

CMPS 290G – Topics in Software Engineering
Winter 2004 – Software Validation and Defect Detection
Homework 1

Due: 22 Jan 2004

1. Give five examples of specifications that you believe would be appropriate “implicit” specifications for the vast majority of programs. Include at least one liveness specification.
2. Give five examples of specifications that would be appropriate for some program you have written. Include at least one liveness specification.
3. Prove that there is no algorithm A , that, given a program P , decides if P can ever fail a run time assertion on some input. You may assume the undecidability of the halting problem. Hint: Show how to implement the predicate $halt(P)$ using A .
4. Write the following program as a control-flow graph.

```
y = 0;
while (x>0) {
  if (even(x)) {
    x = x-1;
    y = y+1;
  } else {
    x = x-1;
  }
}
```

5. Compute the “available expressions” at the entry and exit points of each node in this graph.
6. On a separate copy of the graph, compute the “live variables” at the entry and exit points of each node in the graph.
7. Characterize the distinction between flow-sensitive and flow-insensitive analyses.
8. In the format of lecture 2 slide 12, write the type derivation for the program:

$$\lambda x : \text{int} \rightarrow \text{int}. \lambda y : \text{int}. (x \ y)$$

9. Given the untyped program:

$$\lambda x. \lambda y. (y \ x)$$

derive the corresponding constraints and solve them (using the format of lecture 3a slide 25).

10. Do the same for the untyped program:

$$\lambda x. (x \ x)$$

11. Extra credit: Splint is descendent of LCLint and is available at <http://www.splint.org/>. Evaluate Splint on some C program, preferably one you have written. (You may collaborate with classmates on the Splint installation.) Write a short report (less than one page) about your experience, including an evaluation of the usefulness of Splint.