Programming Languages and Compilers Comprehensive Exam

May 1999

Please answer each question in a separate booklet.

1. **Enter your answer in the booklet labeled 1.**

   It has been noted that first class functions, such as in Scheme, provide a mechanism for programming in an object oriented style.

   (a) Write, in Scheme, code that behaves similar to the C++ code below but uses first class functions instead of any built-in object oriented programming feature.

   ```
   class A {
   public:
     A(int vx, int vy) { x = vx; y = vy; }
     void print() { cout << "x = " << x << ", y = " << y << "\n"; }
     void increment() { y++; }
     int x;
   private:
     int y;
   };

   A a(3,4);
   ```

   Be sure to preserve the distinction between public and private members.

   (b) Inheritance, i.e. public derivation in C++, can also be simulated in Scheme. Write Scheme code that behaves similar to

   ```
   class B: public A {
   public:
     B(int vx, int vy, int vz) : A(vx,vy) { z = vz; }
     void decrement() { z--; }
     void print() { A::print(); cout << "z = " << z << "\n"; }
   private:
     int z;
   };

   B b(3,4,5);
   ```

2. **Enter your answer in the booklet labeled 2.**

   (a) In an imperative language, what does it mean for a variable to be *live* (in the context of live variable analysis).

   (b) Give the dataflow equations for live variable analysis. Be sure to define each function used in the equation. If you prefer, you may use some other analysis framework (e.g. abstract interpretation) for live variable analysis, as long as you describe the analysis formally.

   (c) Give the algorithm for solving the dataflow equations above (or whatever equations were induced by the analysis framework that you used).

3. **Enter your answer in the booklet labeled 3.**

   (a) Give an example of a grammar containing left recursion.

   (b) For what style of parsing does left recursion cause difficulties? Explain.

   (c) Give the algorithm for removing left recursion from a grammar. Show the results of applying this algorithm to the grammar you gave above.

   (d) For what style of parsing does left recursion not cause difficulty? Explain in detail why not.

4. **Enter your answer in the booklet labeled 4.**

   The default parameter passing mechanism in Algol60 is *pass-by-name*.

   (a) Pass-by-name is formally defined using the Algol Copy Rule. State that rule.

   (b) Consider the following Algol60 program

```
integer x,y;

procedure f(a,b);
integer a,b;
begin
   integer i;
   for i := 1 to 10 do
   begin
      b := b + 1;
      print(a);
   end;
end;

begin
  x := 0; y := 1;
  f(x+y, x);
end;
```

   Write, in Pascal or C, a program whose run-time behavior is identical (in terms of its implementation) to the above program. It is not sufficient to write a program that happens to produce the same output. Think about how Algol's parameter passing mechanism is actually implemented (it's not textual substitution), and have your program mimic that as much as possible.

5. **Enter your answer in the booklet labeled 5.**

   In the eight queens problem, one is required to place 8 queens on an 8x8 chess board without having any queen able to capture any other queen. The solution can be roughly described as follows:

   > A queen can be placed in row I of column J on a chess board in which queens have already been placed in columns 1 through J-1 iff
   >
   > - The queen does not conflict with any queens placed in columns 1 through J-1, and
   > - Either J is the last column of the chess board, or, after putting the queen in the row I and column J, another queen can be placed placed in some row of column J+1.
   >
   > (Note: A queen in row I column J conflicts with a queen in row I' column J' if one of the following holds: $I = I', J = J', |I - I'| = |J - J'|$.)

(a) Implement the eight queens problem in Prolog, making use of Prolog's backtracking mechanism. Your program should return one solution to the 8-queens problem at a time and be able to return the next solution (if there is one) if requested.

(b) Implement the eight queens problem using a parallel search in Ada. It should print out every solution. Don't worry if you don't know the exact syntax of Ada, but be sure to show that you know what Ada's model of parallelism is.

6. **Enter your answer in the booklet labeled 6.**

(a) Define the term *basic block* as used in the context of compilation.

(b) Give an algorithm for indentifying the basic blocks in a program that is represented in three-address code.

(c) Why is it useful to represent a program as a set of basic blocks? How do basic blocks aid in the optimization phase?

(d) Give an example of a local (i.e. intra-basic block) optimization. Show the transformation of a basic block as the result of this optimization.