Ministério da Educação
Universidade Federal de Ouro Preto (Ufop)
Departamento de Computação e Sistemas (Decsi)

Professor: Rafael Alexandre
rfalexandre@ufop.edu.br
Disciplina: CSI032

Trabalho Prático 07 (TP07)

Instruções:

i - O arquivo deve ser entregue em formato .ZIP ou .RAR seguindo a nomenclatura: "XXXX.KKK" onde XXXX é o número de sua matrículo e KKK a extensão do arquivo.

ii - Cada um dos exercícios deve criado em um diretório com o seguinte nome: Exercicio_XX onde XX é o número da questão solucionada.

iii - Para cada programa desenvolvido deverão ser entregues **SOMENTE** os arquivos de projeto e classes Java em seus respectivos pacotes.

iv - O arquivo deve ser enviado via moodle limitado a data e hora de entrega definida no Plano de Ensino. Não serão aceitos trabalhos enviados por e-mail.

.

Questão 1. Considere um objeto Set:

(a) Escreva um programa que lê uma série de nomes e elimina as duplicatas armazenando-os em um Set;

i) O usuário deve poder remover um determinado nome, se o nome não pertencer ao conjunto, uma mensagem de erro deve ser informada;

ii) O usuário deve poder verificar se um determinado nome já foi armazenado;

iii) Por fim, um usuário pode apagar todos os nomes armazenados;

(b) Escreva um programa que armazena produtos de um supermercado em um Set, cada produto possui um código de barras (ID), nome e preço;

i) o usuário deve informar os dados do produto a ser armazenado, se o produto já existir no sistema, um aviso deve ser exibido. Dois produtos são iquais se e somente se possuírem o mesmo ID;

ii) Percorra o conjunto e exiba os dados de todos os produtos cadastrados.

Questão 2. Considere um Map:

(a) Faça um mapa que associe um ID de um empregado a seu nome. Chaves e valores de um mapa podem ser Objetos de qualquer tipo, mas para facilitar o exercício defina as chaves e valores como String.

Tabela 1: Identificadores de funcionários.

ID	Nome
a1234	Steve Jobs
a1235	Scott McNealy
a1236	Jeff Bezos
a1237	Larry Ellison
a1238	Bill Gates

(b) Crie teste onde são recuperados e impressos os funcionários pelo seus respectivos identificadores. Tente recuperar um funcionário cujo ID não está mapeado, observe o resultado.

(c) Faça um método de busca no qual as chaves funcionam para qualquer caso. Exemplo: as chaves a1234 e A1234 devem retornar o funcionário Steve Jobs.

(d) Altere o programa para que um funcionário seja representado por uma instância da classe funcionário (atributos: nome, cargo, salário). Uma vez recuperado o funcionário pelo seu ID, exiba suas informações;

(e) Percorra o mapa e exiba todos os funcionários cadastrados e suas respectivas chaves.

Questão 3. Considere as Interfaces Comparable e Comparator

(a) Crie um ArrayList de inteiros e armazene 50 números aleatórios pertencentes a um intervalo de [0,99]. Realize as seguintes operações sobre a coleção de números:

i) Obtenha o menor e o maior número armazenado;

ii) Conte quantas vezes o menor número apareceu na coleção;

iii) Ordene a lista em ordem crescente;

iv) Utilize as soluções disponibilizadas no framework Collections para realizar essas operações.

(b) No sistema de uma empresa, cada empregado possui um ID, nome, salário e data em que ele foi contratado. Crie uma classe de testes para seu programa, o teste deve conter um ArrayList de empregados e implementar as seguintes funcionalidades:

i) ordenar (ordem crescente) os funcionários pelo salário;

ii) ordenar os funcionários em ordem alfabética;

iii) exibir qual funcionário possui o maior salário e qual possui o menor;

iv) exibir o funcionário mais experiente (mais antigo) e o menos experiente;

v) Utilize as soluções disponibilizadas no framework Collections para realizar essas operações.

Questão 4. O código da Figura 1 lança uma exceção (propositalmente) e interrompe sua execução. Utilizando o tratamento de exceções, corrija a classe com o objetivo de não parar sua execução. OBS: A Exception lançada é ArrayIndexOutOfBoundsException.

```
public class TesteException {

    public static void main(String[] args) {
        System.out.println("inicio do main");
        metodo1();
        System.out.println("fim do main");
    }

    static void metodo1() {
        System.out.println("inicio do metodo1");
        metodo2();
        System.out.println("fim do metodo1");
    }

    static void metodo2() {
        System.out.println("inicio do metodo2");
        int[] array = new int[10];
        for (int i = 0; i <= 15; i++) {
            array[i] = i;
            System.out.println(i);
        }
        System.out.println("fim do metodo2");
    }
}
```

Figura 1: Classe TesteException.

Questão 5. Nesta questão, considerando o código apresentado pela Figura 2, você deve identificar as partes problemáticas do código e reescrevê-lo utilizando tratamento de exceções. Ou seja, devem ser identificadas todas as exceções que podem ser levantadas e, para cada uma, deve ser dado o tratamento adequado que, nesse exercício, significa alertar o usuário quanto ao problema. Entretanto, nesse programa a leitura dos valores deve ser feita, mesmo que para isso o usuário tenha que tentar informar várias vezes os valores na mesma execução do programa.

```java
public class Questao2 {

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        System.out.println("Eu sei dividir!");
        System.out.print("Informe o primeiro valor: ");
        int x = teclado.nextInt();
        System.out.print("Informe o segundo valor: ");
        int y = teclado.nextInt();
        double r = (x / y);
        System.out.println("O resultado da soma é " + r);
    }
}
```

Figura 2: Classe Questao2.

Questão 6. Considerando o código ilustrado pela Figura 3, suponha que o método ”saca”da classe Conta vai ser rescrito de forma a lançar uma exceção criada por você, cuja classe é ContaExcecao (extends Exception). A exceção é lançada sempre que o saldo da conta for inferior ao valor sacado. Implemente a classe ContaExcecao. Implemente o método saca que lança a exceção. E rescreva o código da caixa com o devido tratamento da exceção.

```java
Conta minhaConta = new  Conta();
minhaConta.deposita(100 );
minhaConta.setLimite(100 );
minhaConta.saca(1000 );
```

Figura 3: Código referente à questão 3.

Questão 7. Retomando o exercício anterior, suponha que quando lançada a exceção ContaExcecao, através do objeto exceção instanciado, seja possível recuperar o saldo da pessoa. Como você implementaria isso? Mostre tudo que deve ser modificado/acrescentado no exercício para que isto funcione.

Questão 8. Crie um programa com um método que demonstra que um método com seu próprio

bloco try não precisa capturar todos os possíveis erros gerados dentro deste bloco. Algumas exceções podem ser deixadas de lado, e serem capturadas em outros escopos. Imprima a stack trace desta exceção;

Questão 9. Crie um método que ilustra o relançamento (encadeamento) de uma exceção. Crie os métodos someMethod e someMethod2. O método someMethod2 deve inicialmente lançar uma exceção. O método someMethod deve invocar o método someMethod2, capturar a sua exceção e relança-la. Invoque o método someMethod a partir da main e captura a exceção relançada. Imprima a stack trace desta exceção.

Questão 10. Implemente as atividades a seguir:

(a) Crie uma classe chamada Calculator, que contém os métodos

i) double div(double, double);

ii) double log10(double).

(b) Crie uma classe do tipo Exception, para tratamento de erros das operações em Calculator.

i) InvalidOperationException.

(c) Os métodos devem lançar objetos de exceção e deixar que o programa que usa a calculadora se responsabilize por tratá-los.

(d) Crie um aplicativo (classe com método main) para testar sua calculadora

i) Crie uma instância da calculadora (ou defina os métodos estáticos);

ii) Leia do usuário a operação que deseja realizar e os operandos;

iii) Trate as exceções. Escreva na tela a mensagem de erro gerada e imprima a pilha de chamadas do runtime (stack trace).

(e) Altere sua classe InvalidOperationException

i) Ao invés de herdar de Exception, agora ela deve herdar de RuntimeException;

ii) Remova o tratamento de erro do aplicativo;

iii) O que acontece agora com o programa quando operações inválidas surgem?