Lecture #26: Project Strategy

```
class B (Object):
   m = 3

a1 = 3

def f1(x2, y2):
   b2 = False
   while y2 > 0:
     if b2:
       print a1 + x2 + d2
     b2 = True
     d2 = b1.m
     y2 -= 1

b1: B
b1 = B()
```

- Must process assignments to d2, b1 *before* can handle their (earlier) uses.
- But don't need to process assignments in f1 when looking at uses outside f1.
- Cannot process b1.m until we know b1's static type.
- Cannot attach b1's static type until we have decided what b1 is (what it's declaration is).

So... An Algorithm

```
class B (Object):
   m = 3

a1 = 3

def f1(x2, y2):
   b2 = False
   while y2 > 0:
     if b2:
       print a1 + x2 + d2
     b2 = True
     d2 = b1.m
     y2 -= 1

b1: B
b1 = B()
```

- To connect simple identifiers to declarations:
  - First find all declarations at outer level (identifiers ending in 1 in example). Add to symbol table.
  - Then find all uses and attach declarations.
  - As you find uses, go into each 'def' you find and
    - Start a new block in the symbol table.
    - (Recursively) connect identifiers to declarations within its body.
    - Exit block.
- Now that all ids have declarations attached, symbol table no longer needed.
- Now pass through entire tree and attach types from type declarations.
- Finally, another pass (now that you know types) to handle things like b1.m.