GABARITO - PROVA 1 - PROGRAMAÇÃO PARA WEB I

QUESTÃO 1

```
package br.edu.ifrs.contatos;
public class Endereco {
    private String logradouro;
    private String complemento;

    public Endereco(){}
    public Endereco(String logradouro, String complemento) {
        this.logradouro = logradouro;
        this.complemento = complemento;
    }

    public String getLogradouro() {
        return logradouro;
    }

    public void setLogradouro(String logradouro) {
        this.logradouro = logradouro;
    }

    public String getComplemento() {
        return complemento;
    }

    public void setComplemento(String complemento) {
        this.complemento = complemento;
    }

    public void listar(){
        System.out.println(toString());
    }
    @Override
    public String toString() {
        return "Logradouro=" + logradouro + "\nComplemento=" +
complemento ;
    }

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

package br.edu.ifrs.pessoas;
public class PessoaJuridica {

    private String cnpj;
    private String razaoSocial;

    public PessoaJuridica() {
    }

    public PessoaJuridica(String cnpj, String razaoSocial) {
        this.cnpj = cnpj;
        this.razaoSocial = razaoSocial;
    }

    public String getCnpj() {
        return cnpj;
    }
```

```
    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

    public String getRazaoSocial() {
        return razaoSocial;
    }

    public void setRazaoSocial(String razaoSocial) {
        this.razaoSocial = razaoSocial;
    }

    @Override
    public String toString() {
        return "Cnpj=" + cnpj + "\nRazão Social=" + razaoSocial ;
    }

    public void listar() {
        System.out.println(toString());
    }

}

------------
package br.edu.ifrs.pessoas;

import br.edu.ifrs.contatos.Endereco;

public class Fornecedor extends PessoaJuridica{
    private String nomeContato;
    private Endereco endereco;

    public Fornecedor(){}
    public Fornecedor(String nomeContato, Endereco endereco, String
cnpj, String razaoSocial) {
        super(cnpj, razaoSocial);
        this.nomeContato = nomeContato;
        this.endereco = endereco;
    }

    public String getNomeContato() {
        return nomeContato;
    }

    public void setNomeContato(String nomeContato) {
        this.nomeContato = nomeContato;
    }

    public Endereco getEndereco() {
        return endereco;
    }

    public void setEndereco(Endereco endereco) {
        this.endereco = endereco;
    }

    @Override
    public String toString() {
        return super.toString() + "\nNome Contato=" + nomeContato +
```

```
                (endereco==null?"\nEndereço
inválido":"\nEndereço:"+endereco.toString());
    }

    public void listar() {
        System.out.println(toString());
    }
}

------------
package br.edu.ifrs.produtos;

import br.edu.ifrs.pessoas.Fornecedor;
import java.util.LinkedList;

public class Produto {
    private long codBarras;
    private String nome;
    private double valor;
    private LinkedList<Fornecedor> fornecedores;

    public Produto(){}
    public Produto(long codBarras, String nome, double valor,
LinkedList<Fornecedor> fornecedores) {
        this.codBarras = codBarras;
        this.nome = nome;
        this.valor = valor;
        this.fornecedores = fornecedores;
    }

    public long getCodBarras() {
        return codBarras;
    }

    public void setCodBarras(long codBarras) {
        this.codBarras = codBarras;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public double getValor() {
        return valor;
    }

    public void setValor(double valor) {
        this.valor = valor;
    }

    public LinkedList<Fornecedor> getFornecedores() {
        return fornecedores;
    }

    public void setFornecedores(LinkedList<Fornecedor> fornecedores) {
        this.fornecedores = fornecedores;
    }
```

```
    @Override
    public String toString() {
        String aux = "\nFornecedores não informados";
        if(fornecedores!=null){
            aux = "\nForncedores:";
            for (Fornecedor fornec : fornecedores) {
                if(fornec != null)
                    aux += fornec.toString() + "\n";
            }
        }
        return "Código de Barras=" + codBarras + "\nNome=" + nome +
"\nValor=" + valor + aux;
    }

    public void listar() {
        System.out.println(toString());
    }

}
```

QUESTÃO 2

```
package br.edu.ifrs.testes;

import br.edu.ifrs.contatos.Endereco;
import br.edu.ifrs.pessoas.Fornecedor;
import br.edu.ifrs.produtos.Produto;
import java.util.LinkedList;

public class TesteQ2 {
    public static void main(String[] args) {
        Produto p = new Produto();
        p.listar();

        LinkedList<Fornecedor> fornec = new LinkedList<>();
        fornec.add(new Fornecedor());
        fornec.add(new Fornecedor("nomeContato1", new Endereco(),
"cnpj1", "razaoSocial1"));
        fornec.add(new Fornecedor("nomeContato2", new
Endereco("logradouro2", "complemento2"), "cnpj2", "razaoSocial2"));
        Produto p2 = new Produto(1234, "nomeproduto1", 100.0, fornec);
        p.listar();
    }
}
```

QUESTÃO 3

```
Path arquivo2 = FileSystems.getDefault().getPath("C:", "Diretorio",
"Teste").resolve("Teste.txt");

new BufferedOutputStream(new FileOutputStream( new File("Q1.txt")
)).write( "questao1_prova".getBytes());

BufferedReader leitura = new BufferedReader(new FileReader(new
File("Q1.txt")));
```

QUESTÃO 4

```
Exception in thread "main" java.lang.IndexOutOfBoundsException: Index:
3, Size: 2

      at java.util.LinkedList.checkPositionIndex(LinkedList.java:560)

      at java.util.LinkedList.add(LinkedList.java:507)

      at excecoes.Questao1.main(Questao1.java:18)

Java Result: 1
```

4.1 Qual o nome da classe em que foi gerada a exceção? Questao1
4.2 Qual o nome do método em que a exceção foi gerada? main
4.3 Qual a linha que gerou a exceção? 18
4.4 Qual exceção foi gerada? `IndexOutOfBoundsException`
4.5 Qual a causa da exceção gerada? O código tentou acessar o índice 3, mas o tamanho da LinkedList era 2

QUESTÃO 5

```
import java.io.*;

public class Questao5 {
  public static void main(String[] args) {
      FileReader in = null;
      BufferedReader buff = null;
      try{
         in = new FileReader("teste.txt"); //caminho do arquivo
         buff = new BufferedReader(in, 1024);
         StringBuilder builder = new StringBuilder();
         String s = null;
         while ((s = buff.readLine()) != null) {
             builder.append(s).append("\n");
            }
         System.out.println("Conteudo do arquivo:\n\n"+builder);
    }catch(FileNotFoundException e){
          System.out.println("ARquivo de leitura não existe!");
    }catch(IllegalArgumentException e){
        System.out.println("Argumento inválido para arquivo de
leitura");
    }catch(IOException e){
        System.out.println("Exceção ao escrever no arquivo");
    }finally{
       try{
         if(buff != null) buff.close();
         if(in!=null) in.close();
```

```
        }catch(IOException e){
          System.out.println("Exceção ao fechar arquivo");
        }
      }
    }
}
```

QUESTÃO 6

```
import java.sql.*;

public class Questao6 {
    public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
         String query = "select nome, cpf from clientes";
         String urlDB = "jdbc:mysql://localhost:3306/testbd";

         try (Connection con = DriverManager.getConnection(urlDB,
                                                  "user", "user");
               Statement stmt = con.createStatement();) {
             ResultSet rs = stmt.executeQuery(query);
             while (rs.next()) {
                 String nome = rs.getString("nome");
                 String cpf = rs.getString("cpf");
                 System.out.printf("Nome:%s\t Cpf:%s %n", nome, cpf);
             }
         } catch (SQLException e) {
             System.out.println("Exceção no SQL");
             System.out.println("Causa:" + e.getSQLState());
         }
     } catch (ClassNotFoundException e) {
         System.out.println("Driver não encontrado");
     }
  }
}
```

QUESTÃO 7

Copie o código, execute conforme as instruções e responda os itens usando SUAS PALAVRAS.

QUESTÃO 8

```
import java.util.Set;
public interface OperacoesMapa<K, V> {
    public void adicionar(K chave, V valor);
    public Set<K> getChaves();
    public V getValor(K chave);
    public V remover(K chave);
    public boolean substituir(K chave, V valorVelho, V valorNovo);
    public String toString();
}
```

QUESTÃO 9

```
import java.util.HashMap;
import java.util.Set;

public class MeuMapa<K, V> implements OperacoesMapa<K, V>{
    private HashMap<K, V> mapa = new HashMap<>();
    @Override
    public void adicionar(K chave, V valor) {
        mapa.put(chave, valor);
    }

    @Override
    public Set<K> getChaves() {
        return mapa.keySet();
    }

    @Override
    public V getValor(K chave) {
       return mapa.get(chave);
    }

    @Override
    public V remover(K chave) {
      return mapa.remove(chave);
    }

    @Override
    public boolean substituir(K chave, V valorVelho, V valorNovo) {
        return mapa.replace(chave, valorVelho, valorVelho);
    }

    @Override
    public String toString(){
        String aux = "";
        Set<K> chaves = mapa.keySet();
        for(K chave : chaves)
           aux += "chave:"+chave + " valor:" + mapa.get(chave) + "\n";
        return aux;
    }

}
```

QUESTÃO 10

```
  public void adicionar(K chave, V valor) throws NullPointerException{
        mapa.put(chave, valor);
    }

    @Override
    public boolean substituir(K chave, V valorVelho, V valorNovo) {
        //return mapa.replace(chave, valorVelho, valorVelho);
        throw new NullPointerException();
    }
```

QUESTÃO 11

```
package br.edu.ifrs.testes;
public class TestesMapa {
    public static void main(String[] args) {
        MeuMapa<Integer, String> dados = new MeuMapa<>();
        try{
            dados.adicionar(1, "um");
            dados.adicionar(2, "dois");
            dados.adicionar(3, "tres");
        }catch(NullPointerException e){
            System.out.println("Exceção ao adicionar");
        }
        try{
            dados.substituir(3, "tres", "três");
        }catch(NullPointerException e){
            System.out.println("Exceção ao substituir");
        }
        dados.remover(2);
        System.out.println(dados.toString());

    }
}
```

QUESTÃO 12

```
public class Produto implements Comparable<Produto> {

// outras definições da classe

@Override
    public int compareTo(Produto obj) {
        if(this.codBarras < obj.getCodBarras())
            return -1;
        if(this.codBarras > obj.getCodBarras())
            return 1;
        return 0;
    }
}
```

QUESTÃO 13

```
package br.edu.ifrs.testes;

import br.edu.ifrs.contatos.Endereco;
import br.edu.ifrs.pessoas.Fornecedor;
import br.edu.ifrs.produtos.Produto;
import java.util.LinkedList;
import java.util.TreeSet;

public class TestesProduto {
    public static void main(String[] args) {
        TreeSet<Produto> arvore = new TreeSet<>();

        LinkedList<Fornecedor> fornec = new LinkedList<>();
        fornec.add(new Fornecedor());
        fornec.add(new Fornecedor("nomeContato1", new Endereco(),
"cnpj1", "razaoSocial1"));
        fornec.add(new Fornecedor("nomeContato2", new
Endereco("logradouro2", "complemento2"), "cnpj2", "razaoSocial2"));
        Produto p1 = new Produto(1234, "nomeproduto1", 100.0, fornec);
        p1.listar();
        arvore.add(p1);
        Produto p2 = new Produto(1235, "nomeproduto2", 101.0, fornec);
        arvore.add(p2);

        for (Produto p : arvore) {
             p.listar();
        }
    }
}
```

QUESTÃO 14

Faça o que pede o exercício usando os códigos desenvolvidos na aula.