**QUESTÃO 01:**

**A) A operação de verificação de envelope. Esta operação é feita apenas pelo bancário através da interface da Agência. Cada depósito feito pelo ATM é marcado como PENDENTE e gera uma pendência associada à conta corrente de depósito. Neste caso, o valor depositado só será de fato adicionado ao saldo da conta após verificação pelo funcionário do banco. Esta pendência deve aparecer no extrato da conta com o rótulo “Crédito não autorizado”. O funcionário, por sua vez, ao se logar no sistema, recebe um aviso de pendências de verificação existentes e deve ser capaz de visualizar todas as pendências. Deve ser disponibilizada uma opção “Verificar depósitos” no menu do funcionário. Nesta opção, o funcionário entra com o número do envelope e recebe os dados do depósito registrado (conta, agência e valor), podendo confirmar ou não a operação. Em caso de confirmação, o saldo da respectiva conta corrente deve ser alterado e a transação passa a aparecer no extrato como um crédito (seu status muda para CONFIRMADO). Caso contrário, a transação é marcada como REJEITADA e aparece no extrato como “Crédito não autorizado”, mas sai da lista de pendências da conta (e da agência/funcionário).**

Para essa operação, criamos duas classes, a classe Pendency, subclasse de Transaction, e a classe State, um enum para indicar os states de uma pendency. A classe Pendency contém um atributo, pendingTransaction correspondente à transação pendente. Contém também dois métodos abstratos, approve e reject, que representam as ações que um Pendency deve tomar ao ser aprovada ou rejeita, esses métodos se estendem à EnvelopeDeposit. Os métodos são abstratos, pois Pendency é uma generalização, e cada transação que pode ser pendente deve implementar uma ação diferente para cada situaçao. Além disso, Pendency contém métodos que relacionam State e Pendency a partir de getters e setters.

Na base de dados, colocamos uma lista de pendências que contém todas as pendências não resolvidas. Sempre que uma pendência é criada, é adicionada à lista, e sempre que é resolvida, é retirada da lista.

Já em AccountManagementService, adicionamos dois métodos: getAllPendencies e CheckPendencies. O primeiro responsável por pegar todas as pendências retornando-as em uma lista, o segundo atribui às pendências seus respectivos novos estados.

Criamos também outra ação: CheckPendenciesAction, mostra todas as pendências não resolvidas em uma tabela ao funcionário. O funcionário pode alterar o estado de qualquer pendência para aceitada, rejeitada, ou mantê-la pendente. Se o funcionário confiamar as mudanças, os novos estados são aplicados e as ações das pendências são efetuadas, e então as pendências resolvidas, aceitadas ou rejeitadas, são retiradas da lista de pendências

**B) A operação de depósito direto no caixa. Deve ser disponibilizada uma opção “Depósito direto” no menu do funcionário. Nesta opção, o funcionário entra com os dados do depósito (agência, conta e valor) e efetua a transação. O saldo da respectiva conta corrente deve ser alterado imediatamente e a transação passa a aparecer no extrato como um crédito.**

Primeiramente, criamos uma nova classe, chamada EnvelopeDeposit, para representar um depósito com envelope. Com isso, utilizamos a classe Deposit para representar um depósito direto. Também criamos uma nova operação de conta em AccountOperationService para lidar com os dois tipos diferentes de depósito. Como foi necessário diferenciar os dois tipos na operação, também criamos dois tipos de depósito na conta corrente, que se diferenciam pelo fato de um criar um Deposit, enquanto outro cria um EnvelopeDeposit, além do segundo não debitar o valor imediatamente.

Além disso, modificamos a interface do programa, criando uma nova opção para o funcionário escolher entre Depósito Direto ou não. Caso queira utilizar o depósito que utiliza um envelope, o programa pedirá o número do envelope o valor desejado, número da conta e sua agência e, caso queira um depósito direto, o envelope não será mais necessário.

**QUESTÃO 02:**

**Inclusão da cobrança de um Imposto sobre Transações Financeiras (ITF) no valor de 0,2% da transação. Este valor deve ser cobrado sempre que houver “circulação escritural ou física de moeda, e de que resulte ou não transferência da titularidade dos mesmos valores, créditos e direitos”1 . Ou seja, qualquer operação que gere débito na conta deve prever esta taxação. Este débito deve aparecer no extrato explicitamente para cada operação que o resultou.**

Para implementar a Inclusão da cobrança de um Imposto, criamos uma classe TaxedTransaction, subclasse de Transaction. Essa nova classe contém dois atributos: um tax público estático do tipo double, correspondente à porcentagem de imposto cobrada sobre uma transação, e outro tax private do tipo double, correspondente à porcentagem de importo cobrada sobre a transação especificada. Ela também inclui um método do tipo getTax, que retorna o double correspondente à taxa.