Purpose: Review of methods, classes, arrays

Details:

GET PROGRAMS AND QUESTION SHEET
From the class Web page, obtain copies of the three files TakeExam.java, Question.java, and Exam.java. Also obtain a copy of lab8questions.txt, which is a plain text file.

Compile and execute the program TakeExam to make sure it works. (It should just print out two questions about loops, followed by an error message saying “No such question as 2; returning null”.)

ANSWER THE QUESTIONS
The question sheet is a way to guide you through the study of these three programs.

Using Emacs, open the lab8questions.txt file for editing. Type your answers into this document and print it out and hand it in when you are finished. Be sure to put your name at the top.

You are allowed to use the programs to experiment as you answer the questions! For instance, you can insert “println” statements anywhere in the programs to print out values of variables, etc.

ADD SOME CODE
In XInteract, there are four new problems that ask you to add more Java code to some of the classes above. Work these exercises under the XInteract system.

ADD AN ANSWER SUBMISSION FACILITY
Modify the “Exam”, “Question”, and “TakeExam” classes as follows (some of these things you have already done using XInteract):

Add a new instance variable named answer of type String to the Question class. Modify the constructor so that it takes a second String parameter and saves it in the new instance variable. Modify the TakeExam program so that it includes a solution string along with a question when it uses the Question constructor.

HANDED OUT ON 29 MARCH 2006

Handout # 13.5
[CORRECTION ADDED ON 29 MARCH:] Add a new method to the Question.java class named getA. It should return the string stored in variable answer.

Add three more questions to the loops variable, along with their answers.

[This would be a good place to test what you have so far.]

Add accessor methods named getMaxQ and getNumQSoFar for the maxQ and numQSoFar variables to Exam.java (see page 257 if you’ve forgotten what an “accessor method” is).

[This would be a good place to test what you have so far — add some statements to the TakeExam program that print out the values of loops.getMaxQ() and loops.getNumQSoFar.]

Add a new instance variable to the Exam.java program named ans. It should be an array of type String. The array should be initialized to size maxQ inside the constructor.

Add a new method to the Exam.java program named addA that takes two parameters: an int variable i and a String named answer. The method should take the answer value and store it in the i-th location of the ans array. However, if the value of i is less than 0 or greater than or equal to maxQ, then the method should print out an appropriate error message and do nothing.

[CORRECTION ADDED ON 29 MARCH:] Add a new method to the Exam.java class named getA that takes one parameter, an integer i. It should return the string stored in position i of the ans array.

[This would be a good place to test what you have so far — add some statements to the TakeExam program that store strings in the ans array, e.g., “loops.addA(3,"this is an answer");”]

Finally, add code to the TakeExam program that asks the user to enter an answer to each question printed by the loop at the bottom of the program. The user’s answers should be stored in the ans array of the loops variable. For simplicity you can limit the answers to a single line.

After the user has answered all five questions, print out the user’s answers together with the correct answers saved in each Question. I will give an example in class.

HAND IN

Hand in your modified copies of all three files, with comments in EACH ONE containing your name and javadoc-style comments for all of the code you added to the programs. Run the program and “take the exam” that you’ve made up (it can be simple).

Checklist:

☐ Answers to questions (by Friday)

☐ XInteract solutions (by Friday)

☐ All requirements met as described above

☐ Javadoc-style comments for all new code added

☐ correctly indented code; no wrapped lines
□ Hard copy (printed using \texttt{a2ps}) of programs (by Tuesday)

□ Hard copy (printed using \texttt{a2ps}) of a sample run (by Tuesday)