Checkin 18

Give an example of a program analysis method which is sound but not complete.

Administrivia

Flipped Wednesday

Written Work #5

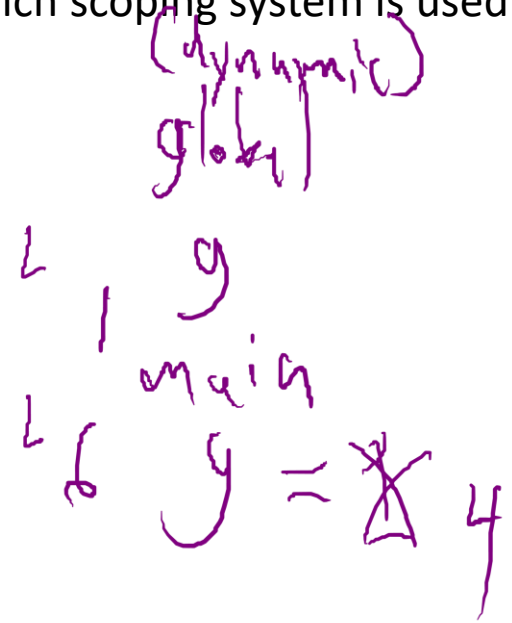
Topics:

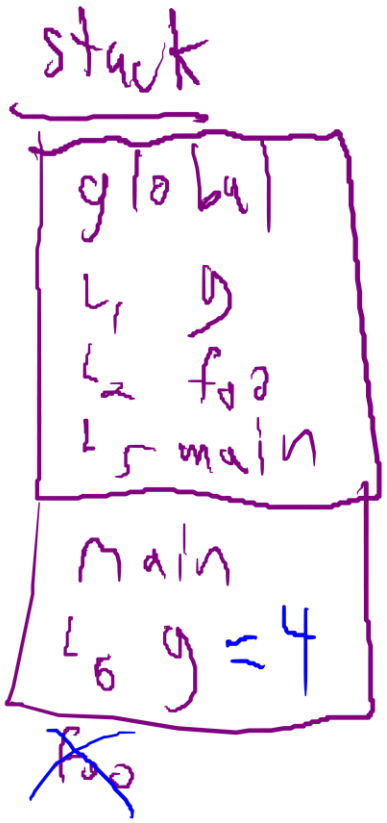
- Saying farewell to parsers (LR)
- Name Analysis
- Type Analysis

Written Work #5: Question 1

Recall that Levi is a statically-scoped language. Assume a new variant of Levi, called Levi', that uses the exact same syntax as Levi but is dynamically-scoped. Provide an input file that would be valid in both Levi and Levi', but prints a different result based on which scoping system is used. Show what the output would be either under scheme.

```
1  g : int;
2  foo : () void {
3      g = 4
4  }
5  main : () void {
6      g : int = 1
7      foo();
8      out << g;
9  }
```

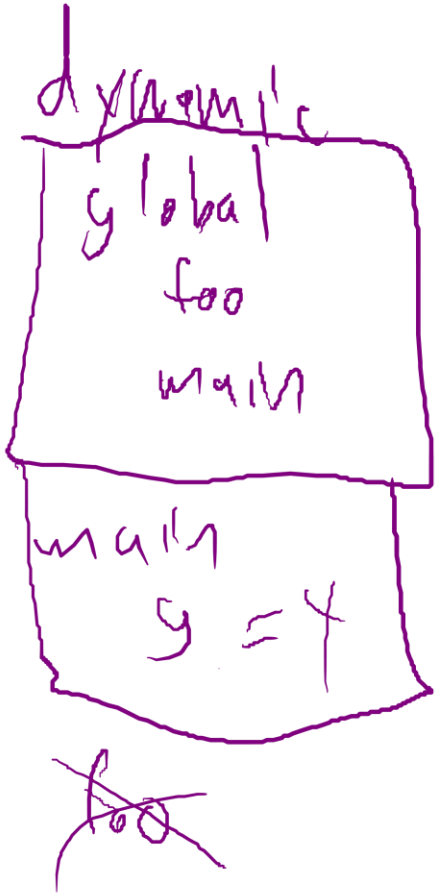




Written Work #5: Question 2

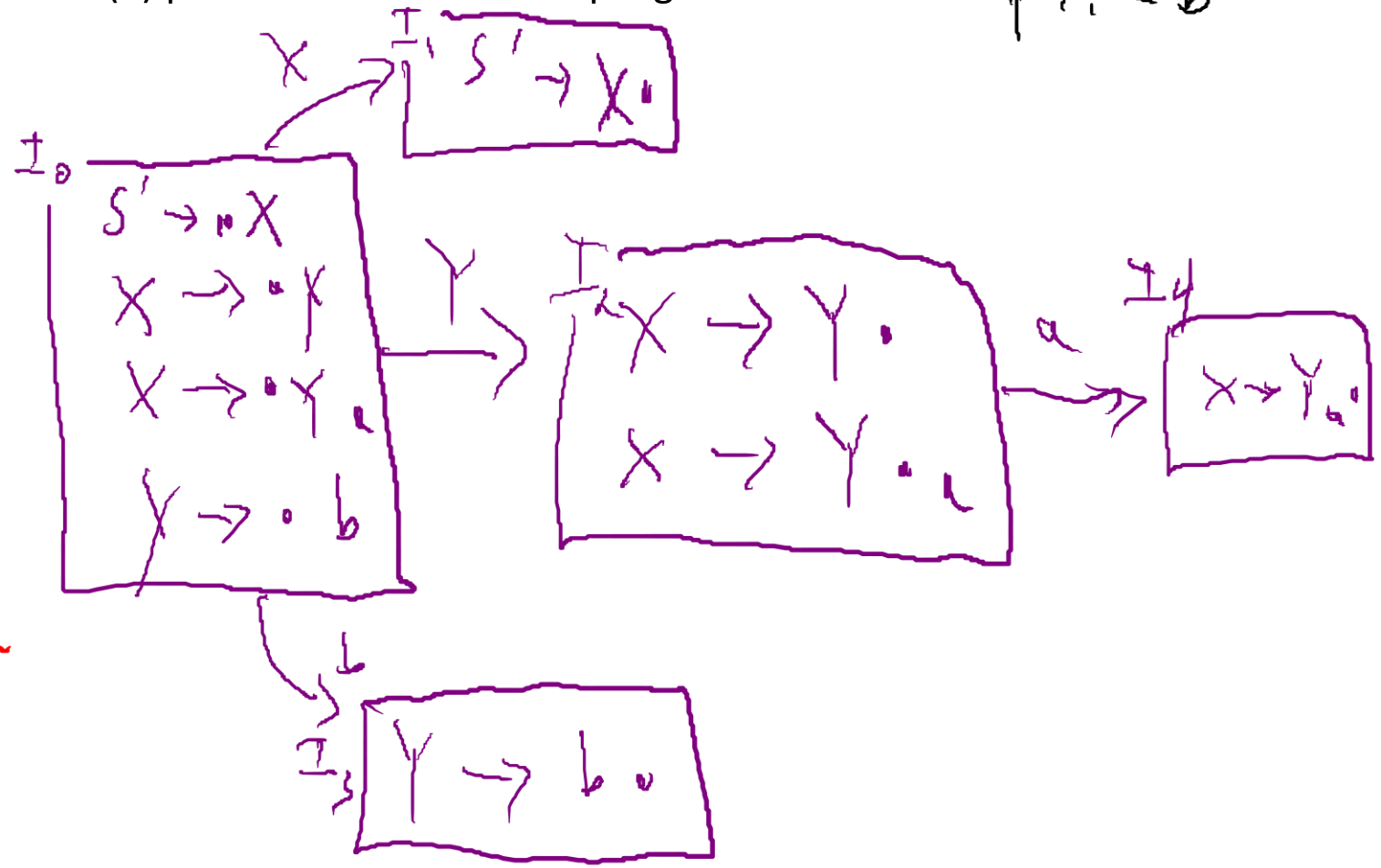
Provide an input file that would be valid in Levi', but **not** in Levi.

```
foo() void {
    y = 4;
}
main : () void {
    g : int = 1;
    foo();
    cout << g;
}
```



Written Work #5: Question 3

Explain the difference between an SLR and an LR(0) parser. Provide an example grammar that is SLR-parseable but not LR(0) parseable.



Written Work #5: Question 3

Explain the difference between an SLR and an LR(0) parser. Provide an example grammar that is SLR-parseable but not LR(0) parseable.

Written Work #5: Question 4 (bonus!)

Explain how a type analysis that propagates an error type reduces cascading error reports.
Give an example of a program where propagating an error type might reduce the number of reports
Give an example of a program where propagating the error type still results in multiple reports within the same expression