

Java programs with arrays and multiple classes

The following program corrects a multiple-choice quiz.. It reads two files of answers: one of answers considered correct (*corr-ans.txt*), one of answers given by test takers (*stu-ans.txt*). It outputs a file with results for each test taker and for each question (*results.txt*). The input and output files are below.

corr-ans.txt:

babbbeededbbab
63.152 s08 miniquiz t3-4

stu-ans.txt:

A aaabacedebcbab
B babbbeecedeadc
C aabcbbbededbaab
D aabcbbbededbaab
E aabcbbbededbaab
F aabbbeededcaab
G aadeabedcaeeaa
H aabbbeededeeab
I aabbbeededcaab
J aabcaddabcbcd
K aabbbeededcaab
L babbbeededebab
M aabbbeededcaab
N aabbabeeedbdab
O babebcqdebc dab
P babbbeededdbbe
Q BABBAECEBCDBA
R aabbbedadbdab
S aabbbeededcaab
T aabbbededeeab
U baabeaqdqabaqb
V babbbeddebeeab
W abbeebadbbcdea

results.txt:

Student:

A 7 correct AAABACEDEBCBAA
B 9 correct BABBBEECEDEADC
C 10 correct AABCBBEDEDBAAB
D 10 correct AABCBBEDEDBAAB
E 10 correct AABCBBEDEDBAAB
F 11 correct AABBBEEDEDCAAB
G 4 correct AADEABEDCAEEAA
H 11 correct AABBBEEDEDEEAB
I 11 correct AABBBEEDEDCAAB
J 5 correct AABCADDABCBCD
K 11 correct AABBBEEDEDCAAB
L 13 correct BABBBEEDEDEBAB
M 11 correct AABBBEEDEDCAAB
N 9 correct AABBBEEDEDDBAB
O 4 correct BABEBCQDEBCDAB
P 11 correct BABBBEEDEDDBBE
Q 6 correct BABBAECEBCDBA
R 9 correct AABBBABEDADBDAB
S 11 correct AABBBEEDEDCAAB
T 10 correct AABBBEDEDEEAB
U 4 correct BAABEAQDQABAQB
V 10 correct BABBBEEDDEEAB
W 2 correct ABEEBADBBCCDEA

Question:

1. B 7 correct (16 7 0 0 0)
2. A 22 correct (22 1 0 0 0)
3. B 20 correct (2 20 0 1 0)
4. B 17 correct (0 17 3 0 3)
5. B 15 correct (5 15 1 0 2)
6. E 10 correct (3 7 2 1 10)
7. E 18 correct (1 0 0 4 18)
8. D 18 correct (1 0 2 18 2)
9. E 17 correct (2 3 1 0 17)
10. D 15 correct (2 5 1 15 0)
11. B 6 correct (0 6 9 2 6)
12. B 4 correct (10 4 0 4 4)
13. A 16 correct (16 3 1 1 1)
14. B 14 correct (4 14 1 1 1)

```
/* cxquiz.java:  
  Reads a file of correct answers to multiple-choice Question  
  ("x-corr-ans.txt") and a file of test-taker answers ("x-stu-ans.txt"),  
  compares letters in ('a'..'e') in these files, generates scores and  
  summary in results file.  D. Keil 4/08 */  
import java.util.Scanner;  
import java.io.FileReader;  
import java.io.PrintWriter;  
import java.io.FileNotFoundException;  
  
class Question  
{  
    public static final int NUM_ANS = 5;  
    private char corr_ans;// in 'a' .. 'e'  
    private int num_corr; // number of correct answers  
    private int[] chosen; // # Students who chose an answer  
    Question()  
    {  
        corr_ans = '\0';  
        num_corr = 0;  
        chosen = new int[NUM_ANS];  
        for (int i=0; i < NUM_ANS; i++)  
            chosen[i] = 0;  
    }  
    public char get_corr_ans() { return corr_ans; }  
    public void set_corr_ans(char x) { corr_ans = x; }  
    public void incr_num_corr() { num_corr++; }  
    public void incr_chosen(char c) { chosen[c-'A']++; }  
    public int get_num_corr() { return num_corr; }  
    public int get_chosen(int n) { return chosen[n]; }  
};
```

```

class Student
{
    private String name;
    private int num_corr; // # correct answers
    Student()
    {
        name = new String("");
        num_corr = 0;
    }
    public void set_name(String nm) { name = new String(nm); }
    public void incr_num_corr() { num_corr++; }
    public String get_name() { return name; }
    public int get_num_corr() { return num_corr; }
    public String retrieve(Scanner stu_in, PrintWriter results_out, Question[] ques,
        int num_ques)
    {
        String buf;
        buf = stu_in.next(); // Name
        set_name(buf);
        buf = stu_in.next(); // Answers
        buf = buf.toUpperCase();
        results_out.printf("%-20s ", get_name());
        int ans_num = 0;
        char ans = '\0';
        int j=0; // question number
        while(j < buf.length() && ans_num < num_ques)
        {
            ans = buf.charAt(j);
            if (ans >= 'A' && ans <= 'E')
            {
                ques[ans_num].incr_chosen(ans);
                if (ans == ques[ans_num].get_corr_ans())
                {
                    incr_num_corr();
                    ques[ans_num].incr_num_corr();
                }
                ans_num++;
            }
            j++;
        }
        if (ans_num != num_ques)
            System.out.println("wrong # Question: " + ans_num);
        return buf;
    }
};

```

```

public class cxquiz
// Application
{
    public static final int MAX_QUES = 100;
    public static final int MAX_STU = 100;
    public static int num_ques;

    private static Question[] read_corr_answers(String quiz_name)
        throws FileNotFoundException
    // Open and read correct-answers file:
    {
        Question[] ques = new Question[MAX_QUES];
        for (int i=0; i < MAX_QUES; i++)
            ques[i] = new Question();
        System.out.println("Reading "+quiz_name+"-corr-ans.txt");
        String corr_ans_name = quiz_name+"-corr-ans.txt";
        FileReader reader = new FileReader(corr_ans_name);
        Scanner corr_in = new Scanner(reader);
        String buf;
        buf = corr_in.nextLine();
        buf = buf.toUpperCase();
        num_ques = buf.length();
        System.out.println(num_ques + " Questions");
        for (int i=0; i < num_ques; i++)
            ques[i].set_corr_ans(buf.charAt(i));
        corr_in.close();
        return ques;
    }
}

```

```

public static void main(String[] args) throws FileNotFoundException
{
    Question[] ques;
    Student[] stu = new Student[MAX_STU];
    for (int i=0; i < MAX_STU; i++)
        stu[i] = new Student();

    System.out.println("Correct answers should be in 'x-corr-ans.txt'");
    System.out.println("Student answers should be in 'x-stu-ans.txt'\n");

    // Get name of quiz:
    System.out.print("Enter quiz name: ");
    Scanner cin = new Scanner(System.in);
    String quiz_name = cin.next();

    // Read correct-answers file:
    ques = read_corr_answers(quiz_name);

    // Open student-answers file:
    String stu_ans_name = quiz_name+"-stu-ans.txt";
    FileReader stu_reader = new FileReader(stu_ans_name);
    Scanner stu_in = new Scanner(stu_reader);

    // Open output file:
    PrintWriter results_out = new PrintWriter(quiz_name+"-results.txt");
    results_out.println("Student:");

    // Read student-answers file:
    int num_exams = 0;
    while (stu_in.hasNext())
    {
        String buf = stu[num_exams].retrieve(stu_in, results_out, ques, num_ques);
        results_out.printf("%2d correct %s\n",
            stu[num_exams].get_num_corr(),buf);
        num_exams++;
    }
    stu_in.close();
    results_out.println("");
    results_out.println("Question:");
    for (int j=0; j < num_ques; j++)
    {
        results_out.printf("%2d. %c %2d correct (", j+1, ques[j].get_corr_ans(),
            ques[j].get_num_corr());
        for (int i=0; i < Question.NUM_ANS; i++)
            results_out.print(ques[j].get_chosen(i) + " ");
        results_out.println(")");
    }
    results_out.close();
}
}

```

The program is designed around classes and methods. Two of the classes represent entities in the problem domain, students and questions. The third class is the application itself.

The program uses an array of student objects and an array of question objects.

Among the Java features it illustrates are classes, methods, arrays, loops, strings, constants, and formatted output using the *printf* method.