**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO**

**LUIGI WASCHENSHIKY LUZ**

# SISTEMA ATM

**CAMPOS DO JORDÃO**

**2024**

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO**

# SISTEMA ATM

Entrega do projeto de Desenvolvimento de um sistema em C++ da disciplina de Programação Orientada a Objetos(POO) apresentado ao Instituto Federal de São Paulo (IFSP), em cumprimento a exigência da disciplina de POO, do curso de Análise e Desenvolvimento de Sistemas.

**PROFESSOR: Paulo Giovani de Faria Zeferine**

**CAMPOS DO JORDÃO**

**2024**

# RESUMO

O presente trabalho apresenta o desenvolvimento de um sistema de caixa eletrônico (ATM) com o objetivo de aplicar e reforçar conceitos de programação orientada a objetos (POO) utilizando a linguagem C++. A construção do sistema seguiu as diretrizes propostas pelo livro "Como Programar: C++" de Deitel, que serviu como base teórica e prática para a implementação. A metodologia empregada foi tanto bibliográfica quanto descritiva, possibilitando o planejamento detalhado e a documentação de cada etapa do desenvolvimento. Para modelar e organizar a estrutura do sistema, foi utilizado um diagrama de classes UML, criado no LucidChart, que orientou a implementação de funcionalidades como autenticação de usuários, consulta de saldo, saques, depósitos e saída do sistema. O sistema foi desenvolvido na IDE CLion, escolhida por suas ferramentas avançadas e suporte robusto à linguagem C++, garantindo controle eficiente de memória e modularidade no código. Os resultados obtidos foram satisfatórios, com o sistema simulando de forma eficaz as interações entre o usuário e o caixa eletrônico, atendendo aos requisitos propostos. Conclui-se que o projeto atendeu aos objetivos de aprendizagem e prática de POO, oferecendo uma base sólida para futuros aprimoramentos. Entre as sugestões de melhorias estão a implementação de funcionalidades para criação dinâmica de contas, integração de contas poupança e corrente, e a possibilidade de transferências entre contas.

**Palavras-Chave**: Programação Orientada a Objetos; C++; Caixa Eletrônico; ATM; Sistema Bancário.

# ABSTRACT

This paper presents the development of an Automated Teller Machine (ATM) system, aiming to apply and reinforce object-oriented programming (OOP) concepts using the C++ language. The system's construction followed the guidelines outlined in the book *"How to Program: C++"* by Deitel, which provided both theoretical and practical foundations for implementation. The methodologies employed included bibliographic and descriptive approaches, enabling detailed planning and documentation of each development stage. To model and organize the system's structure, a UML class diagram was created using LucidChart, guiding the implementation of functionalities such as user authentication, balance inquiries, withdrawals, deposits, and system exit.
The system was developed using the CLion IDE, chosen for its advanced features and robust support for C++, ensuring efficient memory management and modularity in the code. The results achieved were satisfactory, with the system effectively simulating user interactions with the ATM and meeting the proposed requirements. It is concluded that the project met its learning and practical objectives in OOP, providing a solid foundation for future enhancements. Suggested improvements include the dynamic creation of user accounts, the integration of savings and checking accounts with unique functionalities, and the implementation of account-to-account transfers.



**Keywords**: Object-Oriented Programming; C++; Automated Teller Machine; ATM; Banking System.

# LISTA DE ILUSTRAÇÕES

# LISTA DE SIGLAS

| | |
|---|---|
| **IFSP** | Instituto Federal de Educação, Ciência e Tecnologia de São Paulo |
| **POO** | *Programação Orientada a Objetos* |
| **VSCODE** | Visual Studio Code |
| **UML** | Linguagem de Modelagem Unificada |

# SUMÁRIO

# 1 INTRODUÇÃO

Os sistemas de caixas eletrônicos (ATMs) são ferramentas indispensáveis no setor bancário, possibilitando a realização de transações financeiras de maneira rápida, segura e acessível. Com o avanço da tecnologia, o desenvolvimento de sistemas ATM tornou-se uma prática essencial para programadores, permitindo o aprendizado de conceitos fundamentais de programação orientada a objetos.
Este trabalho tem como objetivo implementar um sistema ATM baseado no modelo descrito no livro C++: Como Programar, de Deitel & Deitel, com foco na aplicação de conceitos como herança, encapsulamento e polimorfismo. A implementação busca reproduzir fielmente as funcionalidades propostas, utilizando a linguagem de programação C++.

A relevância deste trabalho reside na sua capacidade de promover a compreensão prática dos conceitos teóricos abordados, consolidando conhecimentos essenciais para o desenvolvimento de sistemas.

## 1.1 Objetivos

Objetivo geral é documentar e desenvolver um sistema ATM baseado no modelo descrito no livro C++*:* Como Programar, de Deitel & Deitel, utilizando a linguagem C++ e aplicando conceitos da programação orientada a objetos, com o propósito de aplicar conhecimentos teóricos adquiridos.

Para a consecução deste objetivo foram estabelecidos os objetivos específicos:

- **Aprimorar conhecimentos em Programação Orientada a Objetos (POO):** Aplicar conceitos como herança, polimorfismo, encapsulamento e abstração na implementação de um sistema funcional.

- **Desenvolver habilidades em estruturação de código:** Criar um código modular, utilizando boas práticas de programação, como separação de responsabilidades e reutilização de componentes.

- **Aprender a trabalhar com diagramas de classes:** Interpretar e criar diagramas de classes para representar a estrutura e as relações entre os componentes do sistema.

- **Simular funcionalidades de um ATM real:** Implementar operações básicas como saque, depósito, consulta de saldo e transferência entre contas.

- **Praticar o uso de bibliotecas padrão do C++:** Utilizar recursos como manipulação de strings, números aleatórios e controle de fluxo para atender às necessidades do projeto.

## 1.2 Justificativa

O desenvolvimento de um sistema ATM como projeto acadêmico é altamente relevante para o estudo e aplicação de conceitos fundamentais da Programação Orientada a Objetos (POO). O ATM é um exemplo prático e direto que permite abstrair de maneira clara e simples as relações entre diferentes classes e seus comportamentos.

Um ATM está diretamente ligado a um banco, o que facilita a identificação de suas funcionalidades principais, como gerenciamento de contas e realização de operações financeiras, incluindo depósitos, saques, transferências e consultas de saldo. Essas funcionalidades, por sua vez, estão diretamente relacionadas aos conceitos básicos de POO, como o:

**Encapsulamento:** Proteger os dados das contas.

**Herança:** Representar tipos específicos de contas, como conta corrente ou poupança.

**Polimorfismo:** Permitir que diferentes tipos de contas compartilhem métodos comuns com implementações específicas.

Um sistema ATM como objeto de estudo também se justifica por sua aplicabilidade prática. Embora simplificado, permite que os estudantes compreendam os desafios e as soluções envolvidas na implementação de sistemas reais.

## 1.3 Aspectos Metodológicos

O trabalho adotou uma abordagem metodologia que integrou duas principais metodologias: a bibliográfica e a descritiva.

## 1.4 Aporte Teórico

O presente trabalho fundamenta-se, principalmente, em uma referência teórica:

- C++: Como Programar de Deitel & Deitel

# 2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção será apresentada uma revisão de textos, artigos, livros, periódicos, enfim, todo o material pertinente à revisão da literatura que será utilizada no desenvolvimento do trabalho.

## 2.1 C++: Como programar de Deitel & Deitel

Este trabalho foi fundamentado no livro C++*:* Como Programar, de Deitel & Deitel, que apresenta um modelo detalhado e funcional para a implementação de um sistema ATM, utilizado para o desenvolvimento do projeto. O livro não apenas descreve a estrutura do sistema, mas também explora conceitos fundamentais de Programação Orientada a Objetos (POO), como encapsulamento, herança e polimorfismo, que foram aplicados de maneira prática ao longo do trabalho.

### Conceitos Utilizados no Trabalho

A **Programação Orientada a Objetos** foi o paradigma central aplicado no projeto. Entre os conceitos fundamentais, destacam-se:

- **Classes e Objetos:** A estrutura do sistema ATM foi organizada em classes principais, como ATM, Account. Cada classe foi projetada para representar uma entidade específica, com atributos e métodos relevantes.
- **Encapsulamento:** Os atributos das classes foram mantidos privados, acessados apenas por meio de métodos públicos (getters e setters), garantindo a segurança e modularidade do sistema.
- **Herança e Polimorfismo:** A herança foi utilizada para criar subclasses que estendem funcionalidades da classe base Transaction. O polimorfismo permitiu a redefinição de métodos, como execute, adaptando-os às características específicas de cada tipo de conta.

### Análise e Utilização de UML

Antes da implementação, foi realizada uma análise sobre quais classes seriam necessárias para representar adequadamente as funcionalidades de um simples sistema ATM. Essa análise inicial permitiu mapear as entidades e suas responsabilidades dentro do sistema.

Após a definição das classes, foi utilizado o UML para criar diagramas de classes que ajudaram a visualizar e organizar a estrutura do sistema. O diagrama de classes forneceu uma visão clara de como as classes se relacionariam. Esse passo foi fundamental para garantir que o sistema fosse bem estruturado, pois proporcionou um planejamento claro antes da codificação.

Esses conceitos foram essenciais para organizar o sistema, conforme orientado pelo livro de Deitel.

# 3 PROJETO PROPOSTO

Neste capítulo, será apresentado o projeto proposto, que consiste no desenvolvimento de um sistema de caixa eletrônico (ATM). O objetivo do projeto é aplicar conceitos de programação orientada a objetos (POO) em C++, utilizando metodologias e ferramentas que garantam a implementação eficiente e estruturada das funcionalidades do sistema.

Inicialmente, será detalhado o escopo do projeto, destacando as principais operações que o sistema é capaz de realizar, como depósitos, saques, transferências e consulta de saldo. Em seguida, será descrita a abordagem adotada para o desenvolvimento do sistema, enfatizando o uso de POO e a ausência de interface gráfica, o que simplifica a aplicação sem comprometer sua funcionalidade.

Ao longo deste capítulo, também será apresentado o diagrama de classes UML que serviu como base para o planejamento e organização da arquitetura do sistema, garantindo a tradução precisa dos requisitos para uma estrutura técnica bem definida.

## 3.1 Considerações Iniciais

O projeto baseou-se no livro "Como Programar: C++" de Deitel, que forneceu uma sólida base teórica para a implementação das funcionalidades do sistema.
Para a execução do projeto, foram empregadas metodologias complementares que garantiram um desenvolvimento eficiente e organizado. A metodologia bibliográfica possibilitou o estudo detalhado de POO e lógica de programação, enquanto a abordagem descritiva assegurou o registro minucioso de todas as etapas do desenvolvimento. Além disso, foram utilizadas ferramentas como a IDE CLion, pela sua robustez em projetos C++, e o LucidChart, para a construção do diagrama de classes UML, o que contribuiu significativamente para a organização e planejamento do sistema.

O projeto propôs a criação de uma aplicação que simula interações reais com um caixa eletrônico, incluindo funcionalidades como saques, depósitos, transferências e consulta de saldo. A segurança do sistema foi garantida por meio de autenticação

com número da conta e senha. Embora o sistema tenha sido projetado sem interface gráfica, ele apresenta uma arquitetura organizada e baseada nos princípios de encapsulamento, herança, polimorfismo e abstração, destacando a aplicação prática de conceitos fundamentais de POO.

## 3.2 Metodologias de Desenvolvimento

A metodologia bibliográfica foi essencial para fornecer a base teórica necessária ao desenvolvimento do sistema ATM. O livro "Como Programar: C++" de Deitel foi a principal fonte de consulta, oferecendo uma abordagem detalhada sobre programação orientada a objetos. Além disso, essa metodologia permitiu aplicar os conceitos adquiridos durante o curso, refletindo diretamente no desenvolvimento prático do projeto. O estudo do livro possibilitou uma implementação eficiente e estruturada das funcionalidades do sistema ATM.

A metodologia descritiva foi adotada ao longo de todo o desenvolvimento do sistema, com o objetivo de documentar cada etapa do projeto. Desde o início até o resultado final, todos os processos foram registrados de maneira clara e objetiva. Essa abordagem possibilita uma visão completa e detalhada de como o sistema foi construído, permitindo ao leitor entender o andamento e as decisões tomadas durante o desenvolvimento.

## 3.3 Ferramentas Utilizadas

Para o desenvolvimento deste sistema ATM, as ferramentas utilizadas foram a IDE CLion, a linguagem de programação C++, e o LucidChart para a construção do diagrama de classes em UML. Abaixo, detalham-se as funcionalidades de cada uma dessas ferramentas essenciais para o desenvolvimento do sistema.

### IDE CLion

A IDE utilizada para o desenvolvimento do sistema foi o CLion, que oferece um ambiente robusto e adequado para o desenvolvimento em C++. A escolha do CLion foi motivada por sua interface e por ser uma ferramenta projetada especificamente para programação em C e C++, com funcionalidades como depuração, verificação de sintaxe e autocompletar código. O CLion facilita o gerenciamento de projetos C++ de forma intuitiva e oferece suporte a bibliotecas externas, o que contribui para um fluxo de trabalho ágil e produtivo.

**C++**

C++ foi utilizada para o desenvolvimento do sistema, oferecendo controle eficiente de memória, o que proporciona em desempenho otimizado. Além disso, a linguagem permite a aplicação de conceitos de POO, garantindo modularidade, flexibilidade e organização no código. Esses recursos são essenciais para a criação de sistemas escaláveis e de fácil manutenção. A lógica do sistema ATM e suas interações com os dados foram implementadas em C++, utilizando POO para garantir a estruturação eficiente das funcionalidades e a manutenção de um código organizado e de fácil expansão.

**LucidChart - Diagrama de Classes UML**

Para a modelagem do sistema e a representação visual das classes e seus relacionamentos, foi utilizado o LucidChart, uma ferramenta online para criação de diagramas. A partir da análise dos requisitos e da estrutura do sistema ATM, foi criado um diagrama de classes UML, que ajudou a planejar e organizar a arquitetura do sistema antes da implementação em C++. A ferramenta permitiu uma visualização clara das classes, atributos e métodos envolvidos, facilitando a comunicação entre as partes envolvidas no desenvolvimento do sistema.

## 3.4 Descrição do Projeto

O projeto proposto consiste no desenvolvimento de um sistema de caixa eletrônico (ATM), com o objetivo de aplicar e reforçar conceitos de programação orientada a objetos (POO) e de lógica de programação em C++. O sistema foi desenvolvido com

base no livro "Como Programar: C++" de Deitel, que segue o paradigma de POO e oferece uma excelente abordagem para entender conceitos fundamentais de desenvolvimento de software.

**Escopo do Projeto:** O desenvolvimento do sistema ATM envolve a criação de uma aplicação que simula a interação de um usuário com um caixa eletrônico. O sistema abrange funcionalidades como:

- **Operações bancárias**: O usuário pode realizar operações como depósitos, saques, transferências e consulta de saldo.
- **Segurança**: O sistema exige autenticação via número da conta e senha para garantir a segurança do usuário.

A abordagem adotada busca integrar os conceitos de POO, como encapsulamento, herança, polimorfismo e abstração, proporcionando uma experiência de desenvolvimento estruturada e organizada. O sistema foi projetado para simular o funcionamento de um caixa eletrônico de maneira simples e eficiente, sem a necessidade de interface gráfica.

### 3.5 Diagrama de Classes

O Diagrama de Classes é uma ferramenta essencial na modelagem de sistemas orientados a objetos, pois permite representar a estrutura estática do sistema de forma clara e organizada. Por meio dele, é possível visualizar as classes que compõem o sistema, seus respectivos atributos, métodos e os relacionamentos entre elas.

No contexto deste projeto, o diagrama de classes foi utilizado como um instrumento fundamental para planejar e organizar a arquitetura do sistema. Essa abordagem garantiu que os requisitos do sistema fossem traduzidos de maneira precisa para uma estrutura técnica compreensível e consistente. Abaixo, está o diagrama de classes que foi utilizado como base para o desenvolvimento do sistema:

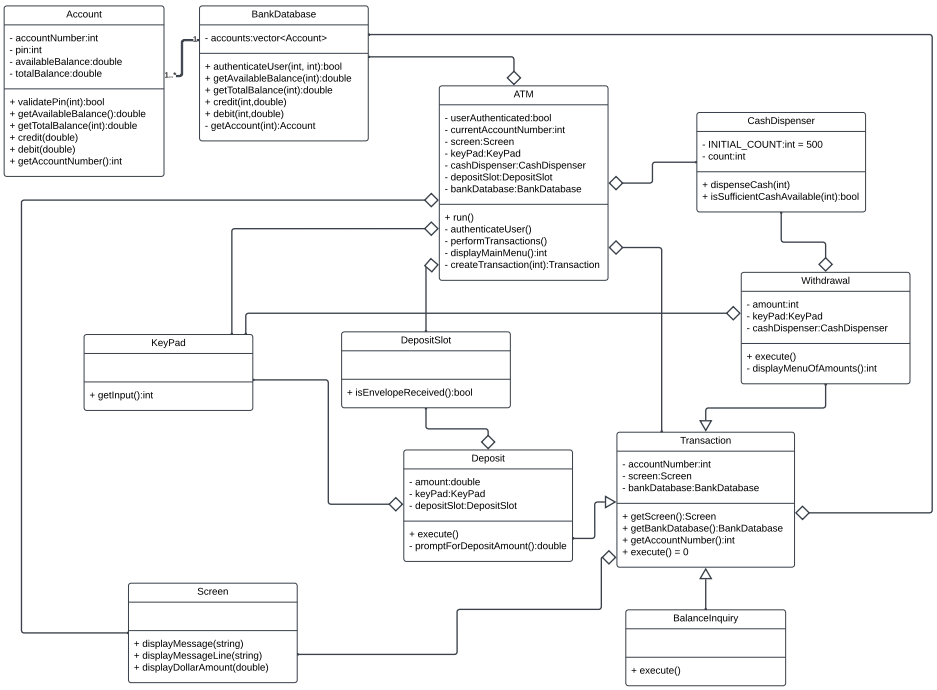


**Figura 1 –** Diagrama de Classes

A descrição do Diagrama de Classes completo está disponível no Apêndice A. Ele ilustra as principais classes do sistema, seus atributos, métodos e relacionamentos.

# 4 RESULTADOS OBTIDOS

Neste capítulo, serão apresentados os resultados obtidos durante o desenvolvimento do sistema ATM, com o objetivo de ilustrar visualmente as principais funcionalidades implementadas. As imagens a seguir demonstram o funcionamento do sistema, incluindo a interface de interação com o usuário e as operações bancárias realizadas, como saques, depósitos e consultas de saldo. Através dessas capturas de tela, é possível observar a implementação da autenticação de usuário e a realização de transações, que foram essenciais para atingir os objetivos do projeto.

## 4.1 Resultados

Nesta imagem, é possível ver a tela inicial de validação do usuário no sistema ATM. Ao iniciar, o sistema exibe uma mensagem de "Bem-vindo!", indicando que o usuário está prestes a interagir com o caixa eletrônico. Em seguida, o sistema solicita o número da conta do usuário, para verificar sua identidade. Após a inserção do número da conta, é solicitado que o usuário informe sua senha, completando a validação necessária para prosseguir com as operações bancárias.


**Figura 2 –** Tela Inicial

Nesta imagem, o sistema exibe uma mensagem de erro informando "Número da conta ou senha inválidos. Por favor, tente novamente...", após o usuário tentar inserir dados incorretos. O sistema solicita novamente que o usuário insira o número da conta e a senha para realizar uma nova tentativa de autenticação.

```
Por Favor insira o numero da conta:54321

Insira sua Senha:12345
 Numero da conta ou senha invalidos. Por favor tente novamente...

Bem-Vindo!

Por Favor insira o numero da conta:12345

Insira sua Senha:54321
```

**Figura 3 -** Conta ou Senha Inválidos

Nesta imagem, o usuário foi autenticado com sucesso e agora está no menu principal do sistema ATM. O menu oferece quatro opções principais para o usuário: Visualizar Saldo, Sacar Dinheiro, Depositar Fundos e Sair. O sistema exibe essas opções de forma clara, permitindo que o usuário selecione a operação desejada.

```
Menu principal:
1 - Visualizar saldo
2 - Sacar dinheiro
3 - Depositar fundos
4 - Sair
```

**Figura 4 -** Menu Principal do Sistema ATM

Nesta imagem, o usuário escolheu a opção Visualizar Saldo no menu principal. O sistema exibe as informações do saldo, apresentando o Saldo Disponível de $ 800,00 e o Saldo Total de $ 1.000,00.

```
Informacoes de saldo:
 - Saldo Disponivel: $800.00
 - Saldo Total: $1000.00
```

**Figura 5 -** Visualização do Saldo

Nesta imagem, o usuário selecionou a opção Sacar Dinheiro no menu principal, e o sistema apresenta as opções de saque disponíveis. As opções variam de $ 20,00 a $ 200,00, com a última opção sendo Cancelar Transação. O usuário pode escolher entre as opções 1 a 5 para realizar o saque. Caso o saldo seja suficiente para a opção escolhida, após a confirmação da transação, o sistema exibe a mensagem: "Por favor, retire seu dinheiro do caixa eletrônico.". Essa mensagem indica que a operação foi concluída com sucesso, e o usuário pode retirar o valor solicitado.

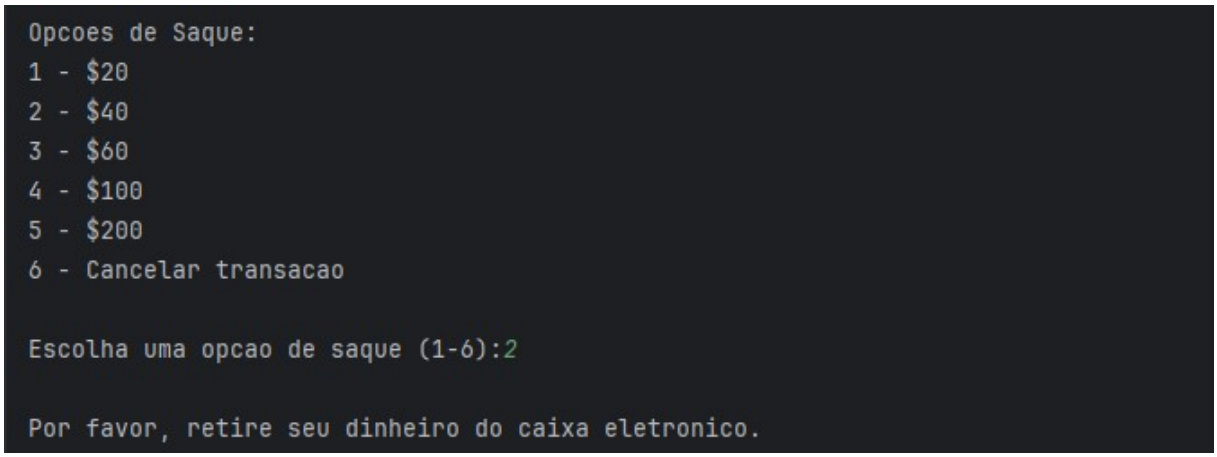


**Figura 6 –** Opções de Saque

Nesta imagem, o usuário escolheu a opção Sacar Dinheiro no menu principal, mas não possui saldo suficiente para realizar a transação. Como resultado, o sistema exibe duas mensagens consecutivas. A primeira mensagem informa o problema: "Saldo insuficiente em sua conta.". Logo em seguida, o sistema orienta o usuário com a mensagem: "Por favor, escolha um valor menor.", indicando que o usuário deve selecionar um valor de saque inferior ao disponível em sua conta.

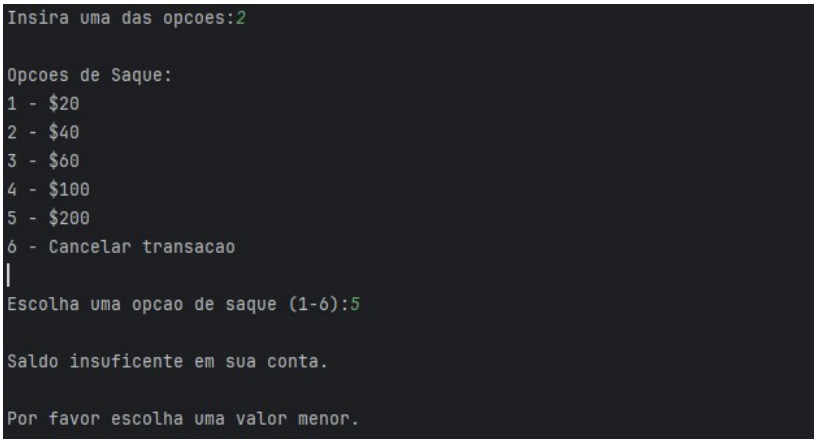


**Figura 7 –** Saldo Insuficiente para Saque

Nesta imagem, o usuário escolheu a opção Depositar Fundos no menu principal. Após selecionar esta opção, o sistema exibe a mensagem: "Por favor, insira um valor de depósito em centavos (ou 0 para cancelar):", solicitando que o usuário informe o valor desejado. O usuário inseriu o valor de 30000, equivalente a $300,00.
Em seguida, o sistema apresenta uma instrução ao usuário, exibindo a mensagem: “Por favor, insira o envelope de depósito contendo $300.00 no slot de depósito.".

Após a confirmação do depósito, o sistema finaliza com a mensagem: "Seu envelope foi recebido. NOTA: O dinheiro não está disponível até que verifique o valor de qualquer dinheiro em espécie incluso e qualquer valor incluso.".

Essas mensagens indicam que o depósito foi registrado, mas os fundos só estarão disponíveis após verificação.

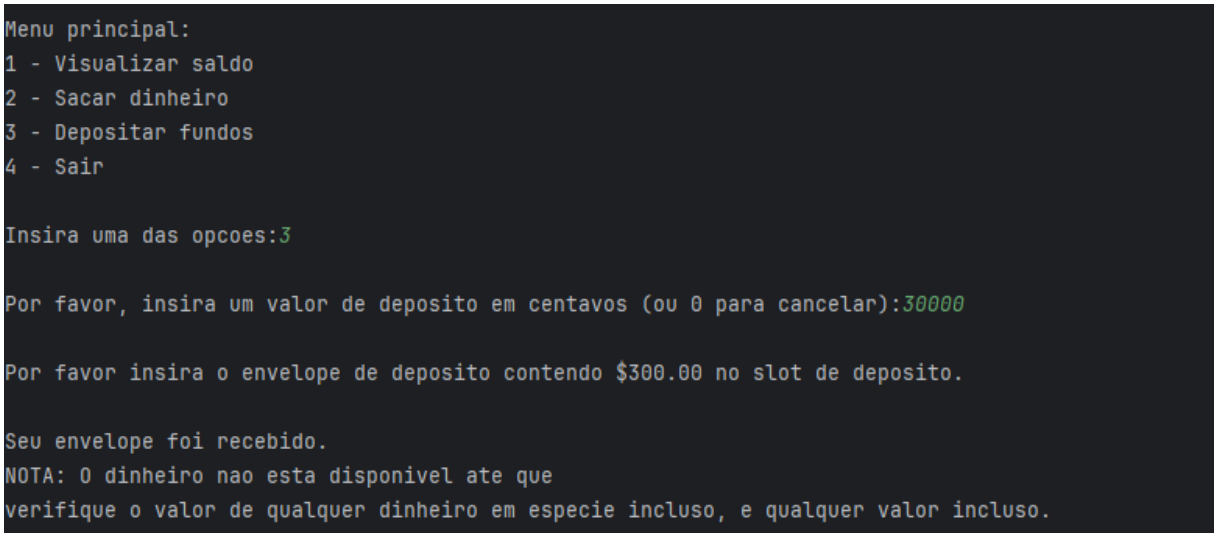


**Figura 8 –** Depósito de Fundos Realizado

Nesta imagem, o usuário selecionou a opção Sair no menu principal. Após essa escolha, o sistema apresenta a mensagem:
"Saindo do sistema...", indicando que o processo de encerramento está em andamento.
Em seguida, uma mensagem de despedida é exibida:
"Obrigado! Até mais!", concluindo a interação com o caixa eletrônico.

```
Menu principal:
1 - Visualizar saldo
2 - Sacar dinheiro
3 - Depositar fundos
4 - Sair

Insira uma das opcoes:4

Saindo do sistema...

Obrigado! Ate mais!
```

**Figura 9 –** Saída do Sistema

# 5 CONCLUSÃO

O desenvolvimento do sistema ATM permitiu a aplicação prática de conceitos de Programação Orientada a Objetos (POO) e lógica de programação, consolidando os conhecimentos adquiridos durante o estudo da linguagem C++ e do livro de Deitel.

A estrutura do sistema abrangeu funcionalidades essenciais de um caixa eletrônico, como autenticação de usuários, consulta de saldo, saques, depósitos e encerramento seguro de sessões. A implementação demonstrou a eficácia dos conceitos de POO, como encapsulamento e modularidade, para criar um sistema organizado, flexível e com potencial de expansão.

Além disso, a criação dos diagramas UML foi fundamental para o planejamento e estruturação do sistema, garantindo clareza nas relações entre as classes e facilitando a implementação. Embora o sistema tenha atendido aos objetivos iniciais, ele oferece diversas oportunidades para aprimoramentos e extensões.

## Melhorias Futuras

Embora o sistema tenha alcançado os objetivos iniciais, há várias áreas que podem ser aprimoradas:

### Criação Dinâmica de Usuários

Atualmente, o sistema exige que as contas sejam previamente instanciadas antes do início da execução. Uma melhoria significativa seria a implementação de uma funcionalidade que permita aos usuários criar novas contas dinamicamente, definindo informações como número da conta, senha e tipo de conta durante o uso do sistema.

### Contas Corrente e Poupança

A adição de contas diferenciadas, como conta corrente e conta poupança, seria um avanço relevante. Cada tipo de conta poderia ter características específicas, como rendimentos automáticos para poupanças e limites de cheque especial para contas correntes, enriquecendo a funcionalidade do sistema.

**Transferência entre Contas**

A implementação de transferências entre contas seria outra funcionalidade valiosa. Esta opção permitiria aos usuários movimentar valores entre diferentes contas de forma prática e segura, replicando uma funcionalidade comum em caixas eletrônicos reais.

# REFERÊNCIAS

## A. LIVROS:

DEITEL, H. M.; DEITEL, P. J. **C++ Como programar.** 5. ed. São Paulo: Pearson, 2006. 1000 p.

# GLOSSÁRIO

**C++**: é uma linguagem de programação de alto desempenho amplamente utilizada em áreas como desenvolvimento de sistemas, jogos, e aplicações que exigem controle direto de recursos de hardware. Criada por Bjarne Stroustrup em 1985, o C++ é uma extensão do C, incorporando conceitos de programação orientada a objetos e outras funcionalidades avançadas.

**POO**: é um paradigma de desenvolvimento de software que organiza o código em torno de objetos, que são entidades que combinam dados (atributos) e comportamentos (métodos). Baseada em conceitos como abstração, encapsulamento, herança e polimorfismo, a POO facilita a modelagem de problemas do mundo real e promove a reutilização e a modularidade do código.

**UML:** é uma linguagem padrão utilizada para modelagem e documentação de sistemas de software, especialmente no contexto de programação orientada a objetos. Desenvolvida para facilitar a compreensão e o planejamento de projetos complexos, a UML fornece uma notação visual que permite representar estruturas, comportamentos e interações de sistemas.

# APÊNDICE A: DESCRIÇÃO DO DIAGRAMA DE CLASSES

## A1. Descrição do diagrama

### Classe Transaction

**Descrição:**

È uma classe abstrata que serve como base para diferentes tipos de transações. Ela contém informações comuns e métodos relacionados à execução de transações bancárias.

**Atributos:**

accountNumber: int - Número da conta associada à transação.

screen: Screen - Interface de exibição para o usuário.

bankDatabase: BankDatabase - Base de dados bancária para acessar as informações da conta.

**Métodos:**

getScreen(): Screen - Retorna a interface de exibição.

getBankDatabase(): BankDatabase - Retorna a base de dados bancária.

getAccountNumber(): int - Retorna o número da conta.

execute() - Método abstrato que deve ser implementado pelas subclasses para realizar a transação.

### Classe BalanceInquiry

**Descrição:**

Herda de Transaction e representa a transação de consulta de saldo, implementando o método para verificar os saldos disponíveis e totais da conta.

**Métodos:**

execute() - Executa a transação para exibir os saldos da conta.

**Classe ATM**

**Descrição:**
Gerência a interação com o usuário do caixa eletrônico, autenticação e execução de transações bancárias.

**Atributos:**

userAuthenticated: bool - Indica se o usuário foi autenticado.

currentAccountNumber: int - Número da conta do usuário atual.

screen: Screen - Interface de exibição para interações com o usuário.

keyPad: KeyPad - Teclado do ATM para entrada de dados.

cashDispenser: CashDispenser - Componente responsável pela liberação de dinheiro.

depositSlot: DepositSlot - Componente para recebimento de depósitos.

bankDatabase: BankDatabase - Base de dados bancária associada ao ATM.

**Métodos:**

run() - Inicia o funcionamento do ATM.

authenticateUser() - Realiza a autenticação do usuário.

performTransactions() - Gerencia as transações realizadas pelo usuário.

displayMainMenu(): int - Exibe o menu principal para o usuário.

createTransaction(int): Transaction - Cria uma transação específica com base no tipo selecionado.

**Classe Account**

**Descrição:**

Representa as contas bancárias no sistema, armazenando informações e oferecendo métodos para validação e operações financeiras.

**Atributos:**

accountNumber: int - Número identificador da conta.

pin: int - Senha da conta.

availableBalance: double - Saldo disponível para saque.

totalBalance: double - Saldo total incluindo valores pendentes.

**Métodos:**

validatePin(int): bool - Valida a senha informada.

getAvailableBalance(): double - Retorna o saldo disponível.

getTotalBalance(): double - Retorna o saldo total.

credit(double) - Credita um valor na conta.

debit(double) - Debita um valor da conta.

getAccountNumber(): int - Retorna o número da conta.

**Classe BankDatabase**

**Descrição:**
É a base de dados responsável por gerenciar informações das contas bancárias.

**Atributos:**

accounts: vector<Account> - Lista de contas registradas no banco.

**Métodos:**

authenticateUser(int, int): bool - Autentica um usuário com base no número da conta e senha.

getAvailableBalance(int): double - Retorna o saldo disponível de uma conta.

getTotalBalance(int): double - Retorna o saldo total de uma conta.

credit(int, double) - Credita um valor em uma conta específica.

debit(int, double) - Debita um valor de uma conta específica.

getAccount(int): Account - Retorna uma conta com base no número.

**Classe CashDispenser**

**Descrição:**

Representa o dispensador de dinheiro no ATM.

**Atributos:**

INITIAL_COUNT: int = 500 - Quantidade inicial de cédulas no dispensador.

count: int - Quantidade atual de cédulas disponíveis.

**Métodos:**

dispenseCash(int) - Dispensa uma quantia específica em dinheiro.

isSufficientCashAvailable(int): bool - Verifica se há dinheiro suficiente para uma transação.

**Classe Deposit**

**Descrição:**

Representa uma transação de depósito.

**Atributos:**

amount: double - Quantia a ser depositada.

keyPad: KeyPad - Teclado para entrada do valor.

depositSlot: DepositSlot - Componente para recebimento do envelope de depósito.

**Métodos:**

execute() - Executa a transação de depósito.

promptForDepositAmount(): double - Solicita o valor a ser depositado.

**Classe Screen**

**Descrição:**
Representa a interface de exibição do ATM.

**Métodos:**

displayMessage(string) - Exibe uma mensagem na tela.

displayMessageLine(string) - Exibe uma linha de mensagem na tela.

displayDollarAmount(double) - Exibe um valor monetário na tela.

**Classe Withdrawal**

**Descrição:**

Herda de Transaction e representa a transação de saque, com métodos para selecionar e realizar saques de valores disponíveis.

**Atributos:**

amount: int - Quantia a ser sacada.

keyPad: KeyPad - Teclado para seleção do valor.

cashDispenser: CashDispenser - Componente para liberação do dinheiro.

**Métodos:**

execute() - Executa a transação de saque.

displayMenuOfAmounts(): int - Exibe as opções de valores disponíveis para saque.

**Classe DepositSlot**

**Descrição:**
Representa o slot de depósito do ATM.

**Métodos:**

isEnvelopeReceived(): bool - Verifica se o envelope foi recebido.