Solução ao Pedido 4:

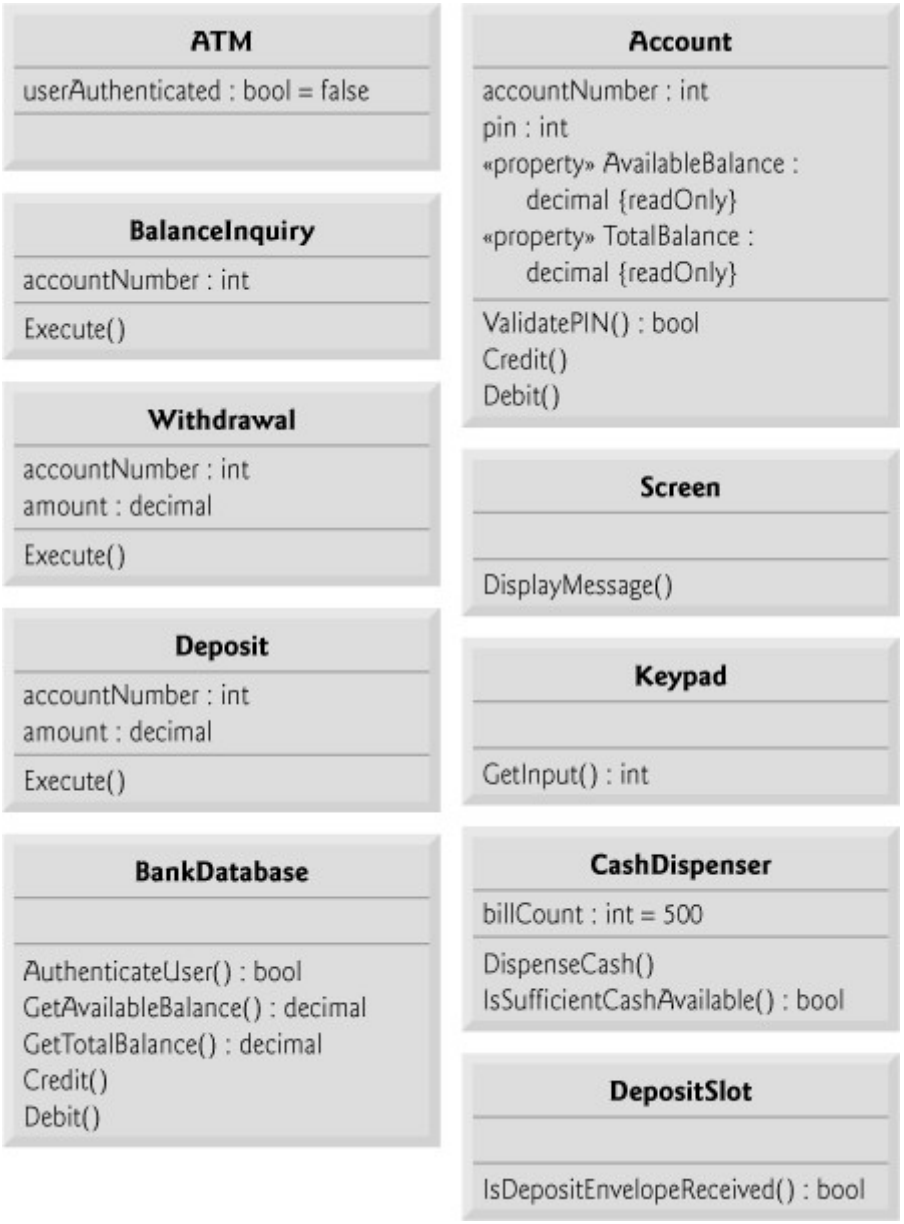


Figura 12.17

Colaboração entre objetos:

Quando dois objetos se comunicam para realizar uma tarefa, diz-se que eles são **colaboradores** — objetos fazem isso invocando as operações um do outro.
Uma **colaboração** consiste em um objeto de uma classe enviar uma **mensagem** a um objeto de outra classe.
As mensagens são enviadas no Java via chamadas de método.
Identifica as colaborações no sistema lendo cuidadosamente as seções do documento de requisitos que especificam o que o ATM deve fazer para autenticar um usuário e realizar cada tipo de transação.
Para cada ação ou passo descrito, decide quais objetos em nosso sistema devem interagir para alcançar o resultado desejado.
Identificamos um objeto como o objeto emissor e outro como o objeto receptor.

Então selecionamos uma das operações do objeto receptor (identificadas na Seção 12.6) que devem ser invocadas pelo objeto emissor para produzir o comportamento adequado.

Um objeto da classe...	envia a mensagem...	para um objeto de classe...
ATM	displayMessage	Screen
	getInput	Keypad
	authenticateUser	BankDatabase
	execute	BalanceInquiry
	execute	Withdrawal
	execute	Deposit
BalanceInquiry	getAvailableBalance	BankDatabase
	getTotalBalance	BankDatabase
	displayMessage	Screen
Withdrawal	displayMessage	Screen
	getInput	Keypad
	getAvailableBalance	BankDatabase
	isSufficientCashAvailable	CashDispenser
	debit	BankDatabase
	dispenseCash	CashDispenser

Figura 12.22 | Colaborações no sistema ATM. (Parte 1 de 2.)

Um objeto da classe...	envia a mensagem...	para um objeto de classe...
Deposit	displayMessage	Screen
	getInput	Keypad
	isEnvelopeReceived	DepositSlot
	credit	BankDatabase
BankDatabase	validatePIN	Account
	getAvailableBalance	Account
	getTotalBalance	Account
	debit	Account
	credit	Account

Figura 12.22 | Colaborações no sistema ATM. (Parte 2 de 2.)

A UML fornece vários tipos de **diagramas de interação** que modelam o comportamento de um sistema modelando a maneira como os objetos interagem.
O **diagrama de comunicação** enfatiza *quais* objetos participam das colaborações.
Como o diagrama de comunicação, o **diagrama de sequência** mostra as colaborações entre objetos, mas enfatiza *quando* as mensagens são enviadas entre objetos ao longo do tempo.

Começando a implementação:

Visibilidade

Os modificadores de acesso determinam a **visibilidade** ou acessibilidade dos atributos e métodos de um objeto a outros objetos.

Antes de iniciarmos a implementação do nosso projeto, devemos considerar quais atributos e métodos das nossas classes devem ser public e quais devem ser private.

Normalmente, atributos devem ser private e os métodos invocados por clientes de uma dada classe devem ser public.

Métodos que são chamados como "métodos utilitários" apenas por outros métodos da classe normalmente devem ser private.

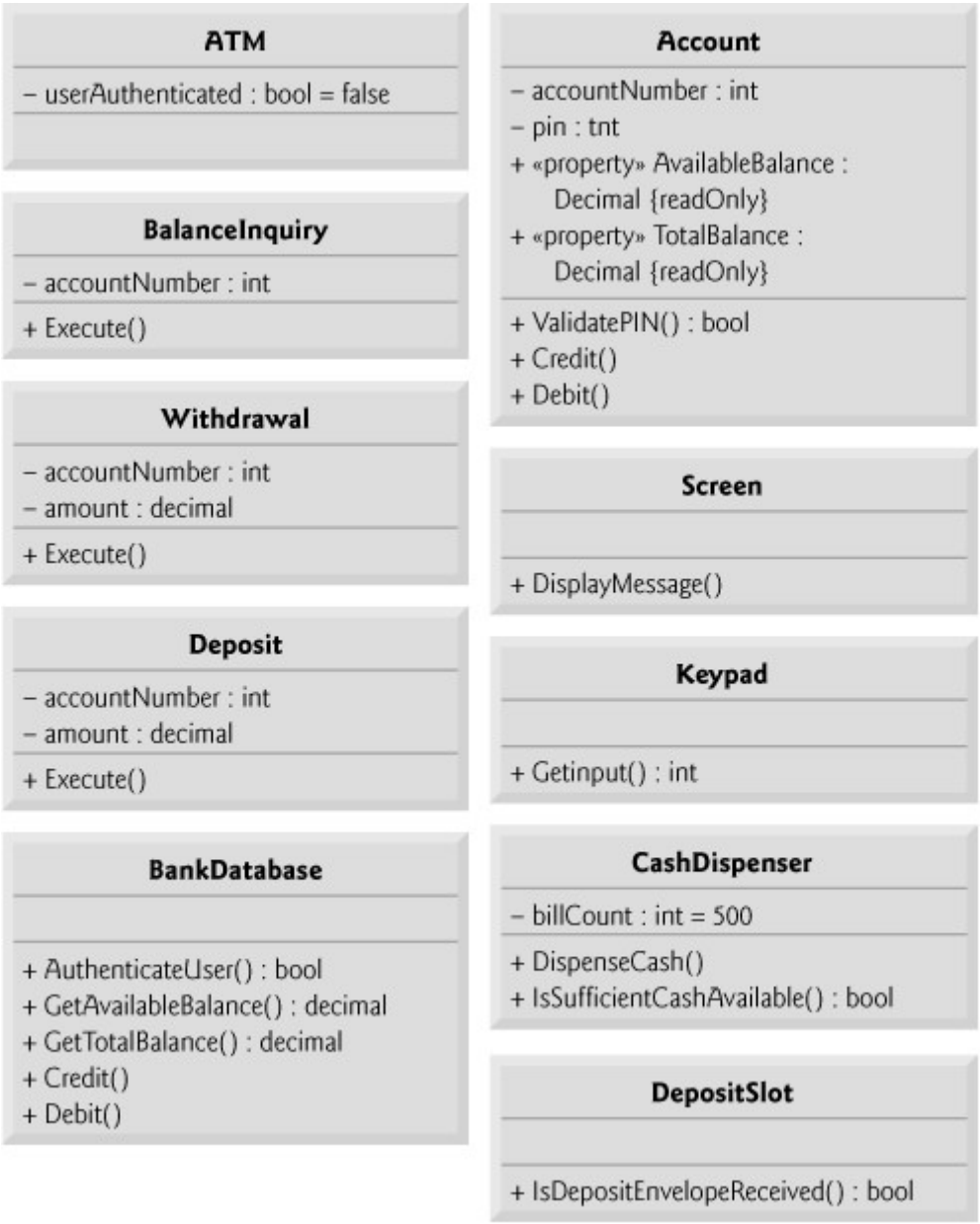


Figura 13.1

Navegabilidade:

O diagrama de classes na Figura 13.2 refina ainda mais os relacionamentos entre as classes no sistema ATM adicionando setas de navegabilidade às linhas de associação.

Setas de navegabilidade

- representadas como setas no diagrama de classe
- indicam a direção em que uma associação pode ser percorrida.

Os programadores utilizam setas de navegabilidade para determinar quais objetos precisam de referências a outros objetos.

As associações que têm setas de navegabilidade nas duas extremidades ou que não têm absolutamente nenhuma indicam **navegabilidade bidirecional** — a navegação pode prosseguir em qualquer direção pela associação.

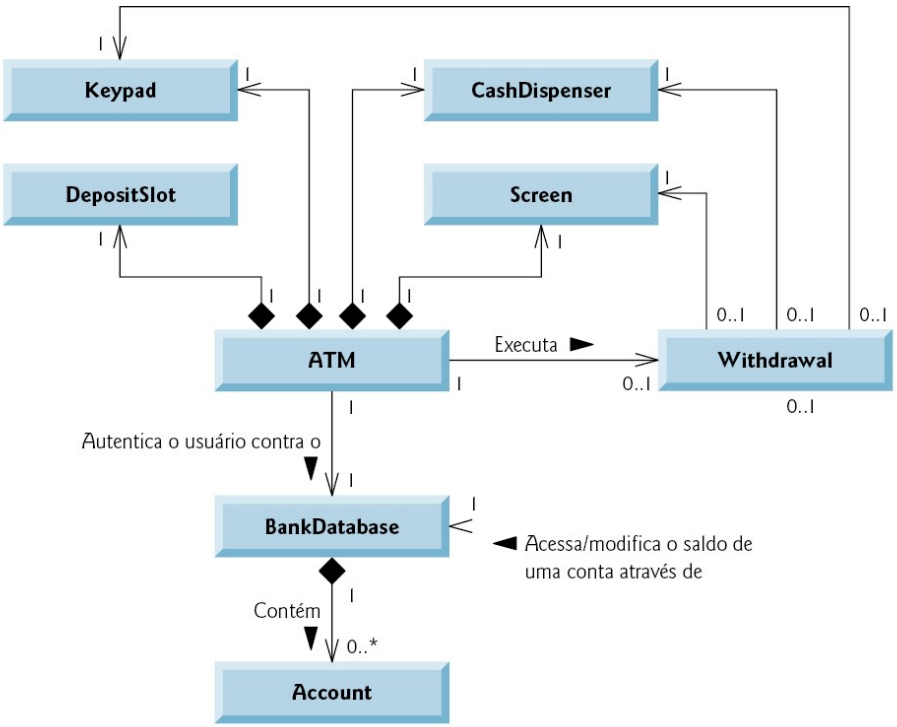
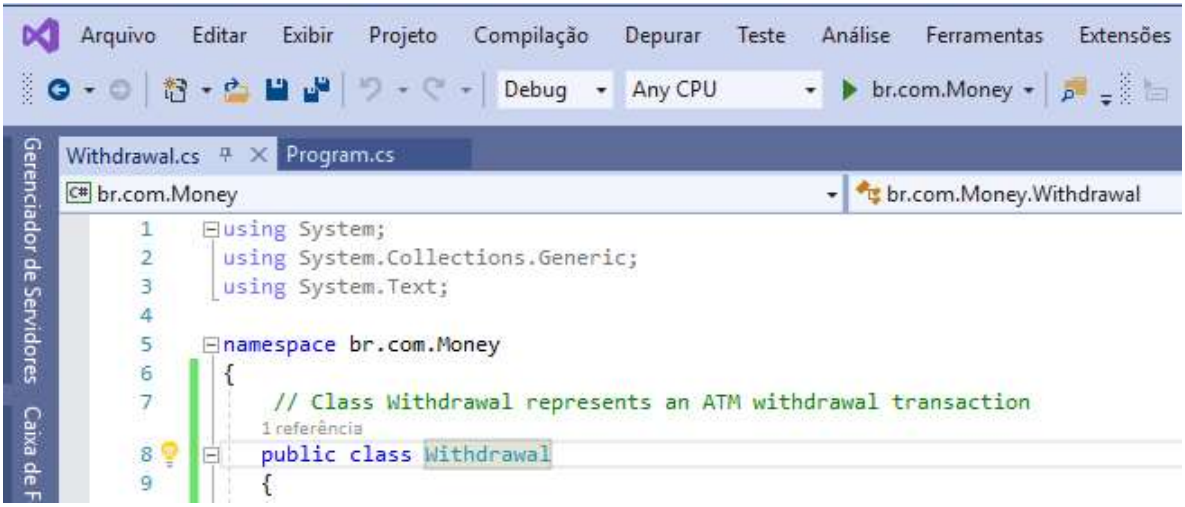


Figura 13.2 | Diagrama de classe com setas de navegabilidade.

Pedido 5:

Crie um novo projeto do tipo console, conforme a figura abaixo e programe a classe Withdrawal.



Tempo para realizar a tarefa: 15 minutos.