PMR2300 – Computação para Mecatrônica

Aluno: Bruno Henrique Lobo Netto Peixoto

Nº. USP: 7206666

**Exercício-Programa 3 – 2012**

O exercício-programa solicitado apresenta 4 classes:

1. public class InfoAluno
2. public class BancoDeDadosInfoAluno
3. public class GUI
4. public class EP37206666

# public class EP37206666

# Esta classe contém o main do programa. Ele basicamente cria um objeto do tipo GUI, que é a interface gráfica do programa, e a torna visível. Nesta são realizadas todas as interações com o usuário.

# public class InfoAluno

Esta classe é a célula básica do exercício-programa. Ela contém as informações referentes a um aluno.

* 1. **Atributos**: Primeiro nome, Nome do Meio, Sobrenome Número USP, Endereço, Complemento e CEP do aluno, do tipo String. Todas são do tipo de acesso private.
  2. **Construtor**:
     1. public InfoAluno(): Atribue valor null para os atributos do tipo String do objeto;
     2. protected InfoAluno(String Nome,…,String Cep): Atribue os valores de entrada do métodos aos respectivos atributos do objeto.
  3. **Métodos**: A classe apresenta apenas um tipo de método: o de coleta de atributos. Para simplificar a explicação, será adotado GetX para designar o método responsável pela coleta deste mesmo atributo X, e TIPO para o tipo do atributo, apenas String, neste exercício programa.

public TIPO GetX(): Retorna o valor presente no atributo X do objeto;

# public class BancoDeDadosInfoAluno

# Esta classe contém os atributos e métodos que caracterizam um Banco de Dados.

1. **Atributos**:
   * 1. private TreeMap <Integer,InfoAluno> tMap: Estrutura de dados na qual será armazenado os alunos;
     2. private boolean rearranjo: Variável que detecta se houve rearranjo do array de alunos, promovido pela inserção, deleção ou mescla de alunos no Banco de Dados;
     3. private String[] Alunos: Arranjo de alunos para possibilitar que se percorra o Banco de Dados.
2. **Construtor**:

public BancoDeDadosInfoAluno(): reserva memória para o TreeMap BancoDeDados do tipo TreeMap, atribue true para a variável rearranjo e inicializa o array Alunos.

1. **Métodos**: A classe apresenta métodos de manipulação de um Banco de Dados.
   * 1. public void ResetaBancoDeDados(): reseta a estrutura de dados tMap;
     2. public void InsereAluno(…): Os parâmetros de entrada deste método são os atributos de um objeto do tipo InfoAluno()(Nome do meio,..., CEP). Estes foram omitidos para manter esta documentação pouco poluída;
     3. public boolean RemoveAluno(String NUSP): Retorna true caso encontrar um aluno com número USP NUSP na estrutura tMap e delete-o, ou false caso contrário;
     4. public InnfoAluno BuscaAluno(String NUSP): Realiza uma busca no tMap por um aluno com número USP NUSP e, caso encontre-o, retorna-o como parâmetro de retorno. Caso contrário, o retorno do método é um object null;
     5. public int IndiceBusca(int nusp): retorna o índice de um aluno com número uso nusp, por suposição que este já encontra-se no Banco de Dados;
     6. public boolean ImportaBancoDeDados(String FileName): Retorna true caso importar os alunos do arquivo de extensão .txt com sucesso, ou false caso contrário. Cada linha de leitura do arquivo deve conter os atributos do aluno separados pelo caracter-separador “//”, inevitavelmente. Caso não haja um dos atributos, o arquivo deve ter ////, que significa que não existe nada entre estes campos. Isso implica que devem sempre haver sete espaços de leitura. As possibilidades para o retorno false são: erro na abertura do arquivo; erro na leitura do arquivo;
     7. public boolean ExportaBancoDeDados(String FileName): Retorna true caso exportar o banco de dados para um arquivo .txt externo, ou false caso contrário. A estrutura de cada linha do arquivo exportado é a mesma esperada pelo método ImportaBancoDeDados(), de forma que este método funcione como um backup. As possibilidades para o retorno false são: erro na abertura do arquivo; ou erro na escrita do arquivo;
     8. public int[] Percorrer(int posicaoatual, int botao): possibilita que usuário percorra o Banco de Dados atual a partir de do atual aluno mostrado na TextArea do Frame, avançando ou recuando neste;

# public class GUI

# Esta classe é responsável pela construção da interface gráfica para interação com o usuário.

1. **Atributos**:
   * 1. public static final int LARGURA: valor fixo da largura inicial do Frame;
     2. public static final int ALTURA: valor fixo da altura inicial do Frame;
     3. Componentes que compõem o Frame (Button, Dialog, CheckBox, TextField, TextArea, etc): Os componentes da interface gráfica permitem uma interação com o usuário por meio do tratamento que cada um captura, atribuindo este a uma determinada operação no Banco de Dados;
     4. private BancoDeDadosInfoAluno Database: Banco de dados que conterá os alunos inseridos;
     5. private boolean Iniciar: Array com duas posições para inidicar o N USP e índice do aluno mostrado atualmente no TextArea;
2. **Construtor**:
   * 1. public GUI(): Cria um Frame com Título e cor pré-determinados. Inicializa a variável Iniciar com false e inicializa o Banco de dados Database;
     2. public GUI(String tituloAplicacao, Color corAplicacao): Cria um Frame com Título e cor à escolha, determinados pelos parâmetros de entrada. Inicializa a variável Iniciar com false e inicializa o Banco de dados Database;
3. **Métodos**:
   * 1. private void init(): Posiciona componentes do Frame (por meio de GridBagConstraints) e atribue as devidas propriedades (cor, tamanho, etc) quando possível. Foi o método, juntamente com o seguinte, que exigiu maior número de linhas no exercício-programa;
     2. public void actionPerformed(): Trata eventos ocorridos no Frame, de acordo com o componente à que ele está associado (botão "Buscar" abre caixa de diálogo respectivo, botão "Carregar" abre janela para acessar o arquivo a ser carregado, etc). Para não tornar esta descrição extensa, preferiu-se omitir a explicação minuciosa de cada um dos tratamentos de evento;
     3. public void CaixaDialogo(boolean x): Gera caixa de diálogo para busca ou deleção de alunos, de acordo com o parâmetro de entrada x (boolean);
     4. public String CarregaArq(Frame f, String title, String defDir, String fileType): Gera caixa de diálogo para carregar um banco de dados externo. Os parâmetros de entrada são: Frame em que será aberto (f), título do Dialog (title), diretório de carregamento (defDir) e tipo de arquivo a ser salvo (fileType). O parâmetro de saída é o diretório final do arquivo a ser carregado. Este será usado como parâmetro do método ImportaBancoDeDados da classe BancoDeDadosInfoAluno;
     5. public String SalvaArq(Frame f, String title, String defDir, String fileType): Gera caixa de diálogo para salvar um banco de dados em um arquivo externo. Os parâmetros de entrada são: Frame em que será aberto (f), título do Dialog (title), diretório de carregamento (defDir) e tipo de arquivo a ser salvo (fileType). O parâmetro de saída é o diretório final do arquivo a ser salvo. Este será usado como parâmetro do método ExportaBancoDeDados da classe BancoDeDadosInfoAluno;