Exercise 7.1

/\*

(Assign grades) Write a program that reads student scores, gets the best score,

and then assigns grades based on the following scheme:

Grade is A if score is >= best - 10;

Grade is B if score is >= best - 20;

Grade is C if score is >= best - 30;

Grade is D if score is >= best - 40;

Grade is F otherwise.

The program prompts the user to enter the total number of students, then prompts

the user to enter all of the scores, and concludes by displaying the grades. Here

is a sample run:

\*/

import java.util.Scanner;

public class Exercise\_07\_01 {

/\*\* Main method \*/

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Prompt the user to enter the total number of students

System.out.print("Enter the nubmer of students: ");

int[] scores = new int[input.nextInt()];

char[] grades = new char[scores.length];

// Prompt the user to enter all the scores

System.out.print("Enter " + scores.length + " scores: ");

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

scores[i] = input.nextInt();

}

// Get grades

getGrades(scores, grades);

// Display results

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

System.out.println("Student " + i + " score is " +

scores[i] + " and grade is " + grades[i]);

}

}

/\*\* Method max returns the is highest score \*/

public static int max(int[] array) {

int max = array[0];

for (int i = 1; i < array.length; i++) {

if (array[i] > max)

max = array[i];

}

return max;

}

/\*\* Method getGrade assigns grades based on grading scheme \*/

public static void getGrades(int[] scores, char[] grades) {

int best = max(scores);

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

if (scores[i] >= best - 10)

grades[i] = 'A';

else if (scores[i] >= best - 20)

grades[i] = 'B';

else if (scores[i] >= best - 30)

grades[i] = 'C';

else if (scores[i] >= best - 40)

grades[i] = 'D';

else

grades[i] = 'F';

}

}

}

8.29

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* (Identical arrays) The two-dimensional arrays m1 and m2 are identical if they \*

\* have the same contents. Write a method that returns true if m1 and m2 are \*

\* identical, using the following header: \*

\* \*

\* public static boolean equals(int[][] m1, int[][] m2) \*

\* \*

\* Write a test program that prompts the user to enter two 3 \* 3 arrays of \*

\* integers and displays whether the two are identical. \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.util.Scanner;

public class Exercise\_08\_29 {

/\*\* Main method \*/

public static void main(String[] args) {

// Prompt the user to enter two 3 x 3 arrays

System.out.print("Enter list1: ");

int[][] list1 = getArray();

System.out.print("Enter list2: ");

int[][] list2 = getArray();

// Displays whether the two lists are identical

System.out.println("The two arrays are" +

(equals(list1, list2) ? " " : " not ") + "identical");

}

/\*\* getArray initializes a 3 x 3 array with user input \*/

public static int[][] getArray() {

Scanner input = new Scanner(System.in);

final int ROWS = 3;

final int COLUMNS = 3;

int[][] m = new int[ROWS][COLUMNS];

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

for (int j = 0; j < m[i].length; j++) {

m[i][j] = input.nextInt();

}

}

return m;

}

/\*\* equals returns true if m1 and m2 are identical \*/

public static boolean equals(int[][] m1, int[][] m2) {

int[] list1 = sort(m1);

int[] list2 = sort(m2);

for (int i = 0; i < list1.length; i++) {

if (list1[i] != list2[i])

return false;

}

return true;

}

/\*\* matrixToArray returns an array initialized with a matrix elements \*/

public static int[] matrixToArray(int[][] m) {

int[] list = new int[m.length \* m[0].length];

int k = 0;

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

for (int j = 0; j < m[i].length; j++) {

list[k] = m[i][j];

k++;

}

}

return list;

}

/\*\* sort sorts each column in a 3 x 3 array in accending order \*/

public static int[] sort(int[][] m) {

int [] list = matrixToArray(m);

for (int i = 0; i < 3; i++) {

// Find the min in the column 1

int min = list[i];

int minIndex = i;

for (int j = i + 1; j < 3; j++) {

if (min > list[j]) {

min = list[j];

minIndex = j;

}

}

// Swap

if (minIndex != i) {

list[minIndex] = list[i];

list[i] = min;

}

}

return list;

}

}