Universidade Federal do Espírito Santo
Centro Tecnológico
Departamento de Informática

Programação III / PAC
(INF 09331 / INF 09324)
Prof. Vítor E. Silva Souza

Curso Básico Java – Lista de Exercícios 3 – Partes 8 a 10

1. Exceções

1) Dado o trecho de código abaixo:

```
int[] vetor = new int[] { 2, 4, 6, 8, 10, 12 };
for (int i = 0; i <= 12; i++) {
     System.out.println(vetor[i]);
}
```

Implemente um programa em Java que execute este trecho. Execute o programa e veja a exceção produzida por acesso a posição fora dos limites do vetor. Trate esta exceção, imprimindo na tela uma mensagem dizendo que o vetor acabou.

2) Dado o trecho de código abaixo:

```
public static void metodo01() {
     Class.forName("ClasseQueNaoExiste");
}
public static void metodo02() {
     java.io.File.createTempFile("pre", "suf");
}
public static void metodo03() {
     Integer.class.newInstance();
}
public static void main(String[] args) {
     metodo01();
     metodo02();
     metodo03();
}
```

Uma classe com estes quatro métodos não compila. Quais passos são necessários para fazê-la compilar? Altere o código para que a classe compile sem erros.

3) Modifique a classe que representa a conta-corrente do exercício 2 da lista de exercícios 2, lançando uma exceção (criada por você) caso o usuário tente depositar ou sacar um valor negativo. Adicione uma outra exceção (também criada por você) caso o usuário tente sacar um valor maior do que o saldo da conta dele. Trate as exceções no seu programa principal, exibindo mensagens de erro adequadas.

2. Arquivos e fluxos

4) Uma loja possui 4 filiais, cada uma com um código de 1 a 4. Um arquivo contendo todas as vendas feitas durante o dia nas quatro filiais é gerado a partir de uma planilha, sendo que cada linha do arquivo contém o número da filial e o valor de uma venda efetuada, separados por vírgula. Ex.:

```
1,189.90
1,45.70
3,29.90
4,55.00
```

No exemplo acima, a filial 1 fez duas vendas, a primeira totalizando R$ 189,90 e a segunda R$ 45,70. A filial 3 fez uma venda de R$ 29,90 e a 4 também uma de R$ 55,00. Faça um programa que leia este arquivo e informe, ao final, o total e o valor médio das vendas de cada filial.

5) No exercício 8 da lista de exercícios 1 foram criados objetos representando empresas, departamentos e funcionários. Divida o exercício em 2 partes. Na primeira os objetos devem ser criados e serializados em um arquivo no disco. Na segunda, o arquivo deve ser lido, os objetos restaurados e as operações (aumento de 10%, transferência de departamento) efetuadas.

6) Crie um programa que liste o conteúdo da raiz do seu disco (ou um diretório qualquer, caso você não tenha acesso à raiz) e informe, para cada item, se é um arquivo ou se é um diretório.

7) Escreva um programa que receba um nome de arquivo e uma sequência de palavras como argumentos na linha de comando, informe se o arquivo existe ou não, caso não exista, crie-o e, por fim, escreva neste arquivo a sequência de palavras passadas como argumentos.

3. Utilidades

8) Escreva um programa que informe em que dia da semana cairá uma determinada data. Dica: investigue a API da classe `SimpleDateFormat`.

9) Escreva um programa que, dado um número de dias "x" como parâmetro, imprima: "Daqui a x dias será dia DIA/MÊS/ANO (DIA DA SEMANA)". Imprima o ano com 4 dígitos e o dia da semana por extenso.

10) Altere o programa do exercício anterior para imprimir o dia da semana em uma língua diferente (português, inglês, francês, espanhol, etc.). Dica: você pode tanto alterar o *locale default* antes de criar o formatador de datas quanto passar o *locale* no construtor do formatador de datas.

11) Crie um programa que receba como parâmetro um valor em reais e converta para dólares (Estados Unidos) e yenes (Japão). Use um formatador de números apropriado para imprimir o resultado, levando o *locale* em consideração. Se não tiver como obter a cotação do dia, use R$ 1 = US$ 0,30 e R$ 1 = ¥ 30,00.

12) Adicione a sequência de números 2, 5, 3, 9, 2, 4, 3, 8, 5 a um conjunto (`Set`) e a uma lista (`List`), escolhendo a implementação que desejar. Em seguida imprima o conteúdo de ambas as coleções usando um iterador e analise as diferenças. Substitua o iterador por um for-each e analise a diferença entre os códigos (tamanho, legibilidade, etc.).

13) Escreva uma aplicação de dicionário com três funções: adicionar um termo ao dicionário, procurar um termo no dicionário e listar todos os termos existentes em ordem alfabética. Qual classe você usou para implementar a coleção de palavras? Por quê?

14) Escreva um programa que leia nomes, idades e alturas de várias pessoas (do teclado ou de um arquivo, como preferir) e armazene numa lista. Em seguida, imprima o conteúdo desta lista ordenado por nome, depois ordenado por idade e por fim ordenado por altura.

15) No exercício 5 da lista de exercícios 2, foi criada uma aplicação de agenda. Incremente-a da seguinte forma:

a) Faça com que a aplicação grave todos os contatos criados num arquivo ao ser encerrada e que leia este arquivo ao ser iniciada (sugestão: use serialização);

b) Atualize o código de toda a aplicação para as novas facilidades da versão 5 e 7 de Java. Por exemplo, faça com que as coleções usem os tipos genéricos, substitua loops for e iteradores pelo *for-each* (`for (o : coleção) { }`), use `try` com recursos, etc.