CS603 Programming Language Organization

8 March 2004

Spring 2004

Department of Computer Science

Overview

- Questions on MP2
- Midterm
- Introduction to ML and type inference

Midterm

- In class, 17 March
- Study guide and review in class Wednesday, 10 March

What the heck was that?

- We skipped a lot of material that is needed to understand chapter 7
- But we'll use that as motivation to go back and grab the bits we need from chapters 5 and 6.

Why do we care about types?

Pair up:
What are types?
What good are they?

- Types are constraints on the values that a variable or expression can have
- Type checking is a way of detecting errors
  - Static checking can detect errors sooner
  - But requires programmer to supply types

Static Typing

- Static typing is the checking of types when a program is compiled

Pair up:
Is Impcore statically typed?
μ-Scheme?

Typed Impcore

```
toplevel   ::=  exp
            |   (use file-name)
            |   (val variable-name exp)
            |   (define type function-name (formals) exp)
exp        ::=  value
            |   variable-name
            |   (set variable-name exp)
            |   (if exp exp exp)
            |   (while exp exp)
            |   (begin {exp})
            |   (function {exp})
formals    ::=  {(type variable-name)}
type       ::=  int |bool |unit |(array type)
value      ::=  integer
function   ::=  function-name | primitive
primitive  ::=  + |- |* |/ |= |< |> |print
```

Typed Impcore Examples

- `(define bool and ((bool b) (bool c)) (if b c b)`
- ```
  (define int mod ((int m) (int n)
      (- m (* n (/ m n))))
  ```
  ```

Type System for Typed Impcore

- *basic type*— `INT`, `BOOL`, or `UNIT`
- *simple type*— basic type or type constructor `ARRAY` applied to a simple type τ
- *type constructor*— `ARRAY`
- *function types*— $\tau_f ::= \tau_1 \times \cdots \times \tau_n \rightarrow \tau_n$
- *type environments*— Γ_ξ, Γ_φ, Γ_ρ globals, functions, formal parameters