



Trabalho Prático III

Regras Básicas

1. extends Trabalho Prático 02
2. Fique atento ao Charset dos arquivos de entrada e saída.
3. Em caso de cópia de trabalhos, o aluno terá a sua nota zerada com aplicação de advertência.

POKÉMON é uma franquia de mídia criada pela Nintendo, Game Freak e Creatures, que começou como uma série de jogos de RPG para o Game Boy em 1996. Criada por Satoshi Tajiri, a série POKÉMON rapidamente se tornou um fenômeno global, evoluindo para uma das franquias mais populares e lucrativas do mundo.

A premissa básica dos jogos gira em torno dos POKÉMON (Pocket Monsters), criaturas fictícias que os jogadores, conhecidos como Treinadores, capturam e treinam para lutar contra outros POKÉMON. Cada jogo geralmente começa com o jogador recebendo seu primeiro POKÉMON de um professor e então viajando por várias regiões para capturar novos POKÉMON, desafiar líderes de ginásio, e eventualmente competir na Liga POKÉMON.

Um elemento central dos jogos é a Pokédex, uma enciclopédia eletrônica que registra informações sobre todos os POKÉMON encontrados ou capturados. A Pokédex fornece detalhes como espécie, altura, peso, e uma descrição única para cada POKÉMON. O objetivo de completar a Pokédex, capturando todos os POKÉMON disponíveis, é uma das principais motivações dos treinadores.

Os POKÉMON possuem individualidades marcantes, como tipos e habilidades específicas. Existem vários tipos, como Fogo, Água, Planta, Elétrico, Psíquico, Dragão, entre outros. Esses tipos determinam as fraquezas e resistências de um POKÉMON em batalhas, criando uma dinâmica estratégica.



Por exemplo, um POKÉMON do tipo Fogo é forte contra POKÉMON do tipo Planta, mas fraco contra POKÉMON do tipo Água.

Além dos tipos, cada POKÉMON tem habilidades especiais que conferem vantagens em batalha. Por exemplo, a habilidade “Levitate” torna um POKÉMON imune a ataques do tipo Terra, enquanto “Intimidate” reduz o poder de ataque do oponente ao início de uma batalha.

Dentro do universo POKÉMON, também existem os POKÉMON lendários, que são raros, poderosos e fundamentais para a história dos jogos. Esses POKÉMON são únicos, com habilidades e estatísticas superiores em comparação com os POKÉMON comuns. Exemplos de POKÉMON lendários incluem Mewtwo, Zapdos, e Rayquaza, cada um com seu próprio lore e importância dentro da série.

POKÉMON se tornou um fenômeno cultural, com impacto significativo em várias gerações. Os jogos não apenas entretêm, mas também promovem valores como amizade, estratégia e a importância de cuidar de outras criaturas. A série de jogos POKÉMON continua a crescer, lançando novos títulos regularmente e mantendo sua relevância na cultura pop global.

O arquivo POKEMON.CSV contém um conjunto de dados dos pokémons do jogo extraídos do site [Kaggle](#). Essa base contém 801 pokémons. Este arquivo sofreu algumas adaptações para ser utilizado neste e nos próximos trabalhos práticos. Tal arquivo deve ser copiado para a pasta /tmp/. Quando reiniciamos o Linux, ele normalmente apaga os arquivos existentes na pasta /tmp/.

Implemente os itens pedidos a seguir.

Estruturas Sequenciais

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 POKÉMON. 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(Pokemon pokemon)` insere um registro na primeira posição da Lista e remaneja os demais. Segundo, o `void inserir(Pokemon pokemon, 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(Pokemon pokemon)` insere um registro na última posição da Lista. O `Pokemon removerInicio()` remove e retorna o primeiro registro cadastrado na Lista e remaneja os demais. O `Pokemon remover(int posição)` remove e retorna o registro cadastrado na p -ésima posição da Lista e remaneja os demais. O `Pokemon 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. 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 Sequencial em C:** Repita a anterior na linguagem C
3. **Pilha com Alocação Sequencial em C:** 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 Circular com Alocação Sequencial em C:** Crie uma classe *Fila Circular* de POKÉMON. Essa fila deve ter tamanho cinco. Em seguida, faça um programa que leia vários registros e insira seus atributos na fila. Quando o programa tiver que inserir um registro e a fila estiver cheia, antes, ele deve fazer uma remoção. A entrada padrão será igual à da questão anterior. A saída padrão será um número inteiro para cada registro inserido na fila. Esse número corresponde à média arredondada do **captureRate** dos registros contidos na fila após cada inserção. Além disso, para cada registro removido da fila, a saída padrão também apresenta a palavra “(R)” e alguns atributos desse registro. Por último, a saída padrão mostra os registros existentes na fila seguindo o padrão da questão anterior.

Estruturas Flexíveis

5. **Lista com Alocação Flexível em C:** Refazer a Questão 1 “Lista com Alocação Sequencial” usando lista dinâmica simples em C.
6. **Pilha com Alocação Flexível em C:** Refazer a Questão 3 “Pilha com Alocação Sequencial” em Java.
7. **Fila com Alocação Flexível em C:** Refazer a Questão 4 “Fila Circular com Alocação Sequencial” em C. Lembre-se que essa fila terá tamanho máximo igual a cinco.
8. **Quicksort com LISTA DINÂMICA DUPLAMENTE ENCADEADA em C:** Refaça a Questão “Quicksort” 10 do Trabalho Prático II - com lista duplamente encadeada. O nome do

arquivo de log será matrícula_quicksort2.txt.

9. **Pilha com Alocação Flexível em Java:** Refaça a questão 3 deste TP na linguagem C.
10. **Quicksort com LISTA DINÂMICA DUPLAMENTE ENCADEADA em Java:** Refaça a questão 8 deste TP na linguagem Java.
11. **Matriz Dinâmica em Java:** Complete o código da classe matriz dinâmica visto na sala de aula. A primeira tarefa consiste em, no construtor da classe Matriz, dados os números de linha e coluna, fazer as devidas alocações de células. As demais tarefas são as implementações dos métodos Matriz soma(Matriz), Matriz multiplicacao(Matriz), void mostrarDiagonalPrincipal() e void mostrarDiagonalSecundaria(). A entrada padrão é composta por vários casos de teste sendo que o número de casos é um inteiro contido na primeira linha da entrada. Em seguida, temos cada um dos casos de teste. Cada caso é composto por duas matrizes. Para cada caso de teste, temos que suas duas primeiras linhas contêm um número inteiro cada representando os números de linhas e de colunas da primeira matriz, respectivamente. Em seguida, temos os elementos da primeira matriz que estão representados nas próximas l linhas onde l é o número de linhas dessa matriz. Cada uma dessas linhas têm c colunas onde c é o número de colunas dessa matriz. Nas duas linhas seguintes, temos os números de linhas e colunas da segunda matriz do caso de teste. As $l2$ linhas seguintes têm $c2$ colunas contendo os elementos da segunda matriz. $l2$ e $c2$ correspondem aos números de linhas e colunas da segunda matriz do caso de teste, respectivamente. A saída padrão contém várias linhas para cada caso de teste. As duas primeiras linhas de saída de um caso de teste correspondem às diagonais principal e secundária da primeira matriz, respectivamente. As demais ls linhas de um caso de teste correspondem as linhas matriz obtida pela soma das duas matrizes do caso de teste sendo que essas linhas contêm cs colunas referentes às colunas da matriz de soma. Da mesma forma, as linhas seguintes do caso teste contêm lm linhas com cm colunas representando os elementos da matriz de multiplicação onde lm e cm são os números de linhas e colunas da matriz de multiplicação.