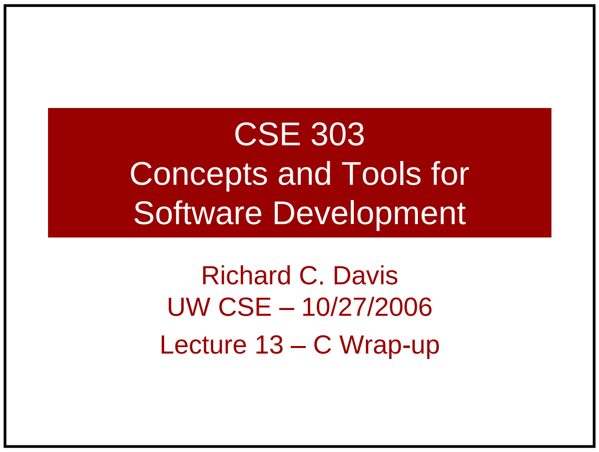
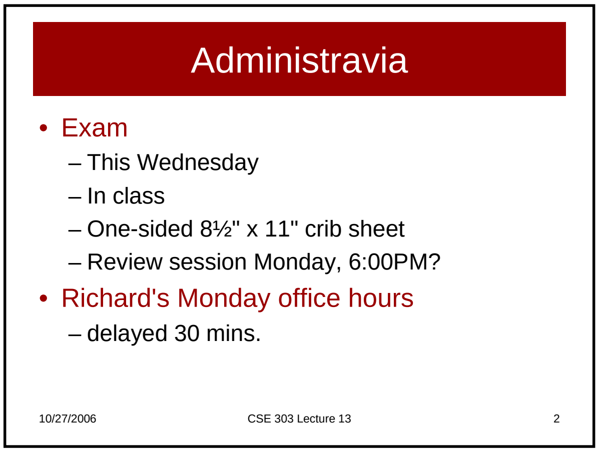


## CSE 303 Concepts and Tools for Software Development

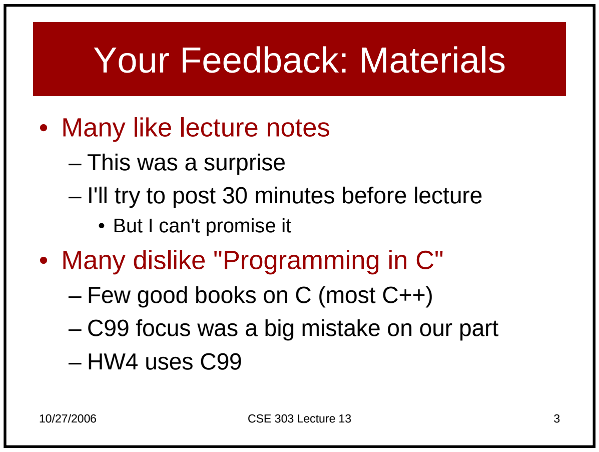
Richard C. Davis
UW CSE – 10/27/2006

Lecture 13 – C Wrap-up



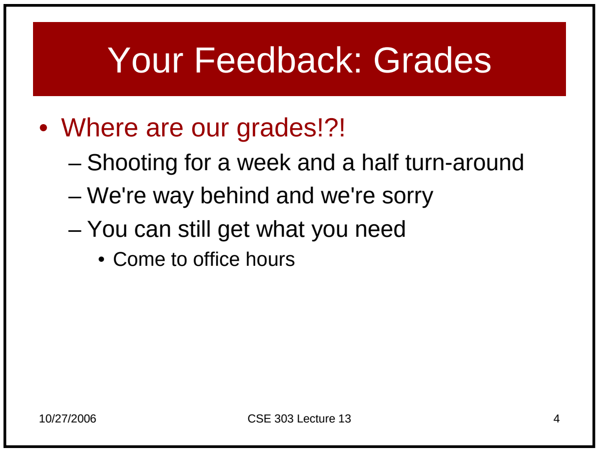
## Administravia

- Exam
  - This Wednesday
  - In class
  - One-sided 8½" x 11" crib sheet
  - Review session Monday, 6:00PM?
- Richard's Monday office hours
  - delayed 30 mins.



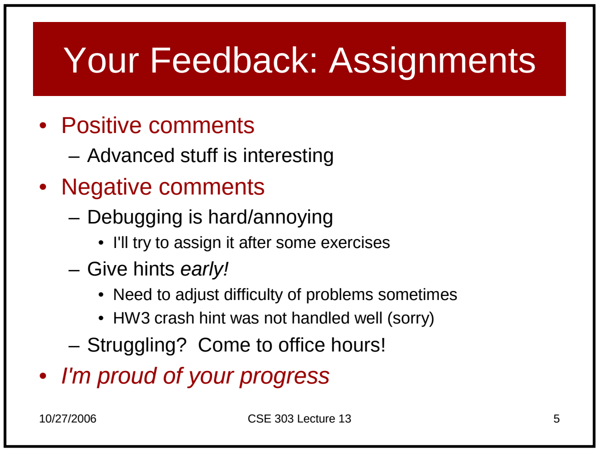
## Your Feedback: Materials

- Many like lecture notes
  - This was a surprise
  - I'll try to post 30 minutes before lecture
    - But I can't promise it
- Many dislike "Programming in C"
  - Few good books on C (most C++)
  - C99 focus was a big mistake on our part
  - HW4 uses C99



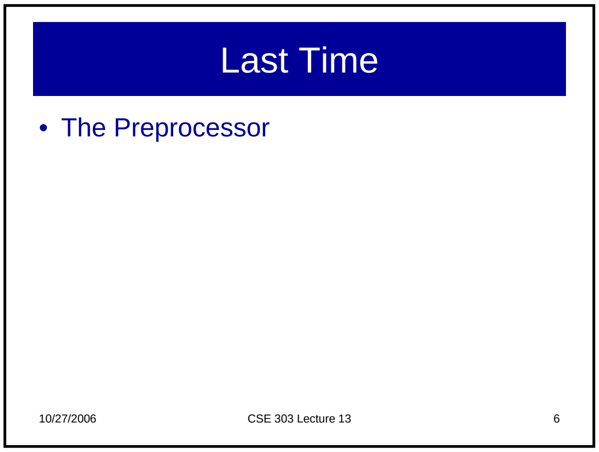
## Your Feedback: Grades

- Where are our grades!?!
  - Shooting for a week and a half turn-around
  - We're way behind and we're sorry
  - You can still get what you need
    - Come to office hours



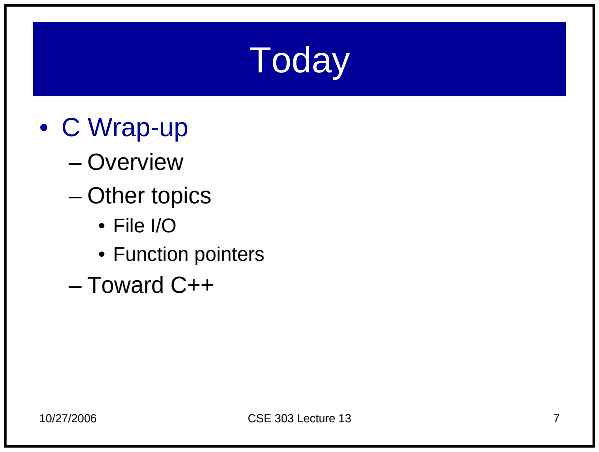
## Your Feedback: Assignments

- Positive comments
  - Advanced stuff is interesting
- Negative comments
  - Debugging is hard/annoying
    - I'll try to assign it after some exercises
  - Give hints *early!*
    - Need to adjust difficulty of problems sometimes
    - HW3 crash hint was not handled well (sorry)
  - Struggling? Come to office hours!
- *I'm proud of your progress*



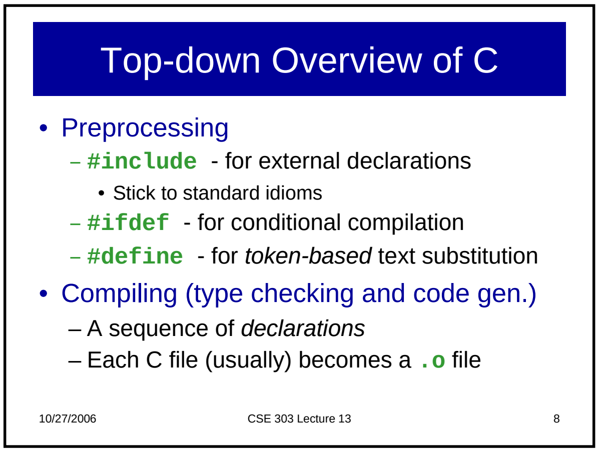
## Last Time

- The Preprocessor



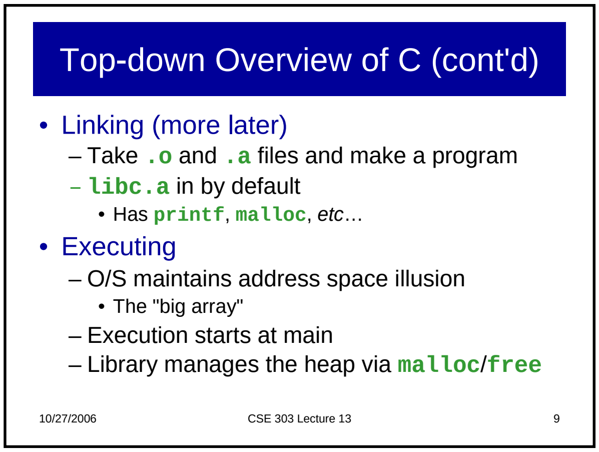
## Today

- C Wrap-up
  - Overview
  - Other topics
    - File I/O
    - Function pointers
  - Toward C++



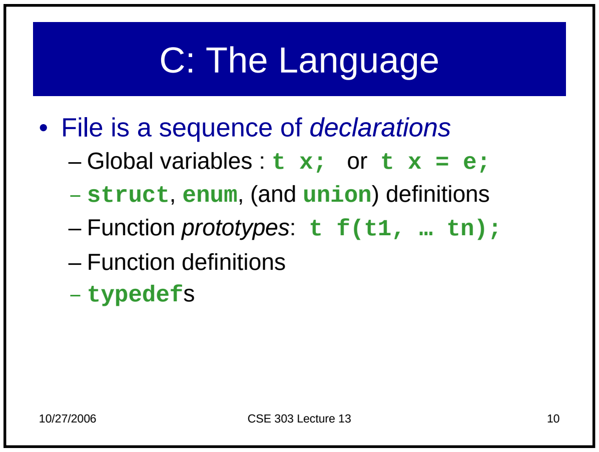
## Top-down Overview of C

- Preprocessing
  - `#include` - for external declarations
    - Stick to standard idioms
  - `#ifdef` - for conditional compilation
  - `#define` - for *token-based* text substitution
- Compiling (type checking and code gen.)
  - A sequence of *declarations*
  - Each C file (usually) becomes a `.o` file



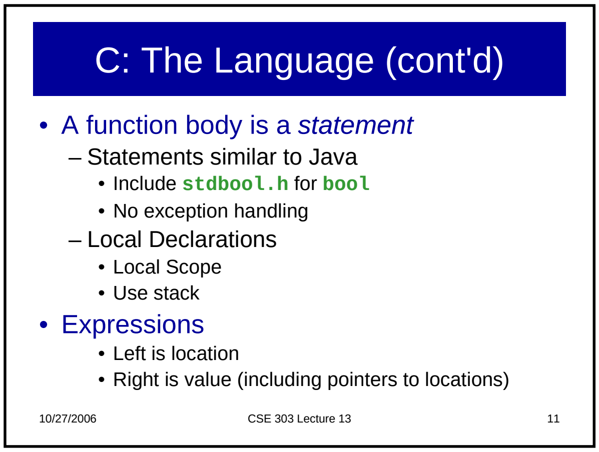
## Top-down Overview of C (cont'd)

- Linking (more later)
  - Take `.o` and `.a` files and make a program
  - `libc.a` in by default
    - Has `printf`, `malloc`, *etc*…
- Executing
  - O/S maintains address space illusion
    - The "big array"
  - Execution starts at main
  - Library manages the heap via `malloc`/`free`



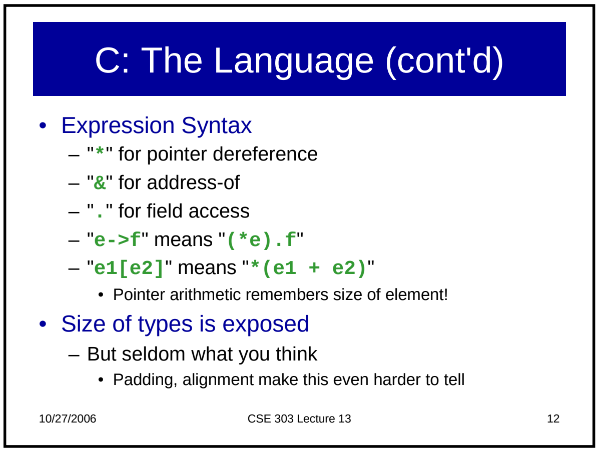
## C: The Language

- File is a sequence of *declarations*
  - Global variables : `t x;` or `t x = e;`
  - `struct`, `enum`, (and `union`) definitions
  - Function *prototypes*: `t f(t1, … tn);`
  - Function definitions
  - `typedef`s



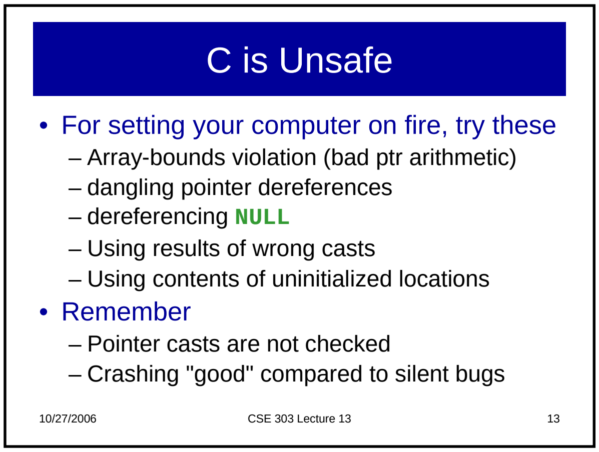
## C: The Language (cont'd)

- A function body is a *statement*
  - Statements similar to Java
    - Include `stdbool.h` for `bool`
    - No exception handling
  - Local Declarations
    - Local Scope
    - Use stack
- Expressions
    - Left is location
    - Right is value (including pointers to locations)



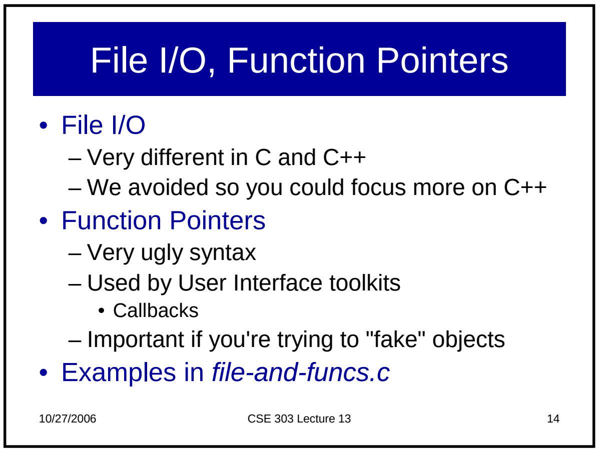
## C: The Language (cont'd)

- Expression Syntax
  - "`*`" for pointer dereference
  - "`&`" for address-of
  - "`.`" for field access
  - "`e->f`" means "`(*e).f`"
  - "`e1[e2]`" means "`*(e1 + e2)`"
    - Pointer arithmetic remembers size of element!
- Size of types is exposed
  - But seldom what you think
    - Padding, alignment make this even harder to tell



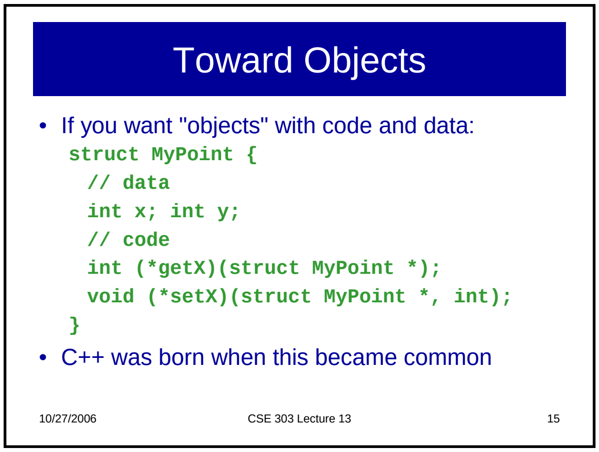
## C is Unsafe

- For setting your computer on fire, try these
  - Array-bounds violation (bad ptr arithmetic)
  - dangling pointer dereferences
  - dereferencing `NULL`
  - Using results of wrong casts
  - Using contents of uninitialized locations
- Remember
  - Pointer casts are not checked
  - Crashing "good" compared to silent bugs



## File I/O, Function Pointers

- File I/O
  - Very different in C and C++
  - We avoided so you could focus more on C++
- Function Pointers
  - Very ugly syntax
  - Used by User Interface toolkits
    - Callbacks
  - Important if you're trying to "fake" objects
- Examples in *file-and-funcs.c*

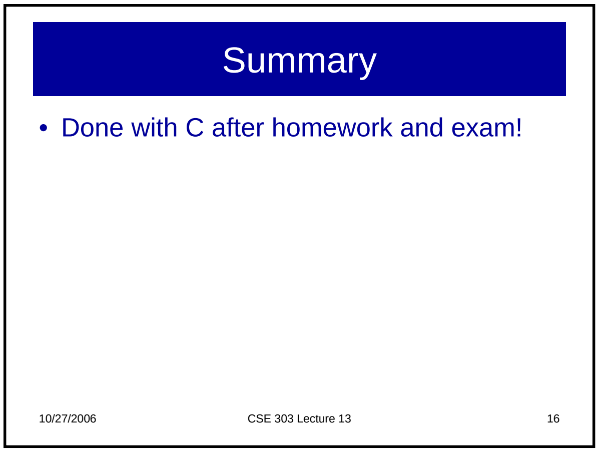


## Toward Objects

- If you want "objects" with code and data:

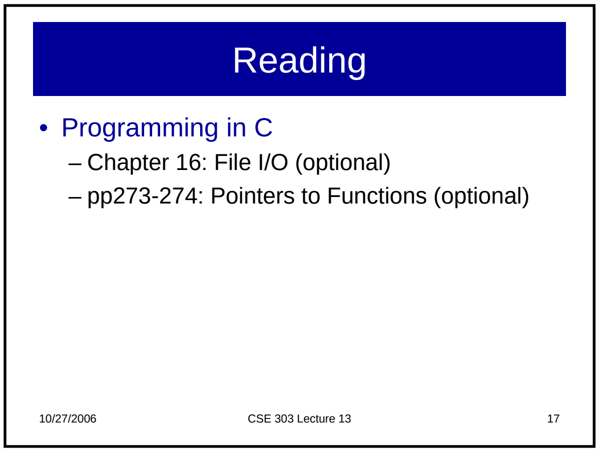
```
struct MyPoint {
  // data
  int x; int y;
  // code
  int (*getX)(struct MyPoint *);
  void (*setX)(struct MyPoint *, int);
}
```

- C++ was born when this became common



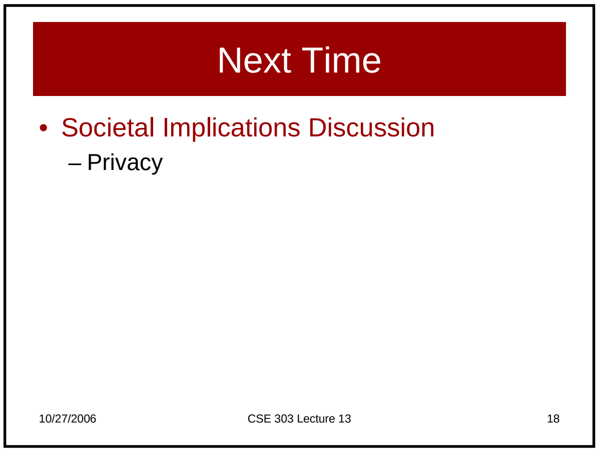
## Summary

- Done with C after homework and exam!



## Reading

- Programming in C
  - Chapter 16: File I/O (optional)
  - pp273-274: Pointers to Functions (optional)



## Next Time

- Societal Implications Discussion
  - Privacy