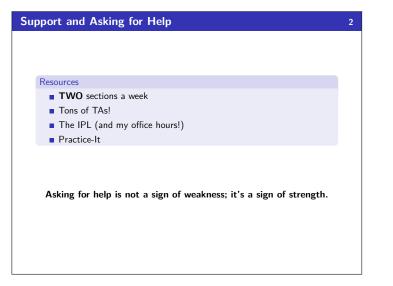


Welcome to CSE 143!



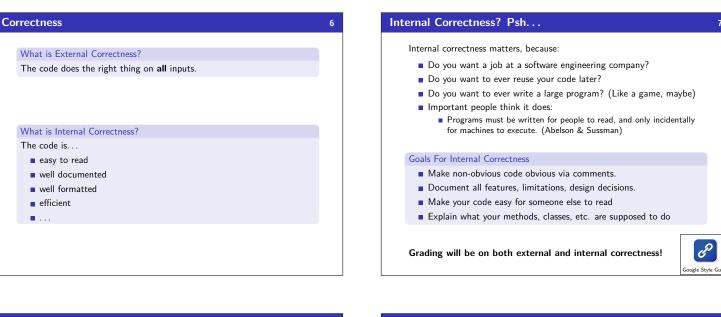
Course Goals 1 CSE 142 vs. CSE 143: The Big Picture In CSE 142, you learned how to use logic, control flow, and decomposition to write programs. In CSE 143, you will learn to solve more complex and larger tasks efficiently. Big Learning Goals • Abstraction (implementation vs. client) • Data Structures (organizing complex data) • Algorithms (standard ways of completing common tasks)



Boring Administrivia 3 Course Website http://cs.uw.edu/143 http://cs.uw.edu/143 4 Section We have two sections a week. Each section has a warm-up; these are completely optional. 5 Grading 9 • Weekly programming projects, 20% midterm, 30% final 9 • Weekly programming projects assigned Fridays, due on Thursdays 9 • 5 "free late days"; -2 points for subsequent days late; up to 3 days late on each hw 1

How Does the Course Fit Together?	4
Descriptive Knowledge ("What is it?")	
Knowing Facts.	
Procedural Knowledge ("How do I do it?")	
Doing Procedures.	
Conceptual Knowledge ("Where does it come from?")	
Understanding HOW things work and WHY they work that way	y.
Lecture provides descriptive knowledge which feeds into	
↓ Section Warm-Ups, which review and feed into	
QuickChecks, which activate knowledge and prepare you for	
The rest of section which turns desc. into proc. knowledge and t	hen
Homework helps you turn desc. and proc. into conceptual	
Exams activate all three types of knowledge	

Program Correctness: Internal & External 5 What does it mean for a program to be "correct"? A program is only correct if it is **internally** correct and **externally** correct. What does this code do? 1 _(__, ___,){___/_<=1?_(__, ___+1, ___):!(___%__)?_(__, ___+1, 0):____ %__==___ / __&&!___?(printf(`%d\t", __/_), _(__, ___+1, 0)):___%__ >1&&____?(__, ___+1, ___):0;}main(){_(100, 0, 0);}



8

Words Exercise

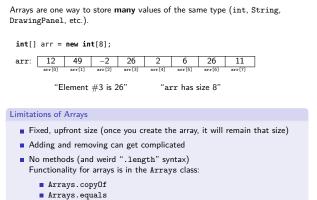
Words Exercise

Write code to read a file and display its words in reverse order.

(Bad) Solution with Arrays

<pre>String[] words = new String[1000];</pre>
int i = 0;
<pre>Scanner inp = new Scanner(new File("words.txt"));</pre>
<pre>while (inp.hasNext()) {</pre>
<pre>String word = inp.next();</pre>
<pre>words[i] = word;</pre>
i++;
}
for (int $j = i - 1; j \ge 0; j$) {
<pre>System.out.println(words[j]);</pre>
}

Review: Arrays



- Arrays.sort
- Arrays.toString

Collections and Lists

Collections

 $\label{eq:collections} \textbf{Collections} \ \textbf{store} \ \textbf{many} \ \textbf{pieces} \ \textbf{of} \ \textbf{data} \ \textbf{of} \ \textbf{the} \ \textbf{same type}.$

10

In Java, collections are in the $\ensuremath{\mathtt{util}}$ package:

import java.util.*;

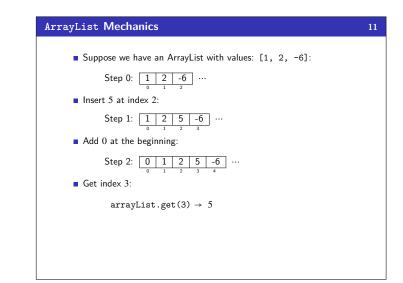
Different collections have different properties: • "Data ordered by indices"

- "Sorted data"
- "Data without duplicates"
- ete
- etc.

Lists

A list is a collection of elements ordered by a 0-based index.

- It supports add/remove from anywhere!
- The size isn't fixed!
- There are multiple implementations; first, ArrayList

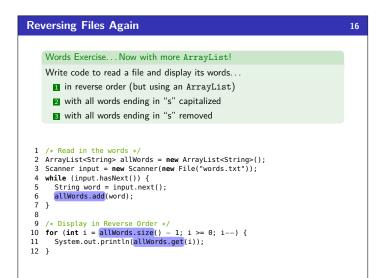


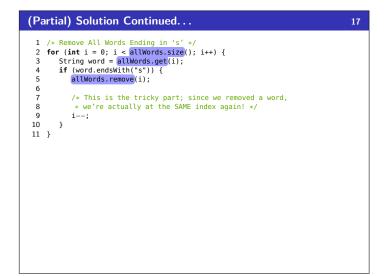
ArrayList Reference			12
	add(val)	Appends val to the end of the list	
	add(idx, val)	Puts val at index idx; all elements at indices idx and larger get shifted forward	
	get(idx)	Returns the value at index idx	
	<pre>set(idx, val)</pre>	Replaces the value at index idx with val	
	remove(idx)	Removes and returns the value at index idx; all elements at higher indices get shifted backward	
	clear()	Removes all elements from the list	
	size()	Returns the number of elements in the list	
	indexOf(val)	Returns the smallest index such that get(idx).equals(val), or -1 if there is no such index	
	toString()	Returns a string representation of the list such as [3, 42, -7, 15]	
			e st Reference

Generics			13
Recall that we can create arrays of different types:			
	<pre>{1, 2, 5, 2} (new int[4])</pre>	{"hi", "banana"} (new String[2])	
	e array initializations specify on for ArrayList's should t	the type of the elements, the oo:	
(ne		["hi", "banana"] (new ArrayList <string>)</string>	
	st is a generic class which ! Java knows the type by w	means that it can handle any type hat you put in <>:	
	ArrayList <mark><string></string></mark> arrayList	t = new ArrayList <mark><string></string></mark> ();	

ArrayList Demo		14
<pre>String[] arr = new String[5]; arr[0] = "hi"; arr[1] = "bye"; String s = arr[0]; for (int i=0; i < arr.length; i++) { if (names[i].contains("b")) {} }</pre>	<pre>→ ArrayList<string> list = new ArrayList<string: → list.add("hi"); → String s = List.get(0); → for (int i = 0; i < List.size(); i++) { → if (List.get(i).contains("b")) {} → }</string: </string></pre>	>();
Note that these two piec	es of code have different loop bounds:	
arr.length == 5	<pre>list.size() == 2</pre>	

ArrayList can be a Parameter or a Return Value	15
ArrayList is just another type (like DrawingPanel or String)!	
<pre>1 public void methodName(, ArrayList<type> name,) { } 2 public ArrayList<type> methodName() { }</type></type></pre>	
The following takes in an ArrayList and returns a new list containing only the words that start with x:	
<pre>public ArrayList<string> startingWithX(ArrayList<string> list) { ArrayList<string> newList = new ArrayList<string>(); for (int i=0; i < list.length; i++) { if (list.get(i).startsWith("x")) { newList.add(list.get(i)); } </string></string></string></string></pre>	
7 } 8 return newList; 9 }	





Today's Takeaways!
Understand the course policies
Learn why internal correctness is important (Are you convinced?)
Recall arrays and how they work from CSE 142
Begin being a client of the ArrayList class