package edu.mills.cs64.extraProgramming;

import javax.swing.JOptionPane;

/\*\*A single class that can be used for calculations.

\* The name and credit value of the class are determined by user input.

\*

\* @author Nell Dosker

\*

\*/

public class Course {

private String name;

private static double credit;

private String[] categories;

private boolean weighted;

private static double[] courseGrades;

/\*\*An error message for NumberFormatErrors.

\*

\*/

public static final String NUMBER\_ERROR = "Error: Please enter a valid number.";

/\*\*An array of weights for a course.

\*

\*/

public double[] weights;

/\*\*The total grade for a course.

\*

\*/

public double total;

/\*\*

\* A constructor for a course with a name and a credit value.

\* @param n name of the course

\* @param c credit value of the course

\*/

public Course(String n, double c){

name = n;

credit = c;

}

/\*\*Gets the name of the course.

\*

\* @return Name of the course.

\*/

public String getName(){

return name;

}

/\*\*Gets the credit value of the course.

\*

\* @return Credit value of the course.

\*/

public double getCredit(){

return credit;

}

/\*\*Determines whether the course is weighted or not weighted.

\*

\* @return The course grade.

\*/

public double total(){

if(categories != null){

courseGrades = new double[categories.length];

}

if(weighted){

weightsGrade();

return total;

}

else if(!weighted){

noWeightsGrade();

return total;

}

return total;

}

private static String inputDialog(String message){

String userInput = "";

new JOptionPane();

userInput = JOptionPane.showInputDialog(null, message, "Grade Calculator", JOptionPane.QUESTION\_MESSAGE);

return userInput;

}

private static String errorMessage(String message){

String userInput = "";

new JOptionPane();

userInput = JOptionPane.showInputDialog(null, message, "Error", JOptionPane.ERROR\_MESSAGE);

return userInput;

}

/\*\*Gets the categories of the course.

\*

\*/

public String getCategories(){

String name = this.getName();

String answer = inputDialog("Does " + name + " have grading categories?");

String categoryString = "";

int num = 0;

switch(answer){

case "yes":

while(true){

String number = inputDialog("How many categories?");

try{

num = Integer.parseInt(number);

break;

}

catch(NumberFormatException e){

CalculateGUI.display.append(NUMBER\_ERROR);

}

}

categories = new String[num];

CalculateGUI.display.append("\nCategories for " + name + ": \n");

CalculateGUI.display.setSelectionStart(CalculateGUI.display.getSelectionEnd());

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

String category = inputDialog("Enter category " + i+1);

categories[i] = category;

categoryString += categories[i] + "\n";

}

return categoryString;

case "no":

CalculateGUI.display.append(name + " does not have categories.");

return categoryString;

default:

answer = errorMessage("Please enter 'yes' or 'no'.");

}

return null;

}

/\*\*Gets the weights of the course.

\*

\* @param answer An answer entered by the user

\*

\*/

public String getWeights(String answer){

String weightString = "";

switch(answer){

case "yes":

weighted = true;

weights = new double[categories.length];

while(true){

try{

int index = 0;

weightString += getWeights(index);

CalculateGUI.setVisibility(true);

break;

}

catch(NumberFormatException e){

CalculateGUI.display.append(NUMBER\_ERROR);

}

}

return weightString;

case "no":

weighted = false;

CalculateGUI.display.append("\n" + this.getName() + " is not weighted.");

return weightString;

default:

answer = errorMessage("Please enter 'yes' or 'no'.");

}

return null;

}

private String getWeights(int index){

String weightString = "";

if(index != weights.length){

try{

double weight = Double.parseDouble(inputDialog("Enter the weight for " + categories[index]));

weights[index] = weight;

weightString += categories[index] + ": " + weight + "\n";

index++;

weightString += getWeights(index);

}

catch(NumberFormatException e){

CalculateGUI.display.append(NUMBER\_ERROR);

}

}

return weightString;

}

private double noWeightsGrade(){

total = 0.0;

try{

if(categories == null){

total = Double.parseDouble(inputDialog("Enter total points earned."));

Double outOf = Double.parseDouble(inputDialog("Enter total possible points."));

total = total/outOf;

if(total <= 1){

total = total\*100;

return total;

}

}

else{

total = noWeightsGrade(0);

}

double outOf = Double.parseDouble(inputDialog("Enter total points possible overall."));

total = total/outOf;

if(total <= 1){

total = total\*100;

return total;

}

}

catch (NumberFormatException e){

CalculateGUI.display.append(NUMBER\_ERROR);

}

return total;

}

private double noWeightsGrade(int index){

double grade = 0;

try{

if(index != categories.length){

grade = Double.parseDouble(inputDialog("Enter total points for " + categories[index] + "."));

courseGrades[index] = grade;

grade += noWeightsGrade(index+1);

}

}

catch (NumberFormatException e){

CalculateGUI.display.append(NUMBER\_ERROR);

}

return grade;

}

private double weightsGrade(){

total = weightsGrade(0);

if(total <= 1 && total != 0){

total = total\*100;

return total;

}

return total;

}

private double weightsGrade(int count){

double grade = 0;

if(count != weights.length){

try{

grade = Double.parseDouble(inputDialog("Enter total points for " + categories[count] + "."));

grade = grade\*weights[count];

courseGrades[count] = grade;

grade += weightsGrade(count+1);

}

catch(NumberFormatException e){

CalculateGUI.display.append(NUMBER\_ERROR);

}

}

return grade;

}

/\*\*Calculates the grade point value of the class based on its grade.

\*

\* @param grade The grade (percentage) for this class.

\* @return Grade point value for this class.

\*/

public double getGradePoints(double grade){

double gradePoint = 0.0;

if(grade <= 62){

gradePoint = 0.67;

return gradePoint;

}

if(grade <= 66){

gradePoint = 1.0;

return gradePoint;

}

if(grade <= 69){

gradePoint = 1.33;

return gradePoint;

}

if(grade <= 72){

gradePoint = 1.67;

return gradePoint;

}

if(grade <= 76){

gradePoint = 2.0;

return gradePoint;

}

if(grade <= 79){

gradePoint = 2.33;

return gradePoint;

}

if(grade <= 82){

gradePoint = 2.67;

return gradePoint;

}

if(grade <= 86){

gradePoint = 3.0;

return gradePoint;

}

if(grade <= 89){

gradePoint = 3.33;

return gradePoint;

}

if(grade <= 92){

gradePoint = 3.67;

return gradePoint;

}

if(grade <= 100){

gradePoint = 4.0;

return gradePoint;

}

return gradePoint;

}

}