**CS 1120 Computer Science II (with Java),** *< “*Spring 2019*”>*

Department of Computer Science, Western Michigan University

Instructor: *< Fola Olagbemi>*, Lab TA: *<Wassnaa Al Mawee>*

<*Anfaal Faisal*>

**SOFTWARE LIFE CYCLE REPORT – FOR LAB ASSIGNMENT** *<LA 1>*

# PHASE 1: SPECIFICATION (“What do we build?”)

Write two classes, **MultipleChoiceExam** and **TestMChExam** than can:

1. Take a content document with the primary line being the measure of inquiries on a test, the second line being the responses to those inquiries on the test, the third line being what number of students took the test, and ultimately, 'the fourth line to anyway numerous understudies took the test lines' being those understudies' responses to that test and read it into arrays.
2. Find the amount of incorrect answers, their percentage grade and their letter grade for each student.
3. Display the answer key and the answers by students the way observed below

Answer Key:

1) a

2) b

3) c

4) a

5) b

6) a

7) b

8) b

9) a

10) d

11) d

12) c

Student Answers:

Student 1 Student 2 Student 3

1) a 1) a 1) a

2) b 2) b 2) b

3) c 3) c 3) c

4) a 4) b (a) 4) a

5) b 5) b 5) b

6) a 6) b (a) 6) a

7) c (b) 7) b 7) b

8) b 8) d (b) 8) d (b)

9) a 9) a 9) a

10) c (d) 10) d 10) d

11) d 11) d 11) d

12) c 12) a (c) 12) c

Student 1: 83% B

Student 2: 67% D

Student 3: 92% A

Test the two classes by performing the following steps:

1. Convert the ArrayList into an array by reading the “input.txt” file using an ArrayList.
2. Make one array with just the answer key, and then one array with just the students’ answers.
3. Make a one-dimensional counter array filled with a zero for however many students there are.
4. Create a new **MultipleChoiceExam** object with the name “examGrader” and sending the amount of questions on the exam, the answer key for the exam, the number of students that took the exam, the students’ answers to the exam, and the counter array that is filled with zeros.
5. Display the answer key with the number that goes with each answer.
6. Display the students’ answer with the numbers that goes with each answer.

Display the students’ percentage and letter grade.

**PHASE 2: DESIGN**

**2.1 Modules and Their Basic Structure**

My program will have two modules.

1. Module 1: Class **MultipleChoiceExam** will contain:
2. Private fields  representing the amount of questions, a character one-dimensional array for the answer key, the number of students, a character two-dimensional array for the students’ answers, and an integer one-dimensional array counter array to count the amount of incorrect answers for each student.
3. A method **examGrader** that compares the students’ answers to the answer key and counts how many questions each student had incorrect.
4. A method **findLetterGrades** that finds the letter grade for each student.
5. A method **Output** that displays the answer key with corresponding numbers, the students’ answers with corresponding numbers and correct answers necessary, and students’ percentage on the exam and their letter grades according to that percentage.
6. Module 2: The testing module **TestMChExam** will contain:
7. **fileReader** that reads the “input.txt” file, put it into an array, and returns that array.
8. **makeStudentArray** that takes the array that was read from the file and takes the lines from 4 on, to however many students took the exam, and turns those lines into separate strings in that array. Then returns that array.
9. **makeCounterArray** that loads an array with zeros, with the size of however many students took the exam.

**2.2 Pseudocode for the Modules**

**2.2.1 Pseudocode for** MultipleChoiceExam

1a) **MultipleChoiceExam** Pseudocode Refinement #1:

// Fields (private): int questions , Character array answerKey, int numOfStu , character Array stuAnswers

// int array incorrect

// Constructor: Initialize fields questions, answerKey, numOfStu, stuAnswers,

// and incorrect (with values from parameters).

// Methods: --------------------------------------------------------------------------------------------------------------

// Method findLetterGrades: Find the letter grade based on the percentage.

// Method examGrader: Find whether the students’ answers are correct or incorrect and count // the incorrect answers for each student

// Method Output: Display the answer key, the students’ answers, students letter grades

// and percentages

1b) **MultipleChoiceExam** Pseudocode Refinement #2:

// Fields (private): int questions , Character array answerKey, numOfStu , character Array stuAnswers

// int array incorrect

// Constructor: Initialize fields questions, answerKey, numOfStudents, studentAnswers,

// and incorrect (with values from parameters).

// setQuestions

// setAnswerKey

// setNumOfStudents

// setStudentAnswers

// setIncorrect

// Methods: --------------------------------------------------------------------------------------------------------------

// Method findLetterGrades: Find the letter grade based on the percentage.

// A percentage score is sent into the method using a for loop in Output method // Initialize a character ‘c’ that is used in the return from this method

// For the percentage score that is sent into the method, it is compared to a

// grading scale to find the exact letter grade of each percentage score sent in

// return ‘c’ as a character to the for loop in Output method

// Method examGrader: Find whether the students’ answers are correct or incorrect and count // the incorrect answers for each student

// Two Integers are sent into this method from Output method ‘i’ and ‘j’ which are // the row and column number to locate the exact student answer with the

// corresponding answer from the answer key (The ‘i’ is from the outer for loop in // Output method and the ‘j’ is from the inner for loop from Output

// method)

// Initialize a String ‘returned’ that is used in the return from this method

// The two-dimensional character array ‘studentAnswers’ takes ‘i’ and ‘j’ to pinpoint the // location of which student answer it is looking for and compares that answer to // the answer key and sets ‘returned’ to a formatted string (The formatted string // is different for if it is true or false)

// If the student’s answer is false then the one-dimensional integer array ‘incorrect’ is

// given a plus one counter to the ‘j’ (the student) in which had it wrong

//return ‘returned’ as a String to the inner for loop in Output method

// Method Output: Display the answer key, the students’ answers, students letter grades

// and percentages

// Display the header for the answer key

// Display the answer key with their corresponding numbers

// Display the header for the students’ answers

// Display the which student number are which (i.e. Student 2:)

// Display the students’ answers with their corresponding numbers and the incorrect

// answers if needed using the examGrader method

// Display the student’s number and calculate their percentage score and send that

// score to the findLetterGrade method, to then display the student’s percentage // score and letter grade

1c) **MultipleChoiceExam** Pseudocode Refinement #3:

// Fields: questions (int), answerKey (Character array), numOfStudents (int), studentAnswers

// (Character array), incorrect (int array)

// Constructor: Initialize fields questions, answerKey, numOfStudents, studentAnswers,

// and incorrect (with values from parameters).

// setQuestions

// setAnswerKey

// setNumOfStudents

// setStudentAnswers

// setIncorrect

// Methods: --------------------------------------------------------------------------------------------------------------

// Method findLetterGrades: Find the letter grade based on the percentage.

// A percentage score is sent into the method using a for loop in Output method // Initialize a character ‘c’ that is used in the return from this method

// For the percentage score that is sent into the method, it is compared to a

// grading scale to find the exact letter grade of each percentage score sent in

// Compare each percentage score that is sent in:

// If its greater than or equal to 90 then it is an ‘A’, if it’s between 90 and // then it’s a ‘B’, if it’s between 80 and 70 then it’s a ‘C’, if its

// between 70 and 60 it’s a ‘D’ and any other score is an ‘E’.

// return ‘c’ as a character to the for loop in Output method

// Method examGrader: Find whether the students’ answers are correct or incorrect and count // the incorrect answers for each student

// Two Integers are sent into this method from Output method ‘i’ and ‘j’ which are // the row and column number to locate the exact student answer with the

// corresponding answer from the answer key (The ‘i’ is from the outer for loop in // Output method and the ‘j’ is from the inner for loop from Output

// method)

// Initialize a String ‘returned’ that is used in the return from this method

// The two-dimensional character array ‘studentAnswers’ takes ‘i’ and ‘j’ to pinpoint the // location of which student answer it is looking for and compares that answer to // the answer key and sets ‘returned’ to a formatted string (The formatted string // is different for if it is true or false)

// If the student’s answer is false then the one-dimensional integer array ‘incorrect’ is

// given a plus one counter to the ‘j’ (the student) in which had it wrong

//return ‘returned’ as a String to the inner for loop in Output method

// Method Output: Display the answer key, the students’ answers, students letter grades

// and percentages

// Display the header for the answer key

// Display the answer key with their corresponding numbers

// Display the header for the students’ answers

// Display the which student number are which (i.e. Student 2:)

// Display the students’ answers with their corresponding numbers and the incorrect

// answers if needed using the examGrader method

// Display the student’s number and calculate their percentage score and send that

// score to the findLetterGrade method, to then display the student’s percentage // score and letter grade

**2.2.2. Pseudocode for** TestMChExam

2a) Pseudocode Refinement #1:

// Methods: --------------------------------------------------------------------------------------------------------------

// Method Main: Calls both static and non-static methods from the program

// Method fileReader: Read the “input.txt” and returns an array with each line being however

// many lines there are in the “input.txt” file

// Method makeStudentArray: Creates a one-dimensional String array with the students’

// answers

// Method makeCounterArray: Creates a one-dimensional integer array, with a size of the

// amount of students that took the exam, as well as zeros loaded into each spot

**PHASE 3: RISK ANALYSIS (“What can go wrong, and how bad can it be?”)**

I do not see any risks involved in the program.

**PHASE 4: VERIFICATION (“Are the algorithms correct?”)**

Only a sequential execution can be seen

**PHASE 5: CODING**

**5a) Code Refinement #1 (class structure with pseudocode only; pseudocode is used as comments)**

**File MultipleChoiceExam.java:**

**package edu.wmich.CS1120.LA1.AFaisal;**

**import java.io.\*;**

**import java.util.Arrays;**

// The MultipleChoiceExam class.

**Public class MultipleChoiceExam {**

// Fields (private): int questions , Character array answerKey, int numOfStu , character Array stuAnswers

// int array incorrect

// Constructor: Initialize fields questions, answerKey, numOfStu, stuAnswers,

// and incorrect (with values from parameters).

// setQuestions

// setAnswerKey

// setNumOfStudents

// setStudentAnswers

// setIncorrect

// Methods: --------------------------------------------------------------------------------------------------------------

// Method findLetterGrades: Find the letter grade based on the percentage.

// A percentage score is sent into the method using a for loop in Output method // Initialize a character ‘c’ that is used in the return from this method

// For the percentage score that is sent into the method, it is compared to a

// grading scale to find the exact letter grade of each percentage score sent in

// Compare each percentage score that is sent in:

// If its greater than or equal to 90 then it is an ‘A’, if it’s between 90 and // then it’s a ‘B’, if it’s between 80 and 70 then it’s a ‘C’, if its

// between 70 and 60 it’s a ‘D’ and any other score is an ‘E’.

// return ‘c’ as a character to the for loop in Output method

// Method examGrader: Find whether the students’ answers are correct or incorrect and count // the incorrect answers for each student

// Two Integers are sent into this method from Output method ‘i’ and ‘j’ which are // the row and column number to locate the exact student answer with the

// corresponding answer from the answer key (The ‘i’ is from the outer for loop in // Output method and the ‘j’ is from the inner for loop from Output

// method)

// Initialize a String ‘returned’ that is used in the return from this method

// The two-dimensional character array ‘studentAnswers’ takes ‘i’ and ‘j’ to pinpoint the // location of which student answer it is looking for and compares that answer to // the answer key and sets ‘returned’ to a formatted string (The formatted string // is different for if it is true or false)

// If the student’s answer is false then the one-dimensional integer array ‘incorrect’ is

// given a plus one counter to the ‘j’ (the student) in which had it wrong

//return ‘returned’ as a String to the inner for loop in Output method

// Method Output: Display the answer key, the students’ answers, students letter grades

// and percentages

// Display the header for the answer key

// Display the answer key with their corresponding numbers

// Display the header for the students’ answers

// Display the which student number are which (i.e. Student 2:)

// Display the students’ answers with their corresponding numbers and the incorrect

// answers if needed using the examGrader method

// Display the student’s number and calculate their percentage score and send that

// score to the findLetterGrade method, to then display the student’s percentage // score and letter grade

**}** // end class MultipleChoiceExam

**File TestMChExam.java:**

**package edu.wmich.CS1120.LA1.AFaisal;**

**import java.io.\*;**

**import java.util.\*;**

**import java.util.Scanner;**

// The TestMChExam class for testing the MultipleChoiceExam class.

**public class TestMChExam {**

// Methods: --------------------------------------------------------------------------------------------------------------

// Method Main: Calls both static and non-static methods from the program

// Initialize a String array equal to the contents of the method fileReader

// Initialize a String array equal to the contents of the method makeStudentArray

// Initialize an integer array equal to the contents of the method makeCounterArray

// Create a new MultipleChoiceExam object to load the amount of questions, the

// answer key, the number of students that took the exam, the students’ answers, // and the incorrect counter array

// Call Output method from the class MultipleChoiceExam

// Method fileReader: Read the “input.txt” and returns an array with each line being however

// many lines there are in the “input.txt” file

// Read file into an ArrayList

// Copy the ArrayList contents into an array

// return the array

// Method makeStudentArray: Creates a one-dimensional String array

// answers

// Take in the array that has the file contents (using parameters)

// Create an array to copy the lines: three to however many students took the exam

// return the array

// Method makeCounterArray: Creates a one-dimensional integer array, with a size of the

// amount of students that took the exam, as well as zeros loaded into each spot

// Create an integer array the size of however many students took the exam

// Fill that array with a zero for every space in the array

// return the array

**}**

**5b) Code Refinement #2 (still incomplete program: class and constructor/method structure with pseudocode only; pseudocode is used as comments)**

**File MultipleChoiceExam.java:**

**package edu.wmich.CS1120.LA1.AFaisal;**

// The MultipleChoiceExam class.

**Public class MultipleChoiceExam {**

// Fields (private): int questions , Character array answerKey, int numOfStu , character Array stuAnswers

// int array incorrect

// Constructor: Initialize fields questions, answerKey, numOfStudents, studentAnswers,

// and incorrect (with values from parameters).

**public MultipleChoiceExam (/\*questions, answerKey, numOfStudents, studentAnswers, incorrect\*/) {**

// setQuestions

// setAnswerKey

// setNumOfStudents

// setStudentAnswers

// setIncorrect

**}** // end constructor

// Methods: --------------------------------------------------------------------------------------------------------------

// Method findLetterGrades: Find the letter grade based on the percentage.

**public char findLetterGrades (double score) {**

// A percentage score is sent into the method using a for loop in Output method // Initialize a character ‘c’ that is used in the return from this method

// For the percentage score that is sent into the method, it is compared to a

// grading scale to find the exact letter grade of each percentage score sent in

// Compare each percentage score that is sent in:

// If its greater than or equal to 90 then it is an ‘A’, if it’s between 90 and // then it’s a ‘B’, if it’s between 80 and 70 then it’s a ‘C’, if its

// between 70 and 60 it’s a ‘D’ and any other score is an ‘E’.

// return ‘c’ as a character to the for loop in Output method

**}** // end method findLetterGrades

// Method examGrader: Find whether the students’ answers are correct or incorrect and count // the incorrect answers for each student

**public String examGrader (int i, int j) {**

// Two Integers are sent into this method from Output method ‘i’ and ‘j’ which are // the row and column number to locate the exact student answer with the

// corresponding answer from the answer key (The ‘i’ is from the outer for loop in // Output method and the ‘j’ is from the inner for loop from Output

// method)

// Initialize a String ‘returned’ that is used in the return from this method

// The two-dimensional character array ‘studentAnswers’ takes ‘i’ and ‘j’ to pinpoint the // location of which student answer it is looking for and compares that answer to // the answer key and sets ‘returned’ to a formatted string (The formatted string // is different for if it is true or false)

// If the student’s answer is false then the one-dimensional integer array ‘incorrect’ is

// given a plus one counter to the ‘j’ (the student) in which had it wrong

//return ‘returned’ as a String to the inner for loop in Output method

**}** // end method examGrader

// Method Output: Display the answer key, the students’ answers, students letter grades

// and percentages

**public void Output () {**

// Display the header for the answer key

// Display the answer key with their corresponding numbers

// Display the header for the students’ answers

// Display the which student number are which (i.e. Student 2:)

// Display the students’ answers with their corresponding numbers and the incorrect

// answers if needed using the examGrader method

// Display the student’s number and calculate their percentage score and send that

// score to the findLetterGrade method, to then display the student’s percentage // score and letter grade

**}** // end method Output

**}**

**File TestMChExam.java:**

**Package edu.wmich.CS1120.LA1.NMacklinCamel;**

// The TestMChExam class for testing the MultipleChoiceExam class.

**public class TestMChExam {**

// Methods: --------------------------------------------------------------------------------------------------------------

// Method Main: Calls both static and non-static methods from the program

**public static void main(String[] args) {**

// Initialize a String array equal to the contents of the method fileReader

// Initialize a String array equal to the contents of the method makeStudentArray

// Initialize an integer array equal to the contents of the method makeCounterArray

// Create a new MultipleChoiceExam object to load the amount of questions, the

// answer key, the number of students that took the exam, the students’ answers, // and the incorrect counter array

// Call Output method from the class MultipleChoiceExam

**}** // end Main

// Method fileReader: Read the “input.txt” and returns an array with each line being however

// many lines there are in the “input.txt” file

**public static String[] fileReader ()** **{**

// Read file into an ArrayList

// Copy the ArrayList contents into an array

// return the array

**}** // end method fileReader

// Method makeStudentArray: Creates a one-dimensional String array with the students’

// answers

**public static String makeStudentArray (String[] arrayReader) {**

// Take in the array that has the file contents (using parameters)

// Create an array to copy the lines: three to however many students took the exam

// return the array

**}** // end method makeStudentArray

// Method makeCounterArray: Creates a one-dimensional integer array, with a size of the

// amount of students that took the exam, as well as zeros loaded into each spot

**public static makeCounterArray (String[] arrayReader) {**

// Create an integer array the size of however many students took the exam

// Fill that array with a zero for every space in the array

// return the array

**}** // end method makeCounterArray

**}** // end class TestMChExam

**5c) Code Refinement #3 (complete program--with complete fields/properties, code for constructor/methods)**

**File MultipleChoiceExam.java:**

**package edu.wmich.CS1120.LA1.AFaisal;**

**import java.io.\*;**

**import java.util.Arrays;**

**public class MultipleChoiceExam {**

**private int questions;**

**private int numOfStu;**

**private int[] incorrect;**

**private Character[][] stuAnswers;**

**private Character [] answerKey;**

**public MultipleChoiceExam(String numQuestions, String Key ,String numStudents,String[] studentsAnswers,int[] incorrectArray) {**

**setQuestions(numQuestions);**

**setAnswerKey(Key);**

**setNumOfStudents(numStudents);**

**setStudentAnswers(studentsAnswers);**

**setIncorrect(incorrectArray);**

**}**

**public int getQuestions() {**

**return questions;**

**}**

**public void setQuestions(String questions) {**

**this.questions= Integer.*parseInt*(questions);**

**}**

**public Character[] getAnswerKey() {**

**return answerKey;**

**}**

**public void setAnswerKey(String Key) {**

**Character [] answers= new Character[questions];**

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

**answers[i]= new Character(Key.charAt(i));**

**}**

**this.answerKey= answers;**

**}**

**public int getNumOfStudents(){**

**return numOfStu;**

**}**

**public void setNumOfStudents(String numOfStu) {**

**this.numOfStu= Integer.*parseInt*(numOfStu);**

**}**

**public Character[][] getStudentAnswers() {**

**return stuAnswers;**

**}**

**public void setStudentAnswers(String[] answers) {**

**Character[][] students= new Character [questions][numOfStu];**

**for (int i=0;i< questions; i++) {**

**for (int j=0; j<numOfStu; j++) {**

**students [i][j]= new Character(answers[j].charAt(i));**

**}**

**}**

**this.stuAnswers= students;**

**}**

**public int[] getIncorrect() {**

**return incorrect;**

**}**

**public void setIncorrect(int[] incorrect) {**

**this.incorrect= incorrect;**

**}**

**public char findLetterGrades(double score) {**

**char c;**

**if(score>=90) {**

**c='A';**

**} else if (score< 90 && score> 80) {**

**c='B';**

**}else if (score< 80 && score> 70) {**

**c= 'C';**

**}else if (score< 70 && score> 60) {**

**c= 'D';**

**}else {**

**c= 'E';**

**}**

**return c;**

**}**

**public String examGrader (int i, int j) {**

**String returned;**

**if(stuAnswers[i][j].equals(answerKey[i])) {**

**returned= String.*format*("%2d)%2s\t\t",i+1, stuAnswers[i][j]);**

**}else {**

**returned = String.*format*("%2d)%2s (%s)\t ", i+1,stuAnswers[i][j], answerKey[i]);**

**incorrect[j]++;**

**}**

**return returned;**

**}**

**public void Output() {**

**System.*out*.printf("%s\n","Answer Key:");**

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

**System.*out*.printf("%2d)%2s\n", i+1,answerKey[i]);**

**}**

**System.*out*.println("\nStudent Answers:");**

**for(int i=0; i< numOfStu; i++){**

**System.*out*.printf("Student %d\t ",i+1);**

**}**

**System.*out*.println();**

**String PrintNice;**

**for (int i=0;i< questions; i++){**

**for (int j=0; j<numOfStu; j++) {**

**PrintNice= examGrader(i,j);**

**System.*out*.printf(PrintNice);**

**}**

**System.*out*.println();**

**}**

**double length= answerKey.length;**

**double score;**

**char letterGrade;**

**for(int i=0;i< numOfStu; i++) {**

**score=(((questions-incorrect[i])\*100)/length);**

**letterGrade= findLetterGrades(score);**

**System.*out*.printf("Student %d: %.0f%% %s\n", i+1,score,letterGrade);**

**}**

**}**

**}**

**File TestMChExam.java:**

**package edu.wmich.CS1120.LA1.AFaisal;**

**import java.io.\*;**

**import java.util.\*;**

**import java.util.Scanner;**

**public class TestMchExam {**

**public static void main(String[] args) throws IOException{**

**String[] arrayReader= *fileReader*();**

**String[] studentArray= *makeStudentArray*(arrayReader);**

**int[] counterArray= *makeCounterArray*(arrayReader);**

**MultipleChoiceExam examGrader= new MultipleChoiceExam(arrayReader[0],**

**arrayReader[1],arrayReader[2],studentArray, counterArray);**

**examGrader.Output();**

**}**

**public static String[] fileReader() throws IOException{**

**File f= new File("input.txt");**

**Scanner inFile= new Scanner(f);**

**ArrayList<String> listReader= new ArrayList<String>();**

**while(inFile.hasNext()) {**

**String record= inFile.nextLine();**

**listReader.add(record);**

**}**

**inFile.close();**

**String arrayReader[]= new String[listReader.size()];**

**for(int i=0;i<listReader.size();i++) {**

**arrayReader[i]=listReader.get(i);**

**}**

**return arrayReader;**

**}**

**public static String[] makeStudentArray(String[] arrayReader) {**

**String[] studentArray= new String[arrayReader.length-3];**

**int j= 0;**

**for(int i = 3 ; i< arrayReader.length; i++) {**

**studentArray[j]= new String (arrayReader[i]);**

**j++;**

**}**

**return studentArray;**

**}**

**public static int [] makeCounterArray(String[] arrayReader) {**

**int size= Integer.*parseInt*(arrayReader[2]);**

**int [] counterArray= new int [size];**

**for (int i=0; i<size; i++) {**

**counterArray[i] = 0;**

**}**

**return counterArray;**

**}**

**}**

**PHASE 6: TESTING (“Did we build it correctly?”)**

The single test produces the following output of the program:

Answer Key:

1) a

2) b

3) c

4) a

5) b

6) a

7) b

8) b

9) a

10) d

11) d

12) c

Student Answers:

Student 1 Student 2 Student 3

1) a 1) a 1) a

2) b 2) b 2) b

3) c 3) c 3) c

4) a 4) b (a) 4) a

5) b 5) b 5) b

6) a 6) b (a) 6) a

7) c (b) 7) b 7) b

8) b 8) d (b) 8) d (b)

9) a 9) a 9) a

10) c (d) 10) d 10) d

11) d 11) d 11) d

12) c 12) a (c) 12) c

Student 1: 83% B

Student 2: 67% D

Student 3: 92% A

output satisfies program requirements.

**PHASE 7: REFINING THE PROGRAM (“Add bells and whistles to the program”)**

This program is simple and requires no refinement

**PHASE 8: PRODUCTION**

**PHASE 9: MAINTENANCE**

No maintenance required