Compiler Design

Assignment No. 1
Dataflow Analysis
Due January 24, 2006

1. Give examples to show why the assumptions in reaching definitions are conservative, in the sense that the set of values that is assumed to reach can be, in general, a superset of the definitions that actually reach.

2. Problem 10.5 in Aho-Sethi-Ullman.

3. Another use of live variable information is the removal of useless assignments. Can you illustrate this with a small example.

4. A concept related to dead variables is the notion of a faint variable. At any point, the faint variables are a superset of the dead variables. A variable x at a point is faint if it is dead at that point, or if it is live only if it is being used in an assignment to a faint variable. A variable used in a control predicate or a print statement is never faint at the point at which it is used. The concepts of liveness and faintness are illustrated in the following example program; comments refer to the program point just after the corresponding statement.

```
main() {
    i = 0; // i is live and not faint
    x = i; // x is live and faint
    y = x; // y is dead and faint
    while (i <= 10){ // i is live and not faint
        x = x + 1; // x is live and faint
        i = i + 1; // i is live and not faint
    } // i is live and not faint
      // x is dead and faint
    print(i); // i is dead and faint
}
```

   (a) Recall that live/dead variable analysis can be used to find and remove useless assignments. What are the advantages of using faintness information in addition to live/dead information in removing useless assignments?

   (b) Call a variable that is not faint truly live. True-liveness identifies the minimal set of truly-live variables at each basic block. Give the data flow equations and a segment of code to compute truly live variables at the beginning and end of each basic block.