**Input Subsystem**

**Group 2:**

John Abueg, William Rios Crespo, Joshua Kerley, Michael Lancaster

CMSC 495 63802

**Author Note:**

Document Version IS002

Date: 2018-10-07

**Version Control**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document** | **Date** | **Action** | **Name** | **Email** |
| IS001 | 2018-09-30 | Created | William Rios Crespo | william.rioscrespo19@gmail.com |
| IS002 | 2018-10-04 | Modified | William Rios Crespo | william.rioscrespo19@gmail.com |

import java.io.File;

import java.util.ArrayList;

import java.util.Scanner;

import java.util.regex.Pattern;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

public class Input {

private File file;

private ArrayList<String> assignmentNameList = new ArrayList<>();

private ArrayList<Double> gradeList = new ArrayList<>();

private ArrayList<Double> weightList = new ArrayList<>();

public Input(String filePath) {

file = new File(filePath);

}

public Input() {

file = null;

}

public void readFile() {

try {

verifyExtension();

Scanner input = new Scanner(file);

String text = "";

while (input.hasNext()) {

text += input.next() + " ";

}

input.close();

String[] words = text.split("\\s+");

for (int i = 0; i < words.length; i++) {

if (words[i].equalsIgnoreCase("Assignment") && words[i + 1].equalsIgnoreCase("Name:")) {

if (words[i + 2].equalsIgnoreCase("Weight:")) {

assignmentNameList.add("-");

i += 1;

} else if (containSpecialCharacters(words[i + 2])) {

throw new Exception(

"Must only enter numbers and letters for assigment name. Special characters are not allowed.");

} else if (words[i + 2].equalsIgnoreCase("Grade:")) {

throw new Exception(

"Invalid Format. Please export a file that meets with the required criteria.");

} else {

assignmentNameList.add(words[i + 2]);

i += 2;

}

} else if (words[i].equalsIgnoreCase("Weight:")) {

if (words[i + 1].equalsIgnoreCase("Grade:")) {

weightList.add(0.0);

} else if (numericValue(words[i + 1])) {

weightList.add(Double.parseDouble(words[i + 1]));

i += 1;

} else if (weightList.size() > assignmentNameList.size()) {

throw new Exception(

"Invalid Format. Please export a file that meets with the required criteria.");

} else {

throw new Exception(

"Invalid Format. Please export a file that meets with the required criteria.");

}

} else if (words[i].equalsIgnoreCase("Grade:")) {

if (words[i + 1].equalsIgnoreCase("Assignment") && words[i + 2].equalsIgnoreCase("Name:")) {

gradeList.add(0.0);

} else if (numericValue(words[i + 1])) {

gradeList.add(Double.parseDouble(words[i + 1]));

i += 1;

} else if (words[i].equalsIgnoreCase("A") || words[i].equalsIgnoreCase("B")

|| words[i].equalsIgnoreCase("C") || words[i].equalsIgnoreCase("D")

|| words[i].equalsIgnoreCase("F")) {

throw new Exception("Invalid Format, must only enter numbers for a grade.");

} else if (gradeList.size() > weightList.size() || gradeList.size() > assignmentNameList.size()) {

throw new Exception(

"Invalid Format. Please export a file that meets with the required criteria.");

} else {

throw new Exception("Invalid Format, must only enter numbers for a grade.");

}

} else {

throw new Exception("Invalid Format. Please export a file that meets with the required criteria.");

}

}

} catch (Exception exception) {

displayMessageDialog(exception.getMessage());

}

}

public void userInput(String assignmentName, String weight, String grade) {

try {

if (containSpecialCharacters(assignmentName)) {

throw new Exception(

"Must only enter numbers and letters for assigment name. Special characters are not allowed.");

} else if (assignmentName.replace(" ", "").equalsIgnoreCase("")) {

assignmentNameList.add("-");

} else {

assignmentNameList.add(assignmentName);

}

if (weight.replace(" ", "").equalsIgnoreCase("")) {

weightList.add(0.0);

} else if (numericValue(weight)) {

weightList.add(Double.parseDouble(weight));

} else {

throw new Exception("Invalid Format. Please export a file that meets with the required criteria.");

}

if (grade.replace(" ", "").equalsIgnoreCase("")) {

gradeList.add(0.0);

} else if (numericValue(grade)) {

gradeList.add(Double.parseDouble(grade));

} else if (grade.equalsIgnoreCase("A") || grade.equalsIgnoreCase("B") || grade.equalsIgnoreCase("C")

|| grade.equalsIgnoreCase("D") || grade.equalsIgnoreCase("F")) {

throw new Exception("Invalid Format, must only enter numbers for a grade.");

} else {

throw new Exception("Invalid Format. Please export a file that meets with the required criteria.");

}

} catch (Exception exception) {

displayMessageDialog(exception.getMessage());

}

}

public File getFile() {

return file;

}

public ArrayList<String> getAssignmentNameList() {

return assignmentNameList;

}

public ArrayList<Double> getGradeList() {

return gradeList;

}

public ArrayList<Double> getWeightList() {

return weightList;

}

public void verifyExtension() throws Exception {

if (!((file.getPath().lastIndexOf('.') != -1) && (file.getPath()

.substring((file.getPath().lastIndexOf('.') + 1), file.getPath().length()).equalsIgnoreCase("dat")))) {

throw new Exception("Invalid extension, please enter a new dat file.");

}

}

public boolean containSpecialCharacters(String assigmentName) throws Exception {

if (Pattern.compile("[^A-Za-z0-9]").matcher(assigmentName).find()) {

return true;

}

return false;

}

public boolean numericValue(String value) {

try {

Double.parseDouble(value);

} catch (Exception exception) {

return false;

}

return true;

}

private static void displayMessageDialog(String message) {

JOptionPane.showMessageDialog(new JFrame(), message, "Input - Error", JOptionPane.INFORMATION\_MESSAGE);

}

}