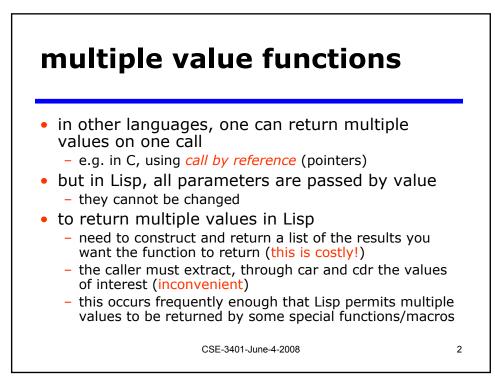
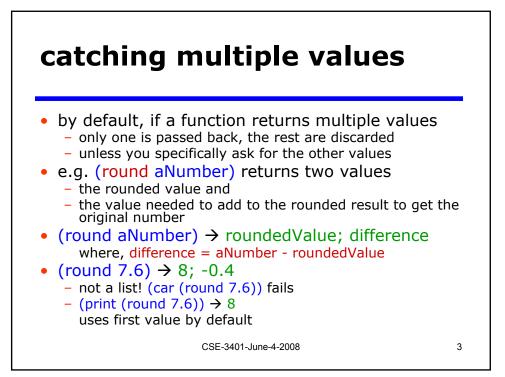
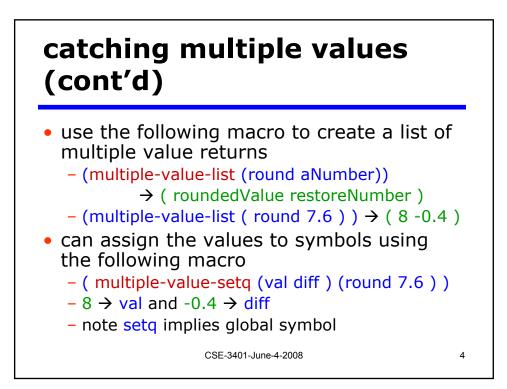
## Multiple Value Functions (see Wilensky Chapter 16.4)

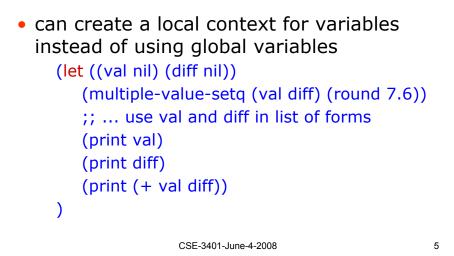
Shakil M. Khan adapted from Gunnar Gotshalks

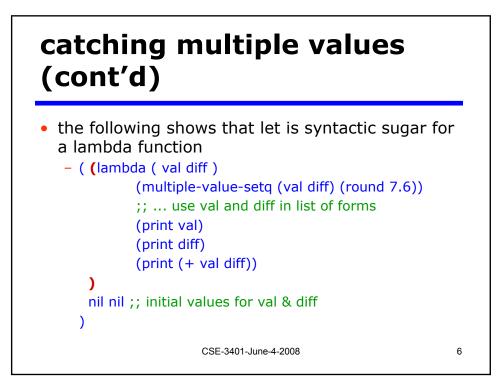


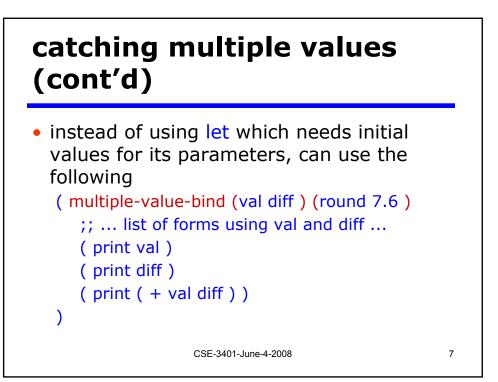


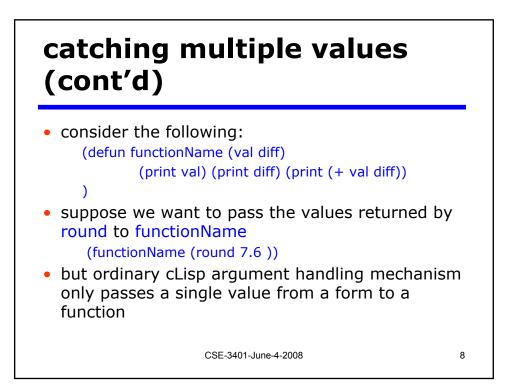


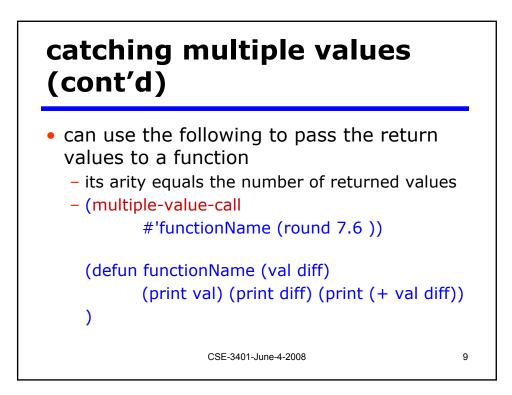


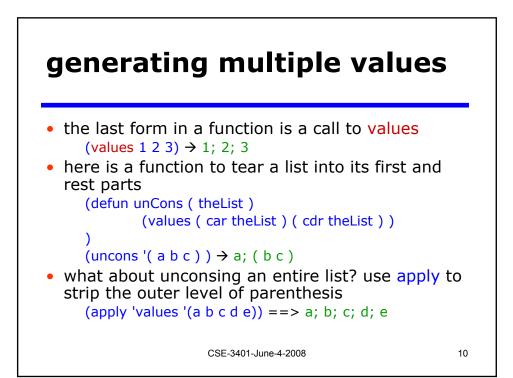












## generating multiple values (cont'd)

- can use values (with no arguments) to build a Lisp function that returns no values at all
  - the last statement of the function: (values)
  - useful if a function is used only for its sideeffects
  - Lisp supplies a gratuitous value of nil if such a function is used in a context where it is not anticipated
    - e.g. (print (values)) → nil

CSE-3401-June-4-2008

11

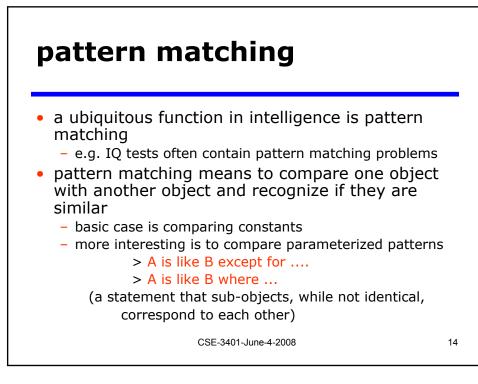
## consequences of passing around multiple values

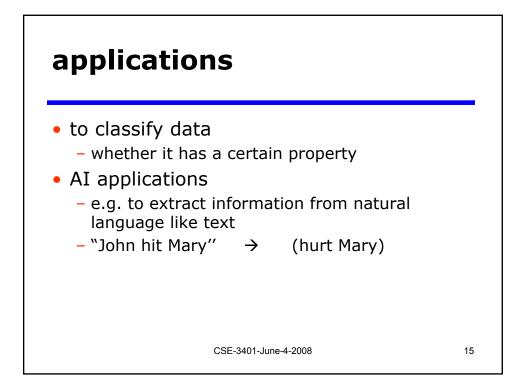
- has some consequences on the design of a Lisp system
- recall: code that does not anticipate multiple values ignores all but the first value
- however, most built-in functions/macros that tends to return the values of other forms are set up to handle multiple values
  - e.g. cond, let, ... passes multiple values
  - and, or passes multiple values from the last sub-forms
  - (and nil (values 3 4)) → nil (and t (values 3 4)) → 3; 4, (and (values 1 2) t) → t

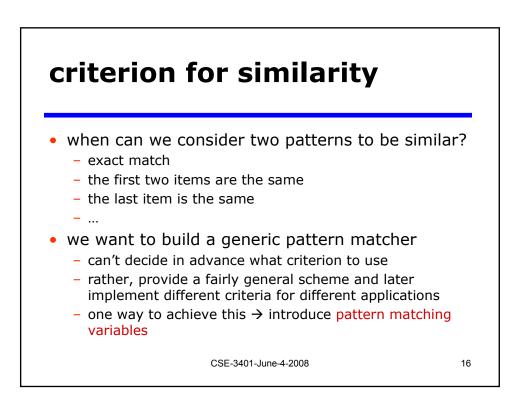
CSE-3401-June-4-2008

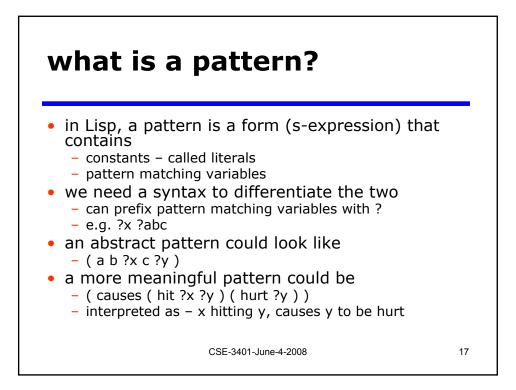
## Pattern Matching (see Wilensky Chapter 21)

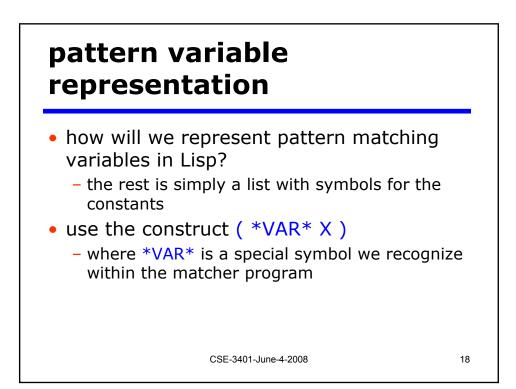
Shakil M. Khan adapted from Gunnar Gotshalks

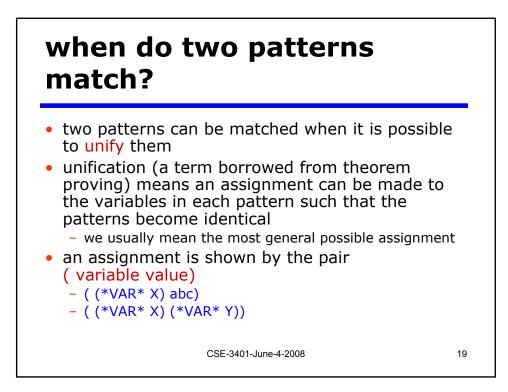


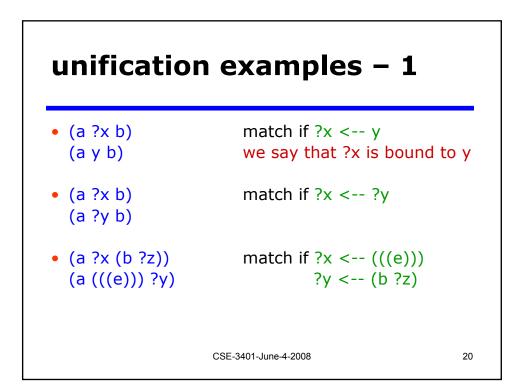


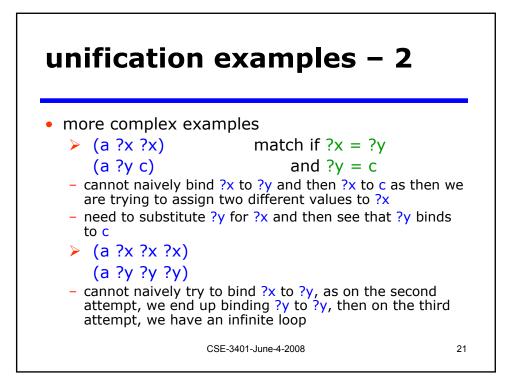


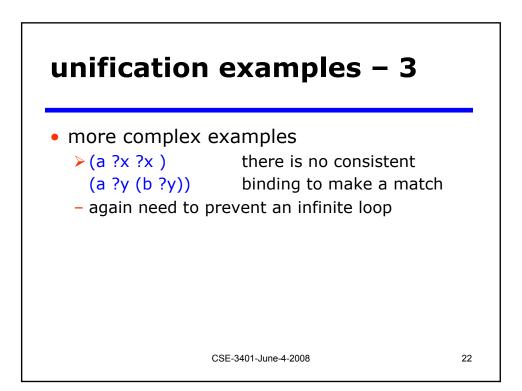


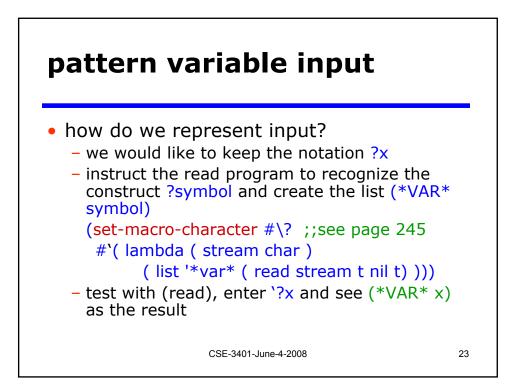


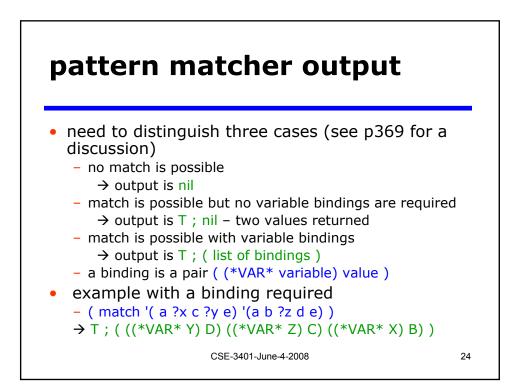


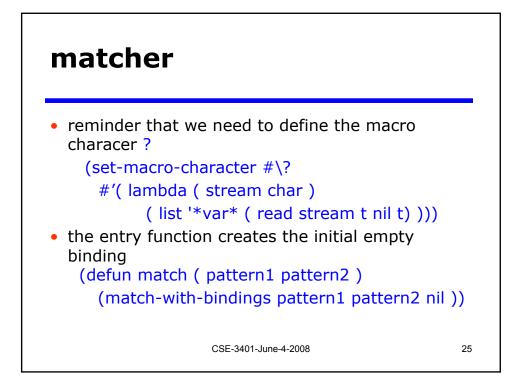


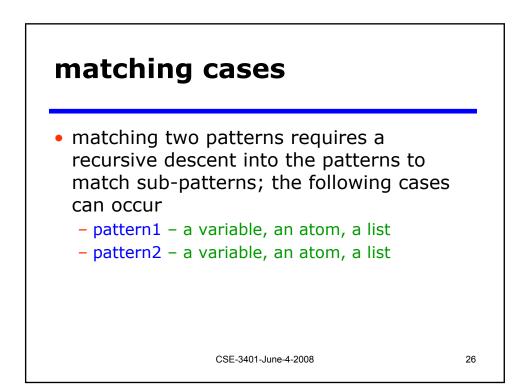












matching cases (cont'd)		
<ul> <li>the matching program has to examine the possible combinations</li> </ul>		
<ul> <li>pattern1</li> </ul>	pattern2	result
atom	atom	match if equal, else no match
atom	variable	try to bind atom to variable
atom	list	no match
variable	atom	try to bind atom to variable
variable	variable	try to bind variable to variable
variable	list	try to bind list to variable
list	atom	no match
list	variable	try to bind list to variable
list	list	recursive descent on first and rest
CSE-3401-June-4-2008 27		

