CS 3344
Lecture 10
Scope & Runtime Storage

Block Structured Code

1. Local: vars declared in current block.
2. Global: vars declared in enclosing blocks.
3. Scope: where in text a declaration is visible.
4. Lifetime: period of time that mem. is allocated for a variable.
Activation Record

- Push A.R. when entering block.
- Pop " " exiting block.
- Local: in top A.R. on stack.
- Global: go back through control links.
  - # of links to follow
  - == # of blocks entered between decl & use of var.

Top level Decs

- Each one is in a new scope
  So new AR at runtime.
sumSquares

control
sumSquares -> code.

control
X

code for line 10.

control

return addr
return result

n  15
i  0
sum  0
... temps...
Parameter Passing

1) In calling `fact(n-1)`, when do you compute `n-1`?
   - eager: eval before call
   - lazy: eval when a param is used for the first time.
How are variables passed to functions?

Pass by Value:
- Vars are passed as R-values.

Pass by Ref:
- Vars are passed as L-values.
Java: primitives by val
  \( \text{int, char, double} \).
  Objects/arrays by ref.
void update(Vector v) {
  v.add(3);
}

ML, C, C++, Pascal, ...
- Explicit language constructs
  void swap(int \&x, int \&y) {
    int t=x; x=y; y=t;
  }
  Swap(a, b);
Accessing Globals

1. Dynamic Scope.
   - Variable refers to the most recently processed decl. \( \Rightarrow \) "newest"
     \( \Rightarrow \) closest to top of stack.
     - Search back through ARs.

2. Static Scope
   - Variable refers to decl. in the closest lexically enclosing scope in source code.