Overloading

CSE 331
University of Washington

Michael Ernst
Overloading vs. overriding

• **Overloading**: Multiple operations in a class with the same name and different parameters (number or type)
  – To Java, the operations are *unrelated* to one another
  – Convenient to avoid making up different method names
  – Style rule: The specifications should be analogous
    • Otherwise the program is confusing

• **Overriding**: Same name and parameters as an implementation in a supertype
  – Specification in subtype must be equal or stronger

• CSE 143 covers overriding, but not overloading
An operation is part of an ADT’s specification
A method implementation appears in Java source code
A method family is all the implementations with the same signature (name and parameter types) in an inheritance tree

“Method” can mean any of these. Be specific when ambiguity is possible.
Which implementation gets run?

1. **Resolve overloading at compile time**
   - Let R be the compile-time type of the receiver
   - Choose the most specific, applicable, accessible operation in R
     * Accessible operations: Visible (public, private, protected)
     * Applicable operations: Those whose parameter types are supertypes of the argument types
     * Most specific: its parameter types are subtypes of the corresponding parameter types for other applicable ops
       - If no most specific exists, compile-time error
   
   This picks a method family or signature

2. **Resolve overriding at run time** (dynamic dispatch)
   - Run the implementation in the run-time type of the receiver
     * Might be inherited from a superclass