PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS - LOURDES

DIEGO HENRIQUE XAVIER DOS SANTOS, MARCOS VINÍCIUS NUNES REIS,
RAFAEL GEORGETTI GROSSI E VITOR DANIEL SILVA MELO

TRABALHO PRÁTICO - FASE III

ALGORITMOS E ESTRUTURAS DE DADOS III

BELO HORIZONTE - MG

2025

DIEGO HENRIQUE XAVIER DOS SANTOS, MARCOS VINÍCIUS NUNES REIS,
RAFAEL GEORGETTI GROSSI E VITOR DANIEL SILVA MELO

TRABALHO PRÁTICO - FASE III

ALGORITMOS E ESTRUTURAS DE DADOS III

Trabalho apresentado ao curso superior de Ciência da Computação da PUC-MG Lourdes para o cumprimento das exigências da disciplina Algoritmos e Estruturas de Dados III

Orientador: Walisson Ferreira de Carvalho

BELO HORIZONTE - MG
2025

SUMÁRIO

FORMULÁRIO

a) Qual foi o relacionamento N:N escolhido e quais tabelas ele conecta?

O relacionamento N:N escolhido foi “TarefaCategoria” e ele conecta as tabelas “Tarefa” e “Categoria”.

b) Qual estrutura de índice foi utilizada (B+ ou Hash Extensível)? Justifique a escolha.

A estrutura de índice utilizada foi o Hash Extensível. A escolha do Hash Extensível se dá pela relação N:N da Tarefas e Categorias onde a listagem é feita sem necessidade de ordenação e permite a busca rápida como quais tarefas pertencem a uma categoria ou vice-versa, assim a escolha de Hash Extensível se enquadra como a mais ótima.

c) Como foi implementada a chave composta da tabela intermediária?

A chave composta da tabela intermediária foi implementada através de dois campos: idTarefa e idCategoria. Cada par de tarefa e categoria é mapeado para um ponteiro que aponta para a posição do registro real no arquivo heap.

d) Como é feita a busca eficiente de registros por meio do índice?

A busca eficiente de registros por meio do índice é feita através do Hash Extensível utilizando uma chave temporária de busca de idTarefa e idCategoria para leitura no hash que retornará um ponteiro do registro correspondente no arquivo heap. Dessa forma, pelo acesso direto, se torna uma operação constante O(1).

e) Como o sistema trata a integridade referencial (remoção/atualização) entre as tabelas?

O sistema trata a integridade referencial entre as tabelas, para a remoção, através de uma *flag* ativo, sendo marcada para falso, ou seja, exclusão lógica,. Já no caso da atualização o código mantém o id original e apenas substitui os campos alterados no heap. Se o registro mudar de posição no arquivo, o hash atualiza para apontar para a nova posição.

f) Como foi organizada a persistência dos dados dessa nova tabela (mesmo padrão de cabeçalho e lápide)?

A persistência dos dados dessa nova tabela foi organizada utilizando o mesmo padrão de cabeçalho e lápide. Possui um cabeçalho com o último ID utilizado e registros armazenados em formato binário, contendo os campos id, idTarefa, idCategoria, prioridade e ativo. E a lápide, um campo booleano ativo que controla a exclusão lógica.

g) Descreva como o código da tabela intermediária se integra com o CRUD das tabelas principais.

A integração da tabela intermediária com o CRUD das tabelas principais pelo código se dá pela classe ArquivoTarefaCategoriaHash que gerencia o vínculo entre as entidades Tarefa e Categoria. Ao criar uma nova associação, o CRUD chama a função *create()* da classe intermediária que insere o relacionamento no arquivo heap e o índice no heap. Quando uma tarefa ou categoria é removida, é utilizado as funções *listarPorTarefa()* e *listarPorCategoria()* que localiza e exclui os relacionamentos correspondentes. A listagem de categorias e tarefas associadas entre si acontece da mesma forma, essas funções acessam o índice hash e recuperam os registros.

h) Descreva como está organizada a estrutura de diretórios e módulos no repositório após esta fase.

Projeto estruturado com arquitetura MVC e DAO.

```
.
├── docs                    # Arquivos de documentação
└── src                     # Código-fonte do projeto
    ├── Main.java           # Função principal a ser executada
    ├── controllers         # Controlador das operações do CRUD
    ├── dao                 # Data Access Objects
    ├── data                # Registros e arquivos de índices
    ├── models              # Entidades do projeto
    │   └── structures      # Entidades de estruturas para chaves de pesquisa
    └── views               # Interface do menu
```

REPOSITÓRIO GITHUB

- https://github.com/vmelooo/tp-aeds3