



Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação
Disciplina: Algoritmos e Estruturas de Dados II


Trabalho Prático VI

(Data de Entrega: 31/10)

Regras Gerais

- A partir deste trabalho, você deve utilizar as soluções em **Java** ou **C**, implementados no TP-04.
- para cada questão, devemos submeter apenas um arquivo (.java ou .c). Essa regra será necessária para a submissão de trabalhos no Verde e no identificador de plágio utilizado na disciplina.
- Para cada exercício: faça (entende-se análise, implementação e comentários), teste (várias vezes) e submeta no Verde. Os exercícios não testados/comentados serão penalizados.
- A correção será realizada automaticamente pelo sistema Verde. Entretanto, você poderá ser entrevistado durante as aulas de laboratório e deverá demonstrar pleno domínio sobre o código desenvolvido. A utilização de ferramentas de IA generativa na elaboração do trabalho implicará em nota zero.




Questões

1.  **Lista com Alocação Sequencial em Java:** Crie uma Lista de registros baseada na de inteiros vista na sala de aula. Sua lista deve conter todos os atributos e métodos existentes na lista de inteiros, contudo, adaptados para a classe GAME. Lembre-se que, na verdade, temos uma lista de ponteiros (ou referências) e cada um deles aponta para um registro. Neste exercício, faremos inserções, remoções e mostraremos os elementos de nossa lista.

Os métodos de inserir e remover devem operar conforme descrito a seguir, respeitando parâmetros e retornos. Primeiro, o `void inserirInicio(Game game)` insere um registro na primeira posição da Lista e remaneja os demais. Segundo, o `void inserir(Game game, int posição)` insere um registro na posição p da Lista, onde $p < n$ e n é o número de registros cadastrados. Em seguida, esse método remaneja os demais registros. O `void inserirFim(Game game)` insere um registro

na última posição da Lista. O `Game removerInicio()` remove e retorna o primeiro registro cadastrado na Lista e remaneja os demais. O `Game remover(int posição)` remove e retorna o registro cadastrado na p -ésima posição da Lista e remaneja os demais. O `Game removerFim()` remove e retorna o último registro cadastrado na lista.

A entrada padrão é composta por duas partes. A primeira é igual a entrada da primeira questão do TP04. As demais linhas correspondem a segunda parte. A primeira linha da segunda parte tem um número inteiro n indicando a quantidade de registros a serem inseridos/removidos. Nas próximas n linhas, tem-se n comandos de inserção/remoção a serem processados neste exercício. Cada uma dessas linhas tem uma palavra de comando: **II** inserir no início, **I*** inserir em qualquer posição, **IF** inserir no fim, **RI** remover no início, **R*** remover em qualquer posição e **RF** remover no fim. No caso dos comandos de inserir, temos também o nome do arquivo que contém o registro a ser inserido. No caso dos comandos de “em qualquer posição”, temos também esse nome. No Inserir, a posição fica imediatamente após a palavra de comando. A saída padrão tem uma linha para cada registro removido sendo que essa informação será constituída pela palavra “(R)” e o atributo **Name**. No final, a saída mostra os atributos relativos a cada registro cadastrado na lista após as operações de inserção e remoção.

2.  **Lista com Alocação Flexível em C:** Refazer a questão anterior usando uma lista dinâmica simples.
3.  **Pilha com Alocação Flexível em Java:** Crie uma Pilha de registros baseada na pilha de inteiros vista na sala de aula. Neste exercício, faremos inserções, remoções e mostraremos os elementos de nossa pilha. A entrada e a saída padrão serão como as da questão anterior, contudo, teremos apenas os comandos **I** para inserir na pilha (empilhar) e **R** para remover (desempilhar).
4.  **Fila com Alocação Flexível em Java:** Crie uma Fila de registros baseada na fila de inteiros vista na sala de aula. Neste exercício, faremos inserções, remoções e mostraremos os elementos de nossa fila. A entrada e a saída padrão serão como as da questão anterior, contudo, teremos apenas os comandos **I** para inserir na fila (enfileirar) e **R** para remover (desenfileirar).