655 Performance Criteria-based Questions for POCAT

Changes in POCAT: The plan is to change POCAT questions to be based on the learning outcomes/ performance criteria of various courses such as 655. The idea is to create a bank of such questions and use a selection of 20 or so questions for each test. Each test will include two or three questions from each course (covered in the POCAT). We should also make sure that over the course of three or four POCATs, at least one question related to each p.c. for each course is included. We can ensure this by rotating regularly through the questions corresponding to each course from one POCAT to the next. This will allow us to evaluate the extent to which each p.c. of each course is met.

First, though, we will have to decide what it means for a p.c. to be met. In other words, what percentage of students should be expected to answer question(s) related to a given p.c. in order for us to decide that the p.c. has been met? One simple answer would be to say that if the particular p.c. is at the "mastery"-level, say, 70% of the students must answer the corresponding question(s) correctly; if it is at the "familiarity"-level, say, 50% of the students must answer the question(s) correctly; and if it is at the "exposure"-level, say, 25% of the students must answer the question(s) correctly; else the p.c. would be considered not to have been met. Whatever numbers we agree on, having such a uniform set of numbers will ensure that the mastery/familiarity/exposure terminology is used in a uniform way in the syllabi of different courses. Another alternative would be the level of difficulty of the questions corresponding to the different levels of performance.

Processing results: Currently, the results processor is given a file that contains, for each question, information about which course it concerns, which particular program outcome it concerns, what the correct answer is, etc. With the proposed change, maybe it should also be given a file that contains the p.c.'s of each course, which p.o. it concerns, and, for each question, which course and which p.c. the question concerns. This way, the process of evaluating the extent to which each l.o./p.c. is achieved can be automated. It may also be necessary to do an annual summary (since each test will not contain questions for each l.o. of each course).

It should also be possible to have this same processor take care of on-line grading etc. In addition, we could consider accessing information about students' grades (or may be don't bother with the grades?) in the relevant courses and possibly who they took the course with etc. while protecting student anonymity so that no one can determine how an individual student performed in the POCAT. One way to ensure student anonymity would be to do summary evaluation of the POCAT results, i.e., evaluation of how well the p.c.'s of each course were achieved, just once a year; and, in this evaluation, we may factor in the grades in those courses of the students who took those POCATs. Or, possibly, we can do this on a rotating basis, evaluating two or three courses each quarter. That way, the focus will clearly be on the courses and and how well they met their stated p.c.'s rather than on individual students.

Proposed questions for CSE 655: The following is a set of such questions for CSE 655. For each question, the related learning outcome/performance criterion from the published syllabus of the course is stated followed by the question. For each question, the correct answer is flagged with "**". (This will, of course, be removed during actual tests!)

Note: For each question, circle the *one* answer you think is most appropriate. Do not circle more than one.

1. *Learning outcome: Master using syntax-related concepts including context-free grammars, parse trees, and recursive-descent parsing, printing, execution and code generation.*

   One common problem in programs is that of *uninitialized variables*, i.e., using a variable without having initialized it. This is commonly a run-time error but Java flags this error at compile time. How does it do this?

   (a) Java uses a special technology that converts run-time errors into compile-time errors;
   (b) Java uses a "conservative" approach, sometimes flagging situations which are not actually erroneous; **
   (c) Java does automatic initialization of all variables so the problem of uninitialized variables cannot arise in Java programs;
   (d) Java is an interpreted language, so this question is meaningless;
   (e) I have no idea.

2. *Learning outcome: Master analyzing programming language design issues related to data types, expressions and control structures.*

   Suppose there are two languages L1, L2 that are otherwise similar but L1 provides facilities for *data abstraction*, L2 does not. Assume that the languages L1 and L2 are implemented in similar ways. You would expect:

   (a) Programs in L1 to run faster than equivalent programs in L2;
   (b) Programs in L1 to run at about the same speed as equivalent programs in L2; **
   (c) None of the above; this question is meaningless;
   (d) I have no idea.

3. *Learning outcome: Master analyzing data abstraction-related issues.*

   Consider a standard OO language such as C++ or Java and a set of classes written in such a language and some client code that uses these classes. In this context, *data abstraction* is important because:

   (a) It enables us to implement the classes to be more efficient than if we did not make use of data abstraction;
   (b) It enables us to implement the client code to be more efficient than if we did not make use of data abstraction;
   (c) It enables us to implement the client code without regard to the internal details of the classes; **
   (d) It enables us to implement the code of the classes without regard to the internal details of the client code;
   (e) I have no idea.

4. *Learning outcome: Master implementation techniques for imperative languages. [This is not a good question for* this *outcome; the next one is probably better.]*

   Consider the language C++. The C++ programmer is responsible for managing:

   (a) The stack memory;
   (b) The heap memory; **
   (c) Both the stack and heap;
   (d) Neither the stack nor the heap;
   (e) I have no idea.

5. *Learning outcome: Master implementation techniques for imperative languages.*

   Consider a typical imperative language such as *C*. Since a program in such a language may include recursive functions, how does the compiler allocate the space needed by all the variables of the various functions, given that we don't know, at compile time, how deep the recursive calls will be?

   (a) The compiler doesn't directly allocate the needed space; instead, it produces code that, at run-time, will allocate the needed space on the stack; **
   (b) The compiler uses the heap to decide how much space needs to be allocated;
   (c) It is the responsibility of the C programmer to use pointers appropriately to take care of allocating the needed space;
   (d) I have no idea.

6. *Learning outcome: Be familiar with analyzing variable binding and scope rules.*

   This question concerns *scope rules*. Which of the following is true:

   (a) Languages with static scope rule are more likely to be compiled than are languages with dynamic scope rule; **
   (b) Languages with static scope rule are more likely to be interpreted than are languages with dynamic scope rule;
   (c) There is no relation between the scope rule of a language and the question of whether it will be compiled or interpreted
   (d) I have no idea.

7. *Learning outcome: Be familiar with analyzing parameter passing methods.*

   Suppose you are writing a function `f()` in C++; suppose this function has a parameter, `p1`. Consider a call to `f()`. Suppose that, in this call, `a1` is going to be passed as the argument corresponding to `p1`. The question is about whether the parameter `p1` of `f()` should be passed *by value* or *by reference*:

   (a) If you want the argument `a1` to be modifiable by `f()`, `p1` should be passed by-reference, not by-value; **
   (b) If you do NOT want the argument `a1` to be modified, `p1` should be passed by-reference, not by-value;
   (c) The question of whether `p1` should be passed by-value or by-reference has nothing to do with whether or not the argument should be modified;
   (d) I have no idea.

8. *Learning outcome: Be familiar with principles of object-oriented languages.*

   Consider the OO language C++. Which of the following statements is true?

   (a) Objects are compile-time entities and have no real existence at runtime whereas classes are runtime entities;
   (b) Classes are compile-time entities and have no real existence at runtime whereas objects are runtime entities; **
   (c) Both objects and classes exist at both compile-time and at runtime;
   (d) I have no idea.

9. *Learning outcome: Be familiar with implementing object-oriented languages.*

   Consider an OO language such as C++ or Java. Consider an object created with a call such as "new C(...)" where C is a class defined in the user program. Which of the following is true?

   (a) Space for this object will be allocated on the stack;
   (b) Space for this object will be allocated on the heap; **
   (c) Where space is allocated for the object varies from C++ to Java;
   (d) This depends on the details of the constructor function defined in the class C;
   (e) I have no idea.

10. *Learning outcome: Be familiar with using functional programming languages.*

    Consider a typical functional programming language such as Scheme and a typical imperative language such as C or C++. What is the *essential* difference between a functional programming language and an imperative language?

    (a) Functional languages use lists (and s-expressions) whereas imperative languages do not;
    (b) A typical program in a functional language consists of function definitions and function calls whereas a program in an imperative language consists of classes and objects;
    (c) A program in a functional language computes values that are combined in various ways to obtain other values whereas a program in an imperative language has a state that is modified as the program is executed; **
    (d) There is no particular essential difference between functional programming languages and imperative languages that applies to all functional and imperative languages; it depends on the particular languages in question;
    (e) I have no idea.

11. *Learning outcome: Be familiar with implementing functional programming languages.*

    Garbage collection is widely used in implementing functional programming languages because:

    (a) Programs in functional programming languages tend to produce a lot of garbage so garbage collection is essential;
    (b) Garbage collection improves efficiency of the programs;
    (c) Functional programs (and programmers) are not supposed to worry about memory management issues; **
    (d) This is a trick question; in fact, garbage collection is *not* widely used in implementing functional programming languages;
    (e) I have no idea.