Nome: Maria Eduarda Silveira

Thomas do Vale

Weverton Rodrigues Arantes

**Documentação Técnica**

O projeto possui as seguintes structs: Cliente, Filme, Categoria, Funcionário, Fornecedor, Locadora, Locação e Conta

O projeto tem três tipos de persistência: arquivo binário, arquivo texto e array dinâmico. Para armazenar os tamanhos (quantidade em cada persistência) as persistências possuem uma biblioteca chamada tamanho.

**Tamanho**

A biblioteca “tamanho” contem 2 dos 3 tipos de tamanhos (um tipo para cada tipo de persistência) das persistências de cada struct, nela contém os tamanhos para arquivo texto e arquivo binário. O array dinâmico possui uma variável chamada de tamanhoNomeStruct.

**DAOS**

Os daos consistem em 5 funções com 3 tipo de persistências, na seguinte ordem dos tipos binário, texto e array dinâmico, e as funções na seguinte ordem Inclusão, Listar, Consultar, Alterar e Atualizar.

As funções de inclusão esperam uma struct das suas respectivas bibliotecas e o incluem em um dos tipos citados acima.

As funções de listagem não há parâmetros e retorna um array dinâmico das suas respectivos structs.

As funções de consultar esperam um código de parâmetro da struct a consultar e retorna uma struct respectiva.

As funções de alteração esperam uma struct e um código da struct a ser alterada.

As funções de deleção funcionam de 2 formas, para texto faz uma exclusão verdadeira, enquanto para os arquivos binários e array dinâmico faz uma exclusão logica utilizando o campo deletado de cada struct.

**Controle**

Os Controles têm como objetivo intermediar a passagem de dados entre a VISÃO e os DAOS e vice-versa, fazendo as operações necessárias para que a parte receptora da informação consiga interpreta-la.

Além disso os Controles monitoram o tipo de persistência e o tamanho de cada uma delas. Uma das funções importantes dessa camada do projeto é o controle de inclusão que faz cálculos para fazer uma espécie de auto incremento do código de cada struct a ser inserida para persistência.

As funções padrão de cada controle são:

int qtdNomeStruct() – Essa função verifica o tipo de persistência a ser utilizada pelo sistema (memoria, texto ou binário) e retorna a quantidade de elementos persistidos da respectiva struct em tal tipo de persistência.

int salvaNomeStruct(Struct \* s) – Essa função recebe um ponteiro para a struct a ser persistida. Após isso ela pega a quantidade de clientes persistidos no tipo de persistência em uso pelo sistema (memória, texto ou binário). A partir dessa quantidade é somado mais uma unidade para se gerar o código da struct que será persistida passando a mesma para o DAO que efetivará a persistência da struct. A função retorna um int significando 1 para operação com sucesso e 0 para operação falha.

int atualizaStruct(Struct c) – Essa função recebe a struct já com os valores alterados pela Visao, (exceto o código, este nunca é alterado). Após isso encaminha ao DAO a struct e seu código, para que seja efetivada a alteração. Retorna 1 em caso de operação com sucesso e 0 em caso de operação falha.

int deletaStruct(float codigoDaStruct) – Essa função recebe o código do tipo da struct a ser deletada e o informa ao método do DAO correto de acordo com o tipo de persistência em uso pelo sistema.

Struct \* listaStruct() – Essa função retorna um ponteiro contendo todas as structs persistidas no tipo de persistência em uso pelo sistema. Isto é feito chamando o método de listagem correto do DAO.

Struct\* consultaStruct(flot codigoDaStruct) – Essa função recebe o código da struct a ser consultada. Após isso verifica o tipo de persistência em uso pelo sistema e chama o método correto do DAO para retornar a struct dejesada.

Há um tipo especial de controle que lida com a persistência usada pelo sistema no momento. Os métodos do conPersistencia são:

int getTipoPersistencia() – Retorna o tipo de persistência em uso pelo sistema atualmente.

void setTipoPersistencia(int tipoPersistencia) – Define o tipo de persistência a ser usada pelo sistema.

**Visão**

Os DAOS de visão passam primeiro por 2 menu, mas basicamente chamam as funções do controle em seus respectivos números de chamadas e retorna uma mensagem de sucesso se necessário.

**Menus**

Antes efetivamente do DAO o programa passa por 2 menus, como dito anteriormente. MARIA ESCREVE AQUI AS PRINCIPAIS FUNÇÕES DOS DOIS MENUS.

**Exportação**

A exportação espera a quantidade de dados que serão inseridos naquela struct e inclui seus dados em sua respectiva struct. O usuário pode escolher qual struct que ele deseja exportar os dados. Os arquivos precisam estar em XML

**Importação**

A importação não tem parâmetros e gera um arquivo em XML para que o usuário possa o utilizar com achar melhor. O usuário pode escolher qual struct que ele deseja exportar os dados.

**Funções uteis**

O programa possui duas funções uteis, a primeira é “to\_seconds()” que retorna um time\_t e ela espera uma constate de char com uma data separada da seguinte maneira xx/xx/xxxx.

A segunda é para pegar a data atual e ela retorna um array de char e não espera parâmetros. Ela basicamente pega a data atual e a coloca no formato mostrado acima. Ela se chama “pegaDataAtual()”.

**FeedBack**

O FeedBack se consiste em 9 funções para dar relatórios ao usuário.

As duas primeiras funções são de cliente. A primeira dá um feedback baseado em uma distancia de códigos. A segunda pelo o sexo de cada um.

A próxima é por filme e se consiste em um distancia de códigos assim como a de cliente.

Outro retorna um int e calcula o número de vezes necessárias para que um filme que já foi comprado seja pago pelas suas locações.

A próxima se consiste em Locações. A primeira de locação se consiste na distância dos códigos. Após ela por nome e espera um cliente por parâmetro. Então a função de comparação entre datas que usa a função to\_seconts e espera 2 ponteiros de char com datas no formato já citado. Depois uma função de contas a receber de clientes que espera 2 códigos.

Por final a de contas. Sua primeira função se consiste em comparação entre datas assim como a de locação. A ultima se consiste em distância de códigos das contas a pagar e espera 2 parâmetros de código.

**Entrada Filmes**

A entrada de filmes requer as informações da nota fiscal. Após essas informações estarem no sistema a quantidade dos filmes comprados pela locadora é atualizada. Após isso é gerada e persistida a nota fiscal correspondente na pasta do sistema.

**Nota Fiscal**

Está função gera as notas fiscais dos filmes adquiridos pela locadora. Ela espera como parâmetro o fornecedor, frete, imposto, os filmes, suas quantidades e seus valores. Gera uma nota fiscal e calcula o preço do frete, o total e do imposto.

**Locação Filmes**

Cada cliente só pode alugar 3 filmes por vez. A locação pede o código do cliente, os códigos dos filmes alugados, a quantidade de filme alugada (max 3), o total pago pelo cliente, o tipo de locação (se a vista ou parcelado), o número de parcelas e a entrada nessas parcelas (se o cliente deu uma entrada). Após isso é pega a data atual do sistema que será a data da locação e a locação é salva e persistida.

**Devolução Filmes**

A devolução de filmes pede o código do cliente que está devolvendo o filme e o código do filme que está sendo devolvido. A partir dai o sistema faz os cálculos e pega a última locação feita pelo cliente e muda os dados dela para computar que o filme foi devolvido e atualiza a persistência dessa locação. A devolução também aumenta a quantidade do filme devolvido.

**COLOCAR ALGO QUE FALTOU EM AMBOS**