Pontifícia Universidade Católica de Minas Gerais
Curso de Ciência da Computação
Disciplina: Algoritmos e Estruturas de Dados II
Prof. Felipe Domingos da Cunha / Prof. Max do Val Machado

Trabalho Prático III - Estruturas Sequenciais

1 Regras Básicas

1. extends TP1RegrasBasicas;
2. Todas as classes devem ser entregues em um único arquivo. Essa regra será necessária para a submissão de trabalhos no Verde e no identificador de plágio utilizado na disciplina.
3. Use o padrão Java para comentários.

2 Introdução

Um profissional deve conhecer a história da sua área de atuação. Uma das formas de conhecer a história da Ciência da Computação é através das pessoas importantes na área de Ciências Exatas e Informática.

A coleção de arquivos pessoa.zip contempla informações de pessoas importantes na área de Ciências Exatas e Informática. Essa coleção foi obtida na Wikipédia em 7 de junho de 2016 e sofreu alguns "pequenos" ajustes manuais. Apesar da Wikipédia frequentemente apresentar erros de conteúdo, este trabalho a utiliza porque ela contém alguns caracteres especiais que normalmente tornam a tarefa dos alunos "mais divertida". Os arquivos da coleção usam o padrão de codificação **UTF-8** e estão nomeados como XXX.html, onde XXX corresponde ao identificador (id) de uma pessoa em nossa base de dados. Cada arquivo da coleção contém as seguintes informações de uma pessoa: id, nome, data/local de nascimento e a idade quando a informação foi cadastrada na Wikipédia. Para as pessoas falecidas, temos a data/local da morte e a idade refere ao momento do falecimento. Algumas datas/locais de nascimento/morte são "não conhecida(o)", o termo "ca." indica incerteza de data e "a.C.", antes de Cristo. Cada arquivo tem outras informações que não serão utilizadas neste trabalho. A coleção deve ser descompactada na pasta /tmp/. Quando reiniciamos o Linux, ele normalmente apaga os arquivos existentes na pasta /tmp/.

3 Descrição

1. **Classe Pessoa**: Crie uma classe *Pessoa* seguindo todas as regras apresentadas no slide unidade01g_conceitosBasicos_introducaoOO.pdf. Sua classe terá os atributos privados id (int), nome (String), nacionalidade (String), data de nascimento (String), local de nascimento (String), data de morte (String), local de morte (String) e idade (int). Ela terá também pelo menos dois construtores, e os métodos gets, sets, clone, imprimir e ler. O método imprimir mostra a string "ID nome nacionalidade dataNascimento localNascimento dataMorte localMorte idade", contendo todos os atributos da classe. O método ler deve efetuar um parse do arquivo html para obter os atributos de cada registro. Esse método recebe como parâmetro o nome de um arquivo e, em seguida, ele lê linha a linha esse arquivo para encontrar as informações desejadas. Observa-se que a estrutura "html" de todos os arquivos é similar. As diferenças relevantes para este trabalho foram removidas nos "pequenos" ajustes manuais realizados na coleção.

   A entrada padrão é composta por várias linhas e cada uma corresponde ao nome (e caminho) de um arquivo texto da coleção. A última linha da entrada é a palavra FIM. A saída padrão também contém várias linhas, uma para cada registro contido em uma linha da entrada padrão.

2. **Lista com Alocação Sequencial**: Crie uma Lista de Pessoas baseada na lista de inteiros vista na sala de aula (arquivo Lista.java). Sua lista deve conter todos os atributos e métodos existentes na lista de inteiros, contudo, adaptados para a classe *Pessoa*. De toda forma, lembre-se que, na verdade, temos uma lista de ponteiros e cada um deles aponta para um objeto *Pessoa*. 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(Pessoa pessoa)* insere um objeto na primeira posição da Lista e remaneja os demais. Segundo, o *void inserir(Pessoa pessoa, int posição)* insere um objeto na posição p da Lista, onde $p < n$ e n é o número de objetos cadastrados. Em seguida, esse método remaneja os demais objetos. O *void inserirFim(Pessoa pessoa)* insere um objeto na última posição da Lista. O *Pessoa removerInicio()* remove e retorna o primeiro objeto cadastrado na Lista e remaneja os demais. O *Pessoa remover(int posição)* remove e retorna o objeto cadastrado na p-ésima posição da Lista e remaneja os demais. O *Pessoa removerFim()* remove e retorna o último objeto 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 objetos 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 os atributos do objeto a ser inserido. Esses atributos estão organizados em strings da forma "ID ## nome ## nacionalidade ## dataNascimento ## localNascimento ## dataMorte ## localMorte ## idade". O marcador "##" nunca pertence aos atributos de um registro. No caso dos comandos de "em qualquer posição", temos também a posição. No Inserir, a posição fica antes dos atributos. A saída padrão tem uma linha para cada objeto removido sendo que essa informação será constituída pela palavra "(R)" e o atributo **nome**. No final, a saída mostra os atributos relativos a cada objeto cadastrado na lista após as operações de inserção e remoção.

3. **Pilha com Alocação Sequencial**: Crie uma Pilha de Pessoas 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**: Crie uma classe *Fila Circular* de *Pessoa*. 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 objeto 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 atributo idade dos registros contidos na fila após cada inserção. O final da saída padrão mostra os registros existentes na fila seguindo o padrão da questão anterior.