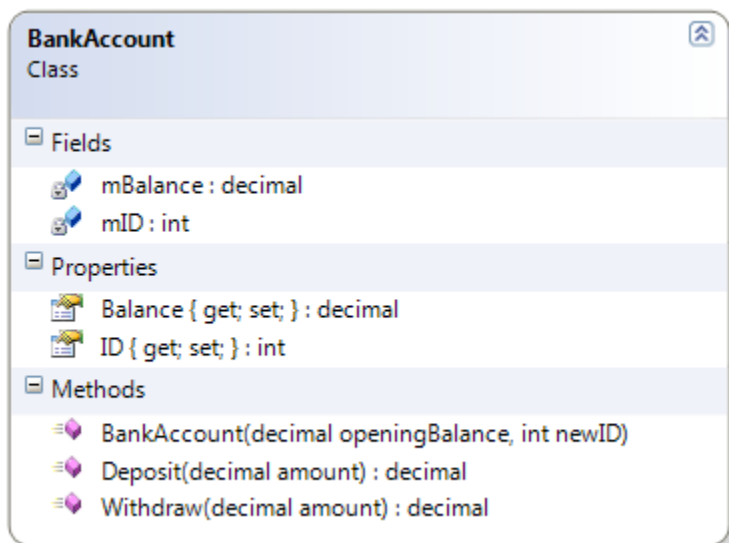


Do all the exercises in order.

1. **Class members.** Create a class **BankAccount** that represents a savings account. The class has these members:

Member	Name	Description
Property	ID	A public integer that holds the account number.
Property	Balance	A public decimal that holds the account balance.
Constructor	BankAccount(decimal openingBalance, int newID)	A public constructor that sets the Balance to openingBalance and the ID to newID.
Method	decimal Deposit(decimal amount)	A public method that adds amount to the account balance and returns the new balance.
Method	decimal Withdraw(decimal amount)	A public method that subtracts amount from the account balance and returns the new balance.

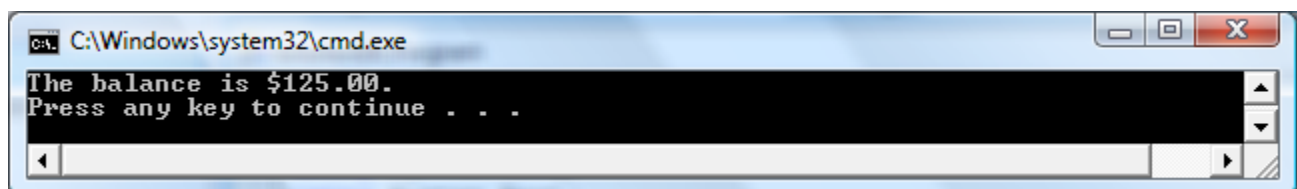
When you are done your code should compile (Ctrl+Shift+B) without error, and your class in the Class Designer should look something like this. To see the full signature as shown below, click Class Diagram on the menu. Then click Change Members Format and then Display Full Signature.



2. **Create an account and execute one transaction.**

- Create one instance of the BankAccount class, with an opening balance of 100 and an ID of 123.
- Add 25 to the account by calling the Deposit method.
- Write out the new balance. If you want to include a \$ character in the balance, the format specifier is `{0:c}`.

The application should look something like this when you run it:



3. Create an array of instances of BankAccount.

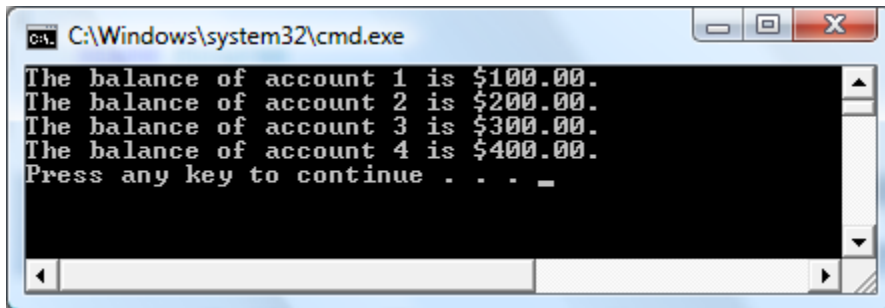
- Create four instances of the BankAccount class with these values:

ID	Balance
1	100
2	200
3	300
4	400

- Create an array of length 4 and of type BankAccount. (We did this in week 3 using the URL class.)
- Assign the four instances as the four elements in the array. (See slides from week 3.)
- Write a for loop to write the ID and Balance properties for each instance. The code inside the loop will look something like this.

```
Console.WriteLine("The balance of account {0} is {1:c}.",  
    accounts[i].ID, accounts[i].Balance);
```

The application should look something like this when you run it:



4. List<T>. (Optional extra challenge problem) Code the bank account exercise using List<T>. No hints. ☺