

Both classes `DodgyBankAccount` and `SecureBankAccount` are designed to perform the same functionality working with bank account:

`DepositAmount(int amount)` – add money to the account

`DisplayAccountBalanceDetails()` – print the balance

`DebitAmount(int amount)` – take money from the account

`AddReward()` – get a reward after depositing money

Problems with `DodgyBankAccount`:

- All properties are public

```
public string AccountNumber;  
public int AccountBalance;  
public int RewardAmount = 50;
```

This allow to change them from outside the class:

```
myDodgyBankAccount.AccountBalance = 1000000;
```

- The method `AddReward()` is also public

```
public void AddReward()
```

So, it can be misused from outside the class, for example called several times.

```
myDodgyBankAccount.AddReward();
```

```
myDodgyBankAccount.AddReward();
```

```
myDodgyBankAccount.AddReward();
```

Also if the business logic is that account is getting a reward after the depositing money, that should not be possible to call `AddReward()` without calling method `DepositAmount`.

In contract, `SecureBankAccount` has all properties and `AddReward()` method private, `AddReward()` method is called from `DepositAmount(int amount)`, therefore it is more secure – properties cannot be changed from outside the class and reward cannot be added without depositing money.