**-------------------------------------------------------------------------------------------------------**

Slides\_10\_excecões.pdf

**-------------------------------------------------------------------------------------------------------**

**Slide 06** - implementar try com recursos (cerca de 12 min)

**Slide 07** - implementar throw e trhows (cerca de 15 min)

**==================== fim aula 07 (de 08) t6 ====================**

Implementar primeira parte do "Exercício M F - lançar execption.txt"

-----------------

Implemente um programa que verifique se uma letra digitada é igual a "F" ou "M",

conforme a letra escrever "F - Feminino", "M - Masculino" ou "Sexo inválido"

caso seja digitado uma letra diferente das duas citadas

(pesquisar no google, "comparar String java")

no caso de "sexo inválido" lançar um Exception

throw new Exception("Sexo inválido");

-----------------

**Slide 09** - implementar criar própria exceção e incluir no ex anterior (cerca de 15 min)

Concluir implementação "Exercício M F - lançar execption.txt"

-----------------

Agora, no caso de "sexo inválido" lançar sua própria exceção: "SexoInvalidoExeption()"

>> throw new SexoInvalidoExeption();

public class SaldoInsuficienteException extends Exception {

public SaldoInsuficienteException() {

super("Saldo insuficiente.");

}

public SaldoInsuficienteException( String mensagem ) {

super(mensagem);

}

}

-----------------

**-------------------------------------------------------------------------------------------------------**

Slides\_11\_java\_collections.pdf

**-------------------------------------------------------------------------------------------------------**

**Slide 04 -** implemetar

***Arquivo: TesteArrayList.java***

**Slide 05 -** implemetar

use o map.get(chave) para recuparar os valores

- ***Arquivo Detalhes.java*** (15 min)

***Arquivo: ContainsKey.java***

**Slide 07 –**

implemetar do slide

**Slide 08 –**

implemetar do slide

**Slide 09 -** implemetar

- crie outros ArrayLists que receba outros tipois de dados (Integer, Long, Double)

crie um hashMap com chave do tipo String e valor do tipo Integer e

adicione alguns elementos nessa coleção de dados

percorra e exiba seus valores

Exemplo para percorrer um HashMap()

for (String chave : mapa.keySet()){

**Slide 11 -** implemetar

"sort reverse java":



--------

hasMap

Implementation: ArrayList implements List Interface while HashMap is an implementation of Map interface.

*exercício: crie uma classe com um método que retorne o nome da pessoa buscando pelo número de CPF*

*cadastre as pessoas em um hashMap com chaves sendo o CPF (String) e os valores sendo o Nome*  (20 min)

**import** java.util.HashMap;

**import** java.util.Map;

**public** **class** CPFUtil {

**public** String retornaNomePorCPF(String cpf) {

Map<String, String> map = **new** HashMap<String, String>();

map.put("842.096.232-53", "Fábio Alves Gomes");

map.put("222.096.222-53", "Fábio Alves Gomes");

map.put("111.096.333-53", "Fábio Alves Gomes");

**if**(map.containsKey(cpf)) {

**return** map.get(cpf);

}**else** {

**return** "CPF não encontrado";

}

}

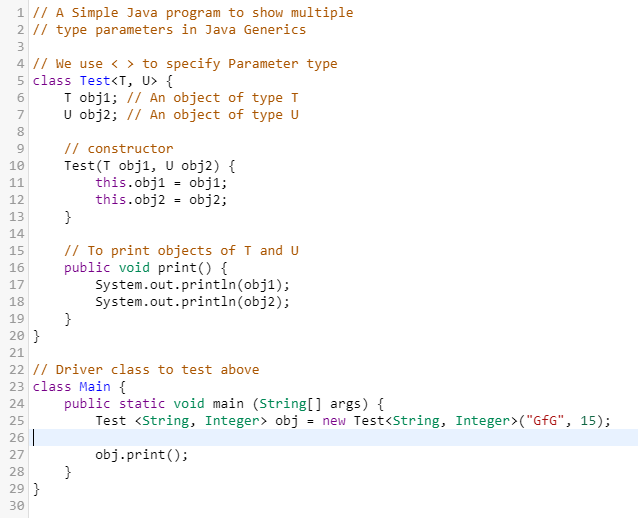
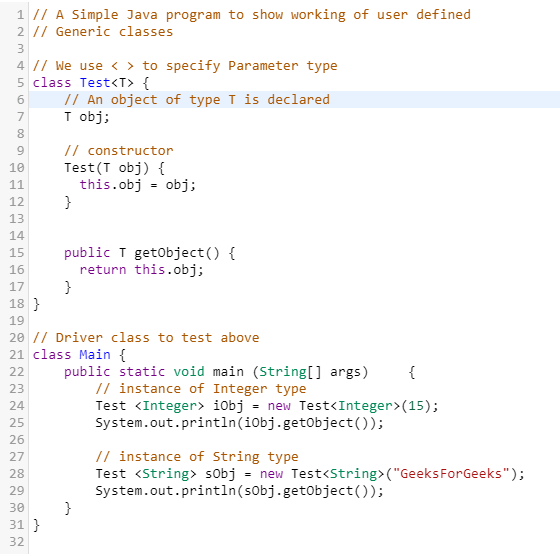
}

se não existir a pessoa cadastrada exibir mensagem "pessoa não encontrada nos registros do hashMap"

**-------------------------------------------------------------------------------------------------------**

slides\_12\_tipos\_genericos.pdf

**-------------------------------------------------------------------------------------------------------**



**-------------------------------------------------------------------------------------------------------**

java\_oo\_slides\_13\_lendo\_e\_escrevendo\_arquivos\_pages\_deleted.pdf

**-------------------------------------------------------------------------------------------------------**

***Implementar arquivo: Nio2.java***