Código-fonte: Processar.java

```
package splitdados;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

public class Processar {

    private File arquivo;
    private String cabecalho;
    private String treinamento;
    private String teste;
    private int ultimaLinhaCabecalho;
    private int nLinhas;
    private int nAmostras;

    public void processar(File arquivo, String aTeste, String aTreinamento, int pTeste,
int pTreinamento, boolean aleatorio) {
        try {
            // Obtém as informações do arquivo
            this.arquivo = arquivo;
            nLinhas = numeroLinhas();
            salvaCabecalho();
            nAmostras = nLinhas - ultimaLinhaCabecalho;
            int meio = 0;

            // Valida as entradas
            if (aTeste.length() < 3 || aTreinamento.length() < 3 || pTeste  < 1 || pTeste
> 100 || pTreinamento < 1 || pTreinamento > 100)
                return;

            // Se não é aleatório
            if (!aleatorio) {

                // Define os intervalos
                int inicioTeste = ultimaLinhaCabecalho + 1;
                int fimTeste = inicioTeste + ((nAmostras * pTeste) / 100);
                int inicioTreinamento = fimTeste + 1;
                int fimTreinamento = nLinhas;

                // Obtém os dados
                teste = lerIntervalo(inicioTeste, fimTeste);
                treinamento = lerIntervalo(inicioTreinamento, fimTreinamento);

                /*
                System.out.println("Aleatorio: NAO\ninicioTeste = " + inicioTeste +
"\nfimTeste = " + fimTeste + "\ninicioTreinamento = " + inicioTreinamento +
"\nfimTreinamento = " + fimTreinamento + "\n");
                System.out.println("Linhas Treinamento: " + linhasString(treinamento) + "
- Linhas Teste: " + linhasString(teste));
                */

            // Se é aleatório
            } else {
```

```
                int numero, linhasPorcentagem;

                do {
                    // Gera um número aleatório
                    numero = ultimaLinhaCabecalho + 1 + (int)(Math.random() * ((nLinhas -
ultimaLinhaCabecalho + 1) + 1));
                    // Calcula a quantidade de linhas para Teste
                    linhasPorcentagem = (nAmostras * pTeste) / 100;
                } while ((numero + linhasPorcentagem) > nLinhas);

                // Define os intervalos
                int inicioTeste = numero;
                int fimTeste = numero + linhasPorcentagem;
                int inicioTreinamento=0, fimTreinamento=0, inicioTreinamento1=0,
fimTreinamento1=0, inicioTreinamento2=0, fimTreinamento2=0;

                // Se está no início
                if (inicioTeste == ultimaLinhaCabecalho + 1) {
                    inicioTreinamento = fimTeste + 1;
                    fimTreinamento = nLinhas;

                    // Obtém os dados
                    teste = lerIntervalo(inicioTeste, fimTeste);
                    treinamento = lerIntervalo(inicioTreinamento, fimTreinamento);
                // Se está no final
                } else if (fimTeste == nLinhas) {
                    inicioTreinamento = ultimaLinhaCabecalho + 1;
                    fimTreinamento = inicioTeste - 1;

                    // Obtém os dados
                    teste = lerIntervalo(inicioTeste, fimTeste);
                    treinamento = lerIntervalo(inicioTreinamento, fimTreinamento);
                // Se está no meio
                } else {
                    meio = 1;
                    inicioTreinamento1 = ultimaLinhaCabecalho + 1;
                    fimTreinamento1 = inicioTeste -1;
                    inicioTreinamento2 = fimTeste + 1;
                    fimTreinamento2 = nLinhas;

                    // Obtém os dados
                    teste = lerIntervalo(inicioTeste, fimTeste);
                    treinamento = lerIntervalo(inicioTreinamento1, fimTreinamento1);
                    treinamento += lerIntervalo(inicioTreinamento2, fimTreinamento2);
                }

                /*
                System.out.print("Aleatorio: SIM\ninicioTeste = " + inicioTeste +
"\nfimTeste = " + fimTeste);
                if (meio == 0)
                    System.out.println("\ninicioTreinamento = " + inicioTreinamento +
"\nfimTreinamento = " + fimTreinamento + "\n");
                else
                    System.out.println("\ninicioTreinamento1 = " + inicioTreinamento1 +
"\nfimTreinamento1 = " + fimTreinamento1 + "\ninicioTreinamento2 = " + inicioTreinamento2
+ "\nfimTreinamento2 = " + fimTreinamento2 + "\n");

                System.out.println("Linhas Treinamento: " + linhasString(treinamento) + "
- Linhas Teste: " + linhasString(teste));
                */
            }
```

```
            // Adiciona o cabeçalho aos dados
            treinamento = cabecalho + treinamento;
            teste = cabecalho + teste;

            // Salva os arquivos
            String diretorio = arquivo.getParent() + "/";
            if (salvar(teste, diretorio + aTeste) && salvar(treinamento, diretorio +
aTreinamento))
                JOptionPane.showMessageDialog(new JFrame(), "Arquivos salvos com sucesso
no mesmo diretório do arquivo original!", "Sucesso", JOptionPane.INFORMATION_MESSAGE);
            else
                JOptionPane.showMessageDialog(new JFrame(), "Ocorreu um erro ao salvar os
arquivos!", "Erro", JOptionPane.ERROR_MESSAGE);

        } catch (Exception e) {}
    }

    private int linhasString(String str) {
        int i = 0;
        String[] linhas = str.split("\n");
        for (String linha : linhas)
            i++;

        return i;
    }

    private String ler() {
        String res = "";

        try (BufferedReader StrR = new BufferedReader(new
FileReader(arquivo.getAbsolutePath()))) {
            String Str;

            while ((Str = StrR.readLine()) != null)
                if (!Str.equals("%") && Str.length() > 1)
                    res += Str + "\n";

            return res;
        }
        catch (FileNotFoundException e) {
            return "";
        } catch (IOException e) {
            return "";
        }
    }

    private String lerIntervalo(int inicio, int fim) {
        String res = "";
        int i = 0;

        if (inicio < 1 || inicio == fim)
            return res;

        try (BufferedReader StrR = new BufferedReader(new
FileReader(arquivo.getAbsolutePath()))) {
            String Str;

            while ((Str = StrR.readLine()) != null) {
                i++;

                if (i >= inicio && i <= fim && !Str.equals("%") && Str.length() > 1)
                    res += Str + "\n";
```

```
                if (i > fim)
                    break;
            }

            return res;
        }
        catch (FileNotFoundException e) {
            return "";
        } catch (IOException e) {
            return "";
        }
    }

    private boolean salvar(String str, String nome) {
        try (BufferedWriter StrW = new BufferedWriter(new FileWriter(nome))) {
            StrW.write(str);

            return true;
        } catch (FileNotFoundException e) {
            return false;
        } catch (IOException e) {
            return false;
        }
    }

    private void salvaCabecalho() {
        String dados = ler();
        String res = "";
        int i = 0;

        String[] linhas = dados.split("\n");
        for (String linha : linhas) {
            if (linha.charAt(0) == '%' || linha.charAt(0) == '@') {
                res += linha + "\n";
                i++;
            } else
                break;
        }

        cabecalho = res;
        ultimaLinhaCabecalho = i;
    }

    private int numeroLinhas() {
        String dados = ler();
        int i = 0;

        String[] linhas = dados.split("\n");
        for (String linha : linhas)
            i++;

        return i;
    }

}
```