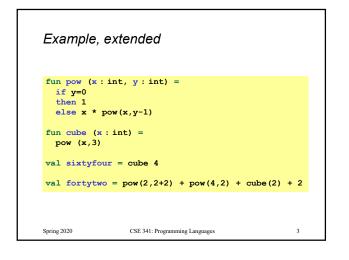
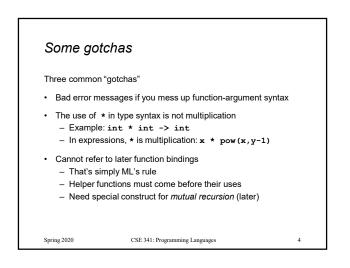
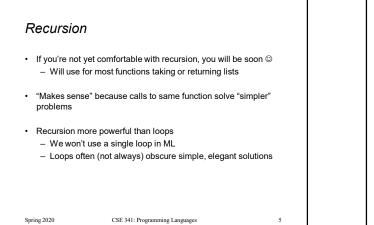
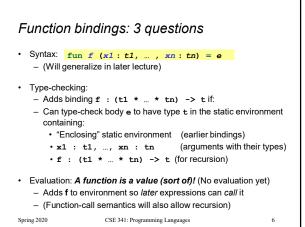
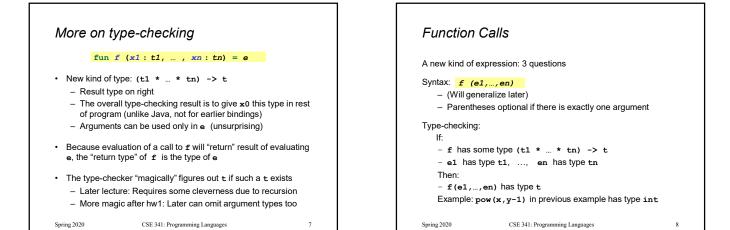
PAUL G. ALLEN SCHOOL of computer science & engineering Function definitions Functions: the most important building block in the whole course - Like Java methods, have arguments and result - But no classes, this, return, etc. CSE341: Programming Languages Example function binding: Lecture 2 (* Note: correct only if y>=0 *) Functions, Pairs, Lists fun pow (x : int, y : int) = if y=0 then 1 else x * pow(x,y-1) Brett Wortzman Spring 2020 Note: The body includes a (recursive) function call: pow(x,y-1) Spring 2020 CSE 341: Programming Languages 2











Function-calls continued
f(e1,...,en)
Evaluation:
1. (Under current dynamic environment,) evaluate f to a function fun f (x1 : t1, ..., xn : tn) = e

Since call type-checked, result will be a function

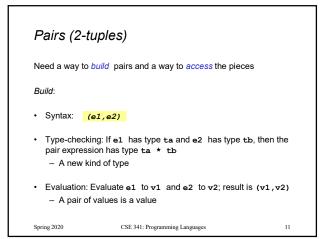
2. (Under current dynamic environment,) evaluate arguments to values v1, ..., vn
3. Result is evaluation of e in an environment extended to map x1 to v1, ..., xn to vn

("An environment" is actually the environment where the function was defined, and includes f for recursion-more on this later)

CSE 341: Programming Languages

Spring 2020

Definition of the problem o



Pairs (2-tuples)
Need a way to <i>build</i> pairs and a way to <i>access</i> the pieces
Access:
• Syntax: <mark>#1 e</mark> and <mark>#2 e</mark>
 Type-checking: If e has type ta * tb, then #1 e has type ta and #2 e has type tb
 Evaluation: Evaluate e to a pair of values and return first or second piece Example: If e is a variable x, then look up x in environment
Spring 2020 CSE 341: Programming Languages 12

