

- values defined by t
- SML, OCAML, Scheme and Ada have sound type systems
- Most implementations of C and C++ do not

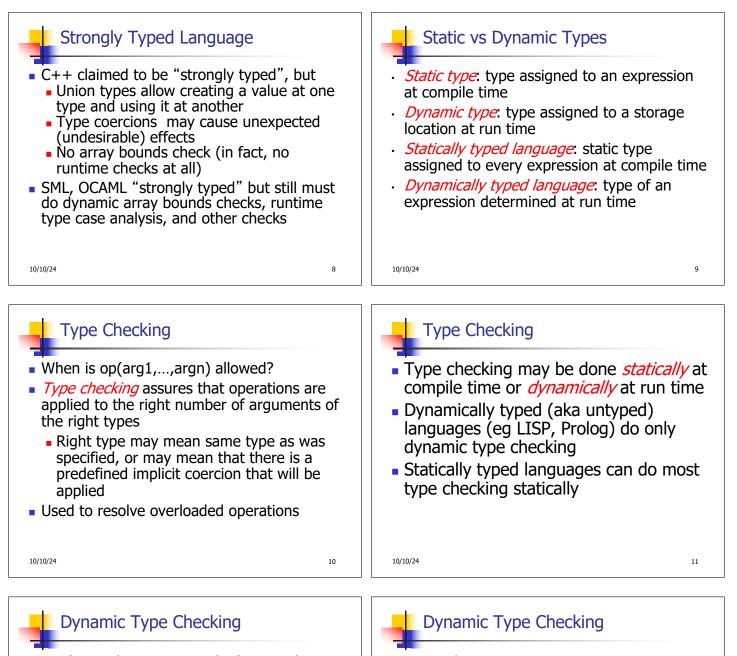
10/10/24

6

- arguments can lead to a run-time type error, language is *strongly typed* 
  - Eq: 1 + 2.3;;
- Depends on definition of "type error"

7

10/10/24



- Performed at run-time before each operation is applied
- Types of variables and operations left unspecified until run-time
  - Same variable may be used at different types

10/10/24

12

- Data object must contain type information
- Errors aren't detected until violating application is executed (maybe years after the code was written)

10/10/24

13

<ul> <li>Static Type Checking</li> <li>Performed after parsing, before code generation</li> <li>Type of every variable and signature of every operator must be known at compile time</li> </ul>	<ul> <li>Static Type Checking</li> <li>Can eliminate need to store type information in data object if no dynamic type checking is needed</li> <li>Catches many programming errors at earliest point</li> <li>Can't check types that depend on dynamically computed values</li> <li>Eg: array bounds</li> </ul>
10/10/24 14	10/10/24 15
<ul> <li>Static Type Checking</li> <li>Typically places restrictions on languages</li> <li>Garbage collection</li> <li>References instead of pointers</li> <li>All variables initialized when created</li> <li>Variable only used at one type</li> <li>Union types allow for work-arounds, but effectively introduce dynamic type checks</li> </ul>	<ul> <li>Type Declarations</li> <li>Type declarations: explicit assignment of types to variables (signatures to functions) in the code of a program</li> <li>Must be checked in a strongly typed language</li> <li>Often not necessary for strong typing or even static typing (depends on the type system)</li> </ul>
10/10/24 16	10/10/24 19
<ul> <li>Type Inference</li> <li><i>Type inference</i>: A program analysis to assign a type to an expression from the</li> </ul>	<ul> <li>Format of Type Judgments</li> <li>A <i>type judgement</i> has the form</li> <li>Γ  - exp : τ</li> <li>Γ is a typing environment</li> </ul>

- program context of the expression
- Fully static type inference first introduced by Robin Milner in ML
- Haskell, OCAML, SML all use type inference
  - Records are a problem for type inference

10/10/24

20

Format of Type Judgments
A type judgement has the form

Γ | - exp : τ

F is a typing environment

Supplies the types of variables (and function names when function names are not variables)
Γ is a set of the form { X:σ,...}
For any X at most one σ such that (X: σ ∈ Γ)

exp is a program expression

τ is a type to be assigned to exp
pronounced "turnstyle", or "entails" (or "satisfies" or, informally, "shows")

