ACH2003 - Computação Orientada a Objetos - 2012
Profa: Patrícia Oliveira
Estagiária PAE: Rosiane Santos

Primeiro Exercício-Programa

Instruções:

I) O trabalho deve ser feito **individualmente**.

II) As postagens das soluções devem ser feitas no Col até às **23:59** do dia **30/05/2012**.

III) Se for detectado **plágio**, os alunos copiados e copiadores serão punidos com nota zero nessa atividade.

IV) Trabalhos enviados por email serão completamente **ignorados**.

V) As respostas a cada uma das questões deve possuir um **método `main`** que torne possível testar a solução.

Questão 1) (3,0 pontos) Um objeto **`Data`** é caracterizado pela seguinte interface:

```
interface Data {

      class ExcecaoData extends Exception{
            ExcecaoData (){
                  super("Tentativa de geração de data invalida" );
            ExcecaoData (String mensagemErro){
                  super(mensagemErro);
            }
      }

      void setData(int ano, int mes, int dia) throws ExcecaoData;
      int getAno();
      int getMes();
      int getDia();
}
```

Note que é possível que uma interface declare uma classe local. Dessa forma, a classe **`ExcecaoData`** pode estar disponível para todas as classes que implementam a interface **`Data`**.

Crie uma classe **`MinhaData`** que implemente a interface **`Data`**. Essa classe deve possuir três atributos inteiros privados: **`ano`**, **`mes`** e **`dia`**. Deve existir, ainda, um construtor que tome três parâmetros (valores para ano, mês e dia) e que internamente faça uso do método **`setData(...)`**. Determine como a exceção **`ExcecaoData`** deve ser tratada pelo construtor. Você deve ignorar os anos bissextos, assumindo que há sempre 28 dias em fevereiro.

Escreva uma classe **ClienteData** que contenha um método **main** para testar a classe **MinhaData.** Para isso, deve ser implementada uma interface de entrada/saída com o usuário para receber os valores de parâmetros de uma data e retornar, na tela, a saída desses dados formatada como “dia/mês/ano”. Inclua, ainda, o tratamento de erros para quando os dados de entrada forem de tipos diferentes de inteiros (**int**). As mensagens de erro para esse tratamento devem apontar qual o campo (dia, mês, ano) teve o seu tipo de dado fornecido incorretamente.

Não é necessário que a interface seja gráfica, mas aqueles que o fizerem terão 0.5 ponto de acréscimo nessa questão.

Questão 2) (2,0 pontos) Modifique o código da questão anterior para que as mensagens de erro sobre datas inválidas sejam mais informativas. Por exemplo, para a tentativa de criar a data 45/03/2009, é adequado informar que o campo errado é o referente ao valor de dia. Para isso, é necessário modificar a classe **ExcecaoData** de modo que ela retenha a informação de que campo está errado:

```
      class ExcecaoData extends Exception{

            public final String campo;
//armazena a informação de campo errado para gerar uma mensagem mais informativa.

            ExcecaoData (){
                  super(“Tentativa de geração de data invalida” );
            ExcecaoData (String campo){
                  super(“A data estah invalidada pelo campo” + campo);
            }
      }
```

Insira a modificação acima na interface **Data** e promova as mudanças necessárias a sua utilização na classe **MinhaData** de modo que o mecanismo de tratamento de erros possa fornecer as mensagens precisas discutidas acima.

Questão 3) (3,0 pontos) Escreva um programa que leia um arquivo de caracteres e “quebre” o texto nele contido em palavras (o nome/caminho para o arquivo deve ser fornecido pelo usuário). Para essa tarefa, você deve, primeiramente, utilizar um objeto **StringTokenizer** para separar os tokens e em seguida trabalhar com os métodos da classe **String** para remover eventuais símbolos de pontuação que tenham ficado juntos às palavras presentes no texto. Após essa etapa, cada palavra deve ser adicionada a uma coleção **LinkedList** (as eventuais ocorrências múltiplas devem ser mantidas). Por fim, utilize um objeto **ListIterator** para imprimir o conteúdo da lista na tela, na ordem reversa em que foram inseridos, e o total de palavras contidas no texto. Não esqueça de implementar o mecanismo de tratamento de exceções.

Não é necessário que a interface seja gráfica, mas aqueles que o fizerem terão 0.5 ponto de acréscimo nessa questão.

Questão 4) (2,0 pontos) Desenvolva um programa que amplie a funcionalidade do programa implementado na questão anterior, fornecendo um método para remover da lista ligada construída na Questão 3 palavras geralmente consideradas muito frequentes e com pouco significado para tarefas de mineração de texto (tais como artigos, preposições, algumas conjunções). Essas palavras (denominadas de *stopwords*) estão contidas em um arquivo texto dado como anexo a essa especificação de Exercício-Trabalho. Como saída, o aplicativo deve imprimir, na tela:

- quantas *stopwords* foram removidas e quantas palavras foram mantidas na lista resultante (sem contar eventuais repetições).

- as palavras, uma por linha, em ordem alfabética. Cada palavra deve aparecer exatamente uma vez, acompanhada do número de ocorrências destas no texto original.

Não esqueça de implementar o mecanismo de tratamento de exceções.