Homework 5: More Functional programming

Due: Wednesday, March 2.
This is a group homework.

1. (40 points) For each of the terms: *closure, first-class function, functional, higher-order, curried function, curry, polymorphism, scope, static scope, dynamic scope, hole in the scope, type checking, type inference, lazy evaluation, normal order evaluation, eager evaluation strict function, infinite list, lazy list, referential transparency* define the term as precisely as you can, give examples where possible, and state the importance of the term as used in functional programming and Standard ML.

   I suggest that your group split up the terms, and then get together to compare and criticize the definitions, examples, and discussions.

2. (15 points) Read Chapter 6 of Paulson's book and answer *one* of the following questions:

   (a) Critically evaluate the relationship between functional programming and program verification or transformation.

   (b) Compare the ease of reasoning about functional and object-oriented programs.

   I suggest that your group decide on one of these, read the chapter, write out a preliminary answer, and then meet to form a consensus answer.

3. (15 points, extra credit; applications) Describe three ways to apply the concepts of functional programming to other areas of computer science.

4. (15 points, extra credit; semantics, programming) Describe the situations in which mutation seems to ease programming, and explain by examples.

5. (15 points, extra credit; programming, applications — parallel processing) Both functional programming and object-oriented programming have been touted as ways to program concurrent or parallel computers. You might want to read some papers about this to help you with this question. Summarize the different approaches to extracting parallelism in a functional and object-oriented context. What implications does each strategy for extracting parallelism have on hardware (or what kinds of hardware best suit each strategy)?