Universidade Federal de Ouro Preto - UFOP

Instituto de Ciências Exatas e Biológicas - ICEB

Departamento de Computação - DECOM

Ciência da Computação

Relatório TP1

BCC221 - Programação Orientada a Objetos

Mariana Macedo Santos, Gabriel Araújo Saldanha, Gustavo Zacarias, Maria Eduarda Bessa, Gabriel Henrique Moreira Rocha, João Victor Ramalho

Professor: Guillermo Cámara-Chaves

Ouro Preto
30 de junho de 2023

Sumário

Lista de Figuras

1 Introdução

Neste projeto, o objetivo é desenvolver um sistema em C++ que permita o cadastro de funcionários e o controle de ponto dos mesmos. O sistema oferece recursos de autenticação para chefes e funcionários. Para acessar o sistema, o chefe deverá autenticar-se utilizando um nome de usuário e senha específicos. Já os funcionários precisam estar previamente cadastrados pelo chefe e também devem se autenticar utilizando um nome de usuário e senha únicos. Após a autenticação, o sistema deverá exibir um menu com as opções disponíveis.

Para este trabalho prático, implementamos as seguintes classes: Pessoa, Chefe, Supervisor, Funcionario, Vendedor, Hora, Funcionário, Venda. Seguindo a seguinte representação:

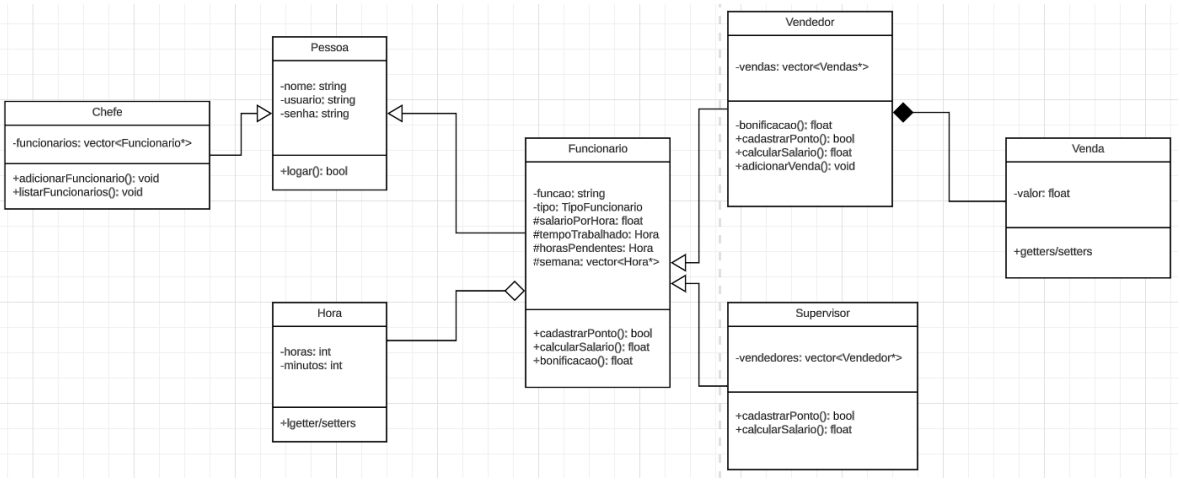

Figura 1: Diagrama UML

Utilizamos diversos conceitos da programação orientada a objetos, como os listados abaixo:

Herança: mecanismo pelo qual elementos mais específicos incorporam a estrututra e comportamento de elementos mais gerais. Como exemplo, na nossa implementação, a classe Funcionário herda da classe Pessoa.

Polimorfismo: permite que objetos de classes diferentes sejam tratados de maneira uniforme, sendo capazes de responder a chamadas de método com o mesmo nome, mesmo que esses métodos tenham comportamentos diferentes em cada classe.

Encapsulamento: Envolve o empacotamento de dados (atributos) e métodos relacionados em uma única entidade chamada de classe. O objetivo principal do encapsulamento é ocultar os detalhes internos da implementação de uma classe e fornecer uma interface controlada e consistente para interagir com os objetos dessa classe.

Classes Abstratas: Ela é projetada para ser usada como um conceito genérico, fornecendo uma estrutura comum e definindo métodos abstratos que devem ser implementados pelas classes derivadas. As classes abstratas são usadas para criar uma hierarquia de classes relacionadas, onde a classe base fornece a estrutura básica e comportamentos gerais, enquanto as classes derivadas fornecem implementações específicas e adicionam funcionalidades adicionais.

2 Desenvolvimento

O objetivo do presente trabalho prático é desenvolver um sistema que permita realizar o cadastro de funcionários e realizar o controle de ponto dos funcionários. Para a realização dos testes, foi desenvolvido um algoritmo na linguagem C++, uma das mais utilizadas quando se trata de programação orientada a objetos, devido a sua alta versatilidade e gama de ferramentas. Utilizou-se o VS Code para a implementação dos códigos, que foram modularizados em diversos arquivos de extensão “.cpp” e ”.h”, que contêm a declaração das classes, o protótipo e a implementação das funções, construtores e

destrutores. Foi utilizada a plataforma LucidChart, que consiste em um aplicativo de diagramação baseado na web que permite aos usuários colaborar visualmente no desenho, revisão e compartilhamento de gráficos e diagramas e melhorar processos, sistemas e estruturas organizacionais, para a criação dos diagramas de classe resultantes da implementação.

Link do LucidChart: https://lucid.app/lucidchart/f168ddd8-798f-40d7-b759-caa4c9a39c2d/edit?viewport$_{l}oc = -800$

Serão mostradas, a seguir, as classes e funções utilizadas no código através do recurso da representação gráfica de Diagramas de Classe (UML - Unified Modeling Language).

2.1 Classe Pessoa

A classe é composta por 3 strings(nome, usuário e senha). Ela é uma classe composta por getters e setters de cada variável. Além disso, ela funciona como uma superclasse no programa.

2.2 Classe Chefe

A classe é composta por 3 strings(nome, usuario, senha) e 1 vector tipo Funcionario*(funcionarios). Ela é uma subclasse de Pessoa, composta por getters e setters de cada variável.

2.3 Classe Funcionário

A classe é composta por 4 strings(nome, usuario, senha, funcao), 1 TipoFuncionario(tipo), 2 tipos Hora(tempoTrabalhado e horasPendentes), 1 float(salario) e 1 vector tipo Hora*(semana). Ela é uma subclasse de Pessoa, com getters e setters para suas variáveis, além de ter uma função tempoSemana, uma função ponto e uma função calculoSalarioPorHoras. Ela também é uma superclasse de Vendedor e Supervisor.

2.4 Classe Vendedor

A classe é composta por 4 strings(nome, usuario, senha, funcao), 1 TipoFuncionario(tipo), 2 tipos Hora(tempoTrabalhado e horasPendentes), 1 float(salario), 1 vector tipo Hora*(semana) e 1 vector tipo Venda* (vendas). Ela é uma subclasse de Funcionário, com getters e setters para suas variáveis, além de ter uma função cadastrarPonto, uma função calcularSalario, uma função bonificacao, uma função adicionarVenda.

2.5 Classe Supervisor

A classe é composta por 4 strings(nome, usuario, senha, funcao), 1 TipoFuncionario(tipo), 2 tipos Hora(tempoTrabalhado e horasPendentes), 1 float(salario), 1 vector tipo Vendedor* (vendedores) e 1 vector tipo Hora* (semana). Ela é uma subclasse de Funcionário, com getters e setters para suas variáveis, além de ter uma função cadastrarPonto e uma função calcularSalario.

2.6 Classe Hora

A classe é composta por 2 int(horas e minutos). Ela possui apenas getters e setters para suas variáveis.

2.7 Classe Venda

A classe é composta por 1 float(valor). Ela possui apenas getters e setters para sua variáveis.

Compilação e Execução:

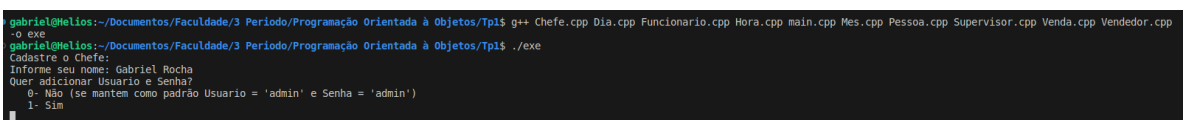

Figura 2: Compilação e Execução

3 Conclusão/Considerações Finais

Destarte, o trabalho permitiu ao grupo uma melhor integração aos conhecimentos da linguagem C++, assim como os paradigmas de uma linguagem orientada a objeto, sendo muito útil no curso e atualmente no mercado de trabalho.

Os conhecimentos de classe, herança, polimorfismo, diagrama UML, além das demais áreas da linguagem foram abordadas, expandindo assim o conhecimento individual dos conceitos apresentados em sala de aula. O trabalho foi concluído em grupo, e foi utilizada a plataforma Git para manter a sincronia do que estava sendo feito mesmo que em plataformas diferentes.

Referências