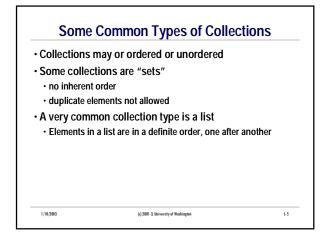
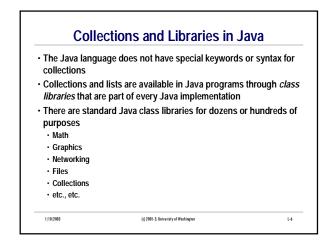
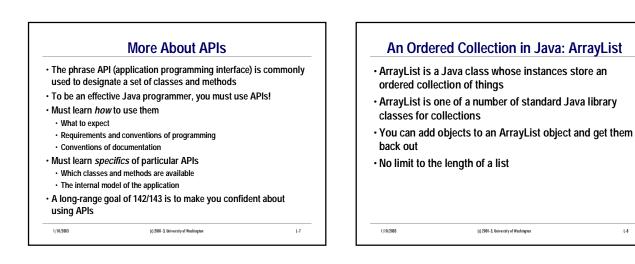


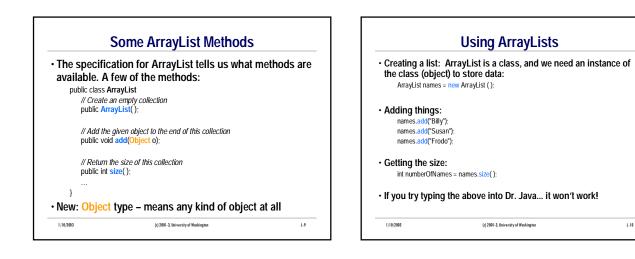
L-4

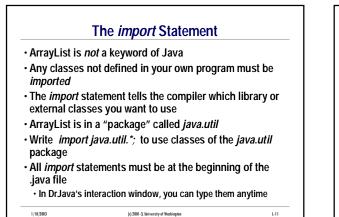


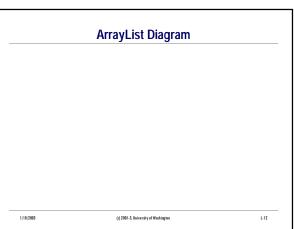


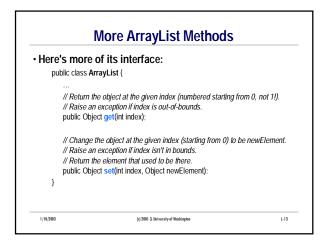


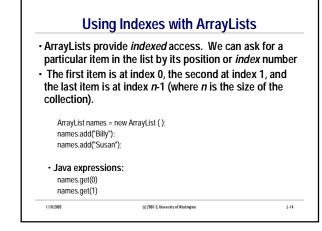
L-8

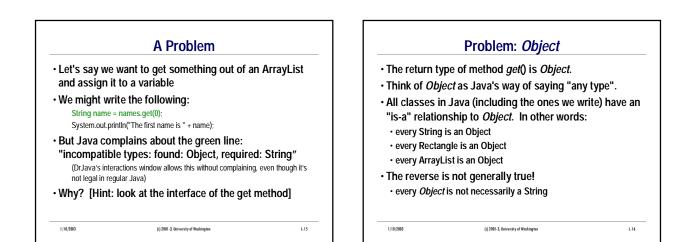


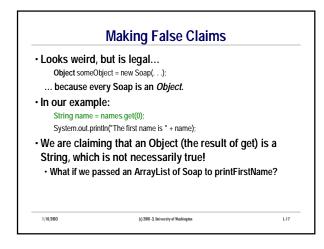


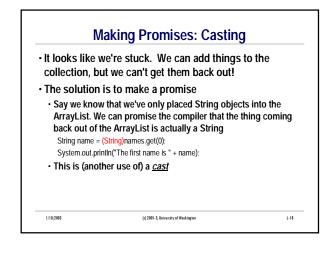


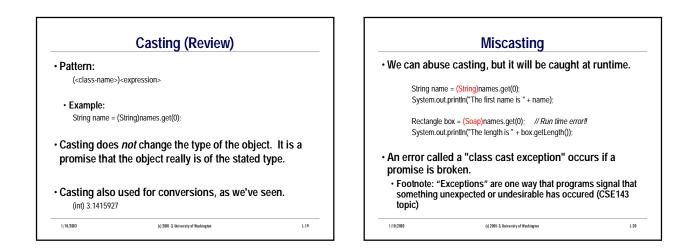












## **Reference vs. Primitive Types**

## • A few Java types are *primitive*

- int, double, char, boolean, and a few other numeric types we normally won't use
- · Are atomic chunks, with no parts (i.e., no instance variables)
- · Exist without having to be allocated with new
- Cannot receive messages (i.e., do not have methods) but can be arguments of messages and unary and binary operators

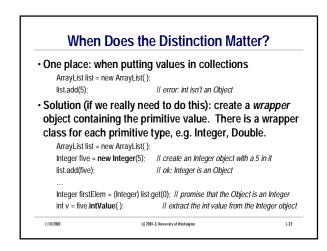
(c) 2001-3, University of Washington

- All others are *reference types*
  - Rectangle, BankAccount, Color, String, etc. Instances of some class
- Created by new

1/10/2003

- · Can have instance variables and methods
- All are special cases of the generic type "Object"

L-21



Collections: Ma	ny kinds	
Common in cor	nputer programs	
Often correspondence	nd to collections of objects in the real world	
A simple collec	tion: ArrayList	
<ul> <li>Sequential, ord</li> </ul>	ered collection	
Part of the java	util package of classes	
Many methods:	add, get, size, isEmpty, (see Sun Java De	ocs)
<ul> <li>import. java.util</li> </ul>	.*; to access	
Casts		
	o specify actual type of object retrieved from n can hold any kind of object)	a collection
	ference types: need to place primitives if we want to store them in a collect	