Trabalho prático: Sistema de gestão de RH: folha de pagamento

Diego Araújo dos Santos

Universidade Federal de Minas Gerais

Resumo: O objetivo desse TP foi de explorar os conceitos básicos de programação orientada a de objeto, como classes, objetos, atributos, comportamentos, encapsulamento, reusabilidade e herança.

Introdução

O sistema desenvolvido possibilita consultar informações sobre o pagamento de funcionários de uma empresa. Essa consulta poderia ser realizada por funcionários do setor de RH ou contábil.

Os dados foram obtidos através de um arquivo *clients.dat* , disponível na pasta do projeto. Esse arquivo, contendo as informações, pode ser obtido numa situação real por meio de outro software ou dispositivo, como uma máquina de apontamento eletrônico.

Recursos

- Realizar consulta de informações sobre pagamento de todos os funcionários de uma empresa.
- Realizar consulta sobre pagamento de cada funcionário individualmente.
- Calcular salário de funcionários que recebem comissão.
- Calcular salário de funcionários que não são comissionados.
- Exibir informações na tela.

UserStories

Como um gerente de RH eu gostaria de visualizar o salário de um funcionário no mês corrente para efetuar o seu pagamento.

Critérios de aceitação:

- Exibir matrícula, nome e cargo.
- Exibir taxa de comissão e vendas realizadas.
- Exibir salário base.
- Exibir cálculo do salário total.

Como contador da empresa eu gostaria de visualizar informações sobre vendas totais realizadas e taxa de comissão de todos os funcionários da empresa para validar um balanço contábil.

Critérios de aceitação:

- Exibir matrícula, nome e cargo.
- Exibir taxa de comissão e vendas realizadas.
- Exibir salário base.

Estrutura do sistema

Foram criadas duas classes ComissaoEmpregado e Salariobase, como mostra o diagrama UML a seguir:

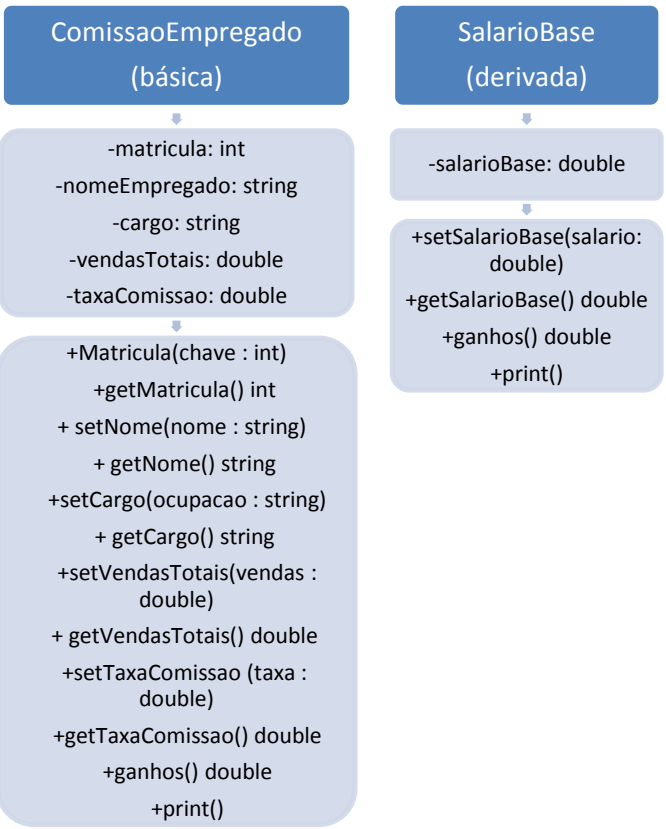

Figura 1. Diagramas UML das classes ComissaoEmpregado e Salariobase .

Conceitos

A. Modularização e reutilização

Para possibilitar a reutilização de código, as interfaces das classes foram separadas de sua implantação.

A interface da classe ComissaoEmpregado descreve que os serviços os clientes podem utilizar e como solicitar, tais como:

- **void setMatricula():** atribuir o valor da matricula ao membro de dados **matricula**.
- **void setNome ():** atribuir o nome do funcionário ao membro de dados **nome.**
- **string getCargo():** obter o nome do cargo do funcionário acessando o membro de dados **cargo**.
- **double ganhos():** calcular o salario total do funcionário somando o salário base com a comissão, quando aplicada.

Os arquivos de cabeçalho (.h) e de código fonte (.cpp) utilizados no projeto podem ser visto na figura 2.

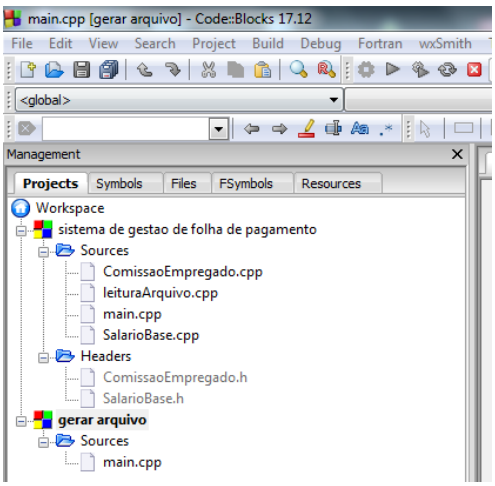

Figura 2. Estrutura do projeto.

A função leituraArquivo() foi implementada num arquivo de condigo fonte separadamente. Ela realiza a busca de dados de registro no arquivo *clients.dat*.

O arquivo foi criado por outro programa, o gerar arquivo, figura 3. Essa etapa pressupõe que numa situação real o arquivo seria disponibilizado por outro dispositivo, como uma maquina de apontamento eletrônico. Foram cadastrados previamente 10 funcionários.

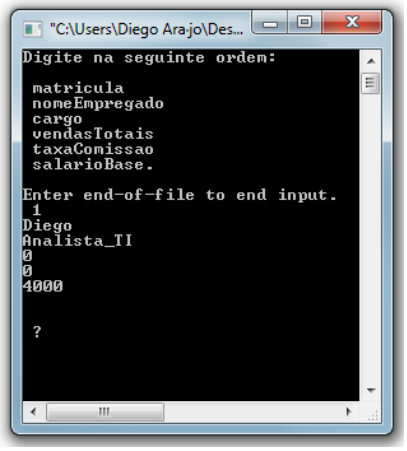

Figura 3. Tela do console do programa gerar arquivo.

B. Herança

A classe SalarioBase deriva da classe ComissaoEmpregado. A interface da classe SalarioBase é mostrada na figura 4.

```
7   #include "ComissaoEmpregado.h"// definição da classe Comiss
8
9   class SalarioBase : public ComissaoEmpregado
10  {
11  public:
12      SalarioBase( int , const string &, const string &, doub
13
14      void setSalarioBase( double ); // configura o salário ba
15      double getSalarioBase() const;// retorna o salário base
16
17      double ganhos() const; // calcula os rendimentos
18      void print() const; // imprime o objeto SalarioBase
19
20  private:
21      double salarioBase; // salário base do empregado
22  };// fim da classe SalarioBase
23
```

Figura 4. Interface da classe SalarioBase.

Portanto, um objeto SalarioBase é um ComissaoEmpregado pois as capacidades da classe básica e transferida por herança à classe derivada.

Figura 5. Relação de herança entre SalarioBase e ComissaoEmpregado.

```
28  double SalarioBase::ganhos() const
29  {
30      return getSalarioBase() + ComissaoEmpregado::ganhos();
31  }
32
```

Figura 6. Função ganhos()da classe SalarioBase

```
77  //Calcula os rendimentos----------------------
78
79  double ComissaoEmpregado::ganhos() const
80  {
81      return getTaxaComissao() * getVendasTotais();
82  }
83
```

Figura 7. Função ganhos()da classe ComissaoEmpregado.

A função ganhos() da classe derivada SalarioBase, figura 6, redefine a função membro ganhos() da classe básica ComissaoEmpregado, figura 7, para calcular salário de um empregado comissionado com salário base.

A função ganhos() da classe SalarioBase obtém a parte do salário do empregado baseada exclusivamente na comissão chamando a função ganhos() da classe ComissaoEmpregado.

Para tal, é utilizando o operador de resolução de escopo binário :: antes do nome da função membro da classe básica, ComissaoEmpregado :: ganhos().

Assim, a classe SalarioBase adiciona o salário base ao valor da comissão, se existir, para calcular os rendimentos totais do empregado.

C. Validação de entradas

Para assegurar que o programa funcione de forma consistente foram validades algumas entradas, como taxa de comissão, salario base, valor de vendas, etc.

```
15  // configura o salário base----------------------
16  void SalarioBase::setSalarioBase( double salario )
17  {
18      salarioBase = ( salario < 0.0 ) ? 0.0 : salario;
19  }
20
```

Figura 8. Validação do valor da entrada salario base.

```
65  // configura a taxa de comissao----------------------
66  void ComissaoEmpregado::setTaxaComissao( double taxa )
67  {
68      taxaComissao = ( taxa > 0.0 && taxa < 1.0 ) ? taxa : 0.0;
69  }
```

Figura 9. Validação do valor da entrada taxa de comissão.

Utilização

O programa possui uma tela de menu onde se pode escolher entre as opções de consultar dados de todos funcionários ou consultar individualmente cada funcionário. Para tal deve se digitar 1 ou 2 no menu de seleção.

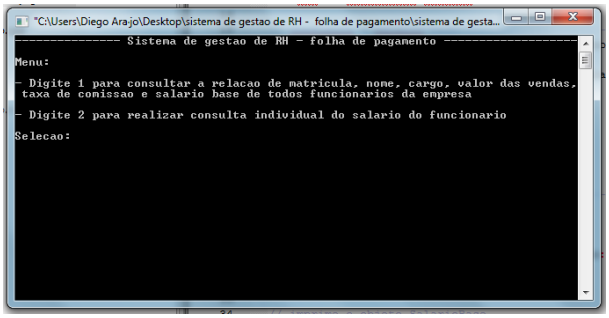

Figura 10. Tela de menu principal.

Digitando um 1, o programa busca no arquivo clients.dat a pelos dados de todos funcionários de uma empresa. Posteriormente exibe essas informações na tela, figura 11.

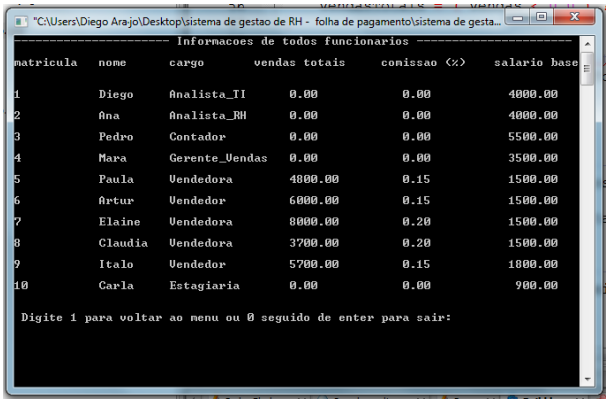

Figura 11. Consulta de dados de todos os funcionários.

Digitando um 2, o programa entra na opção de consulta individual. A matricula do funcionário é requerida. Deve se digitar um numero de 1 a 10, pois o quadro de funcionários é de 10 pessoas, caso contrário o programa volta para tela inicial.

Em seguida é feito uma busca no arquivo clients.dat por um determinado funcionário mediante uma chave. Uma vez encontrado, o um objeto (SalarioBase empregado) é criado.

O objeto é instanciado pelo construtor explicitamente com os dados obtidos pela leitura do registo do funcionário do arquivo clients.dat.

Finamente, as funções membro da Classe SalarioBase e ComissãoEmpregado são chamadas para fornecer os dados e operações requisitadas.

O salario total é calculado pela função ganhos(), que retorna a soma do salario base com a comissão, quando aplicada, e então exibido na tela, figura 12.

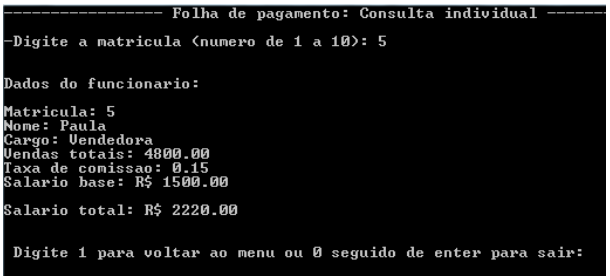

Figura 12. Consulta de dados de todos os funcionários.

Conclusão

O programa funcionou como especificado, e descrito pelo *UserStories* , na primeira etapa do projeto.

Foi possível compreender melhor a aplicabilidade de alguns conceitos de programação orientada a objetos como: encapsulamento, herança, separação entre a interface e implementação de uma classe para reuso de código, etc.

Algumas funcionalidades devem ser acrescidas como casos de testes e tratamento de exceções para assegura que o programa corretamente em tempo de execução.

Referências bibliográficas

Savitch, Walter J. Absolute C++ / Walter Savitch ; contributor, Kenrick Mock. -- 5th ed.

Deitel, H.M.,. C++: como programar / H.M. Deitel, P.J. Deitel; tradução Edson Furmankiewicz ; revisão técnica Fábio Lucchini. — São Paulo: Pearson Prentice Hall, 2006.

Kris Jamsa e Lars Klander. Programando em C/C++ - A Biblia; São Paulo: Makron Books, 1999.