CSE452 Exam II Review

Logic Programming and Data Abstraction:

1. Explain the purpose of the cut (!) in Prolog. How does it relate to **not**?

2. (Textbook problem, 14.8) Write Prolog rules to define a version of the member predicate that will generate all members of a list during backtracking, but without generating duplicates. Note that the cut and not based versions of Section 14.19 (pg. 529) will not suffice; when asked to look for an uninstantiated member, they find only the head of the list.

3. Explain the difference between *row-major* and *column-major* layout for contiguously allocated arrays. Why does a programmer need to know which layout the compiler uses? Why do most language designers consider row-major layout to be better?

Control Flow and Subroutines

4. (6.21) Given the following description for binary search:

   [Bentley] "We are to determine whether the sorted array X[1..N] contains the element T. Binary search solves the problem by keeping track of a range within the array in which T must be if it is anywhere in the array. Initially, the range is the entire array. The range is shrunk by comparing its middle element to T and discarding half the range. The process continues until T is discovered in the array or until the range in which it must lie is known to be empty."

   a. In your favorite programming language, write the code to produce the binary search according to the above requirements:

   b. What looping construct was the most useful for your program?

   c. Give the loop invariant(s) for your solution (indicate where you placed the invariant(s))?

5. (8.8) Write (in a language of your choice) a function or a procedure that will produce 4 different effects depending on whether the parameters are passed by value, by reference, by value-result, or by name.

6. (8.16) Why do you suppose that variable-length argument lists are so seldom used in high-level programming languages?

OO and Concurrency

7. Name three important benefits of abstraction.

8. What happens to the implementation of a class if we redefine a data member? For example, suppose we have:

```
class foo {
public:
    int a;
    char *b;
};
...
class bar : public foo {
public:
    float c;
    int b;
};
```

Does the representation of a bar object contain one b field or two? If two, are both accessible, or only one? Under what circumstances?

9. What is interrupt-driven I/O? What does it have to do with concurrency?

10. What is a critical section? Name two factors that must be satisfied to ensure the integrity of the critical section.

Test Your Understanding: (excerpted from the book)

1. The 3 main elements of object-orientation.
2. What does it mean for an expression to be referentially transparent?
3. What is the difference between a value model of variables and a reference model of variables? Why is the distinction important?
4. What is an l-value? An r-value?
5. Why is the distinction between mutable and immutable values important in the implementation of a language with a reference model of variables?
6. What does it mean for a language to be expression-oriented?
7. Why do most languages leave unspecified the order in which the arugments of an operator or function are evaluated?
8. What is short-circuit Boolean evaluation? Why is it useful?
9. Describe three common uses of the goto statement, and show how to avoid them using structured control- ow alternatives.
10. Why is sequencing a comparatively unimportant form of control ow in Lisp?
11. What does it mean for a function to be idempotent?
12. Explain why it may sometimes be useful for a function to have side effects.
13. Describe the jump code implementation of short-circuit Boolean evaluation.
14. Why do imperative languages commonly provide a case statement in addition to if. . . then. . . else?
15. Describe three different search strategies that might be employed in the implementation of a case statement, and the circumstances in which each would be desirable.
16. What is a tail recursive function? Why is tail recursion important?
17. Explain the difference between applicative and normal order evaluation of expressions. Under what circumstances is each desirable?
18. Describe three common pitfalls associated with the use of macros.
19. What is lazy evaluation? What are promises? What is memoization?
20. Give two reasons why lazy evaluation may be desirable.
21. Name a language in which parameters are always evaluated lazily.
22. Give two reasons why a programmer might sometimes want control ow to be nondeterministic.

Data Typing

23. What purpose(s) do types serve in a programming language?
24. What does it mean for a language to be strongly typed? Statically typed? What prevents, say, C from being strongly typed?
25. Name two important programming languages that are strongly but dynamically typed.