PUC Minas

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Lista de Exercícios 05

Professor: *Marta Noronha*

Disciplina: *Laboratório de Algoritmos e Estrutura de Dados II*
Data de entrega: *29/09/2023*

Requisitos

1. Todos os programas deverão ser desenvolvidos na linguagem de programação Java.
2. Essas práticas poderão ser desenvolvidas em grupos de, no máximo, dois integrantes.
3. Cópias, se existirem, serão encaminhadas ao colegiado de coordenação didática do curso.
4. Fique atento ao charset dos arquivos de entrada e saída. Recomenda-se a utilização dos métodos da classe MyIO.java para leitura de dados do teclado. É necessário definir o charset a ser utilizado antes de começar a leitura de dados do teclado, da seguinte forma: `MyIO.setCharset("UTF-8")`.
5. As saídas esperadas, cadastradas no VERDEpelo professor, foram geradas empregando-se: `System.out.println(`
6. Em cada submissão, enviar apenas um arquivo (.java). A regra será necessária para a submissão de exercícios no VERDE e no identificador de plágios utilizado na disciplina.
7. A resolução (código) de cada exercício deverá ser submetida ao VERDE.
8. A execução do código submetido será realizada automaticamente pelo VERDE, mas o código será analisado e validado pelo professor.

Base de Dados

Para construção desta atividade foi utilizada a base de dados 7k Books.

Este conjunto de dados contém uma lista simplificada de livros populares, incluindo informações de ISBN13, ISBN10, título, subtítulo, categoria, descrição, ano de publicação, número de páginas, avaliação (nota) média e quantidade de avaliações. O data set foi baseado na base de dados Goodread books.

Durante a etapa de filtragem, foram selecionados os livros mais populares, usando como critério a quantidade de avaliações. Foram removidos registros com informações incompletas ou que apresentavam possíveis conteúdos sensíveis / inadequados à disciplina.

Exercícios - Criação da classe Livros

Crie uma classe Livro com os seguintes atributos privados:

- `ISBN` (Long);
- `titulo` (String);

- `autor principal` (String);
- `segundo autor` (String);
- `categoria` (String);
- `descricao` (String);
- `ano publicação` (int);
- `quantidade páginas` (int);
- `nota avaliação` (double / float);
- `quantidade avaliações` (int);

A classe também deve conter, obrigatoriamente, ao menos, dois construtores (*1. padrão (default); 2. título/autor/ano publicação*), e os métodos *gets*, *sets*, métodos `clone()`, `ler()`, `imprimir()` e `toString()`.

O método `clone()` deve retornar um objeto da mesma classe e contendo os mesmos valores de atributos do atual objeto analisado (**vide exemplo do método clone no slide "Unidade 0 - Nivelamento - Ponteiros (C/C++) e Referências (Java)**" postado na disciplina teórica). **Não é permitido usar a interface Cloneable para esta finalidade.**

O método `ler()` deve receber cada linha do arquivo como parâmetro e armazenar os valores contidos em cada linha nos atributos de cada objeto que foi instanciado.

O método `imprimir()` exibe os valores dos atributos do objeto Livro, conforme o modelo indicado no fim deste documento, conforme mostrado AQUI.

O método `toString()` deve ser criado para permitir a impressão da classe, sem necessidade de invocação do método `imprimir()`.

Após a criação da classe, deve ser criado um mecanismo para processamento de uma entrada de dados. A entrada de dados é dividida em 2 partes:

- Parte 1: Armazenamento de informações em vetor;
- Parte 2: Pesquisa de informações armazenadas no vetor criado na parte 1.

Parte 1 (Leitura de Arquivo): Armazenamento de informações contidos no arquivo livros.txt em vetor

Seu programa deve ler um arquivo-texto chamado **"livros.txt"** que, no VERDE, localiza-se na pasta **"/tmp"**.

O aluno(a) deve preencher um vetor de objetos da classe Livro com os dados dos diversos livros informados no arquivo "livros.txt". Use um ArrayList<Livros> para aumentar dinamicamente o *array* contendo os livros que serão incluídos. Consulte um guia disponível em Java ArrayList ou ArrayList para compreender o funcionamento do *ArrayList*.

Cada uma das linhas seguintes apresenta os dados de um livro, separados pelo símbolo "`|`". Os dados possuem, em ordem, as seguintes informações:

- `ISBN` (long);
- `titulo` (String);
- `autor principal` (String);
- `segundo autor` (String);
- `categoria` (String);
- `descricao` (String);
- `ano publicação` (int);
- `quantidade páginas` (int);
- `nota avaliação` (double / float);
- `quantidade avaliações` (int);

Importante: O caractere "`|`" é utilizado para expressões regulares. Para dividir uma palavra por "`|`", utilize a seguinte sequência: "`\\|`".

A última linha do arquivo **"livros.txt"** é vazia.

Parte 2: Pesquisa de informações armazenadas no vetor (ArrayList) criado na parte 1.

Após o processamento da primeira parte da entrada de dados, o programa deve processar a entrada contida no arquivo *pub.in*. Cada linha da segunda parte contém, em ordem, as seguintes informações:

- Título;
- Ano de lançamento;
- Autor principal.

A última linha do arquivo *pub.in* contém a palavra FIM.

As entradas devem ser pesquisadas no vetor de livros. Armazene as entradas em um outro array, que posteriormente será ordenado.

Parte 3: Ordenação do vetor armazenado na Parte 2.

Nesta etapa será feita a ordenação do vetor de registros pesquisados, utilizando 3 algoritmos de ordenação:

- Bubblesort;
- Selectionsort;
- Insertionsort.

Parte 3.1: Critérios de ordenação. A ordenação de elementos no vetor será feita com base em atributos da classe Livro. Estes atributos devem ser utilizados para comparar se um dado livro é maior/menor que outro. A ordenação dos registros deve ser feita de acordo com os seguintes critérios:

- Categoria (crescente);
- Nota média (decrescente);
- Quantidade de avaliação (decrescente).

Para que os métodos utilizem a mesma entrada, devem ser gerados 3 clones do array original - cada um será usado com um método específico de ordenação.

Após o processamento de um dado método de ordenação, devem ser impressas as informações dos livros ordenados, segundo critério previamente informado.

Para cada livro, deve ser escrito na saída padrão uma linha contendo as informações bibliográficas do livro, com os dados do registro correspondente. Também devem ser exibidas informações correspondentes à categoria do livro, nota média e quantidade de avaliação. A saída padrão deve obedecer o seguinte formato (*não incluir chaves*):

[*categoria*] [*nota*] [*qtd. avaliação*] {*autor principal*}, {*segundo autor*}. {*titulo*}. {*ano publicação*}. ISBN: {*isbn*}.

Exemplos (ordenados):

[*Fiction*] [*4.16*] [*59865*] *Isaac Asimov. The Caves of Steel. 1993. ISBN: 9780586008355.*

[*Fiction*] [*4.05*] [*95008*] *Laura Gibbs. Aesop's Fables. 2002. ISBN: 9780192840509.*

[*Fiction*] [*4.05*] [*50175*] *Albert Camus. The Fall. 1956. ISBN: 9780679720225.*

[*Fiction*] [*4.05*] [*9575*] *Mercedes Lackey. Oathblood. 1998. ISBN: 9780886777739.*

Dicas (não obrigatórias):

- Crie *métodos para comparação de objetos ("ehMaior" e "ehMenor"), contendo os critérios indicados;*
- Crie uma *classe para ordenação, contendo os 3 algoritmos (bubble, selection, insertion);*
- *Altere operações dos algoritmos de ordenação (">" ou "<") para os métodos "ehMaior" e "ehMenor".*

Parte 3.2: Resumo da ordenação.

Durante o processo de ordenação, devem ser armazenadas as seguintes métricas:

- Número de comparações entre livros;

- Número de movimentações (trocas) de posições no vetor.

Ao fim da execução de cada um dos algoritmos de ordenação, deve ser exibido um resumo do número de comparações e movimentações executadas. O resumo deve ser impresso com o seguinte padrão:

`## {nome_metodo} [COMPARACOES] [{num_comparacoes}] [MOVIMENTACOES] [{num_movimentacoes}]`

Onde:

- "NOME-METODO" = BUBBLE ou INSERTION ou SELECTION;
- "{num_comparacoes}" = número de comparações divididas por 1000, sem casas decimais;
- "{num_movimentacoes}" = número de movimentações divididas por 1000, sem casas decimais.

Após cada valor do número de comparações e movimentações, acrescentar a letra "k", para indicar que a contagem de números está em função de 1000.

Exemplo de saída:

```
## BUBBLE [COMPARACOES] [13k] [MOVIMENTACOES] [10k]
```

Após a conclusão do trabalho, observe a quantidade de comparações e movimentações realizadas por cada um dos algoritmos. Analise os resultados e avalie se os mesmos estão de acordo com o esperado, a partir do estudo da disciplina teórica.

Dicas (não obrigatórias):

- *Crie uma classe log, para gerenciamento da contagem de operações;*
- *Conte cada um das cláusulas de comparação (IF).*
- *Quando houver IFs aninhados, não se esqueça de contar as cláusulas anteriores, que já foram avaliadas.*
- *A criação de um método único para comparação ("ehMaior" e "ehMenor") facilita o processo de comparação.*