CSSE 220 Day 11

Generic types
Measuring Efficiency
Minesweeper start-up

CSSE 220 Day 11

- Grader comments for JUnit and BigRational assignments should be in your repository.
- There seems to be a problem there with JUnit. Should be resolved this afternoon

- Details about Thursday's exam are in Day 10 PowerPoint Slides

Answers to your questions

- BallWorlds
- Exam
- Anything else

Today's agenda

- Generic types in Java.
- Intro to Algorithm analysis
- Meet your Minesweeper partner, produce a UML diagram for Minesweeper.

Generic types and collections

- Before Java 1.5 (still supported, but gives warnings):

```
ArrayList a = new ArrayList();
Integer b = new Integer(7);
a.add(b);
Integer c = (Integer)(a.get(0));
```

Explicit class cast required.

- New version(generic):

<Integer> is a "type argument" to the class definition.

```
ArrayList<Integer> ag = new ArrayList<Integer>();
ag.add(7);              // automatic wrapping of int.
int cg = ag.get(0);     // auto unwrapping of Integer.
```

automatic unboxing: Integer➔int.

No class cast required.

java.lang

Interface Comparable<T>

Type Parameters:

T - the type of objects that this object may be compared to

- Any class that implements Comparable contracts to provide a **compareTo** method.

Method Detail

String is a Comparable class.
If it did not already have a compareTo method, how would you write it?

compareTo

```
int compareTo(T o)
```

Compares this object with the specified object for order. Returns a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object.

- Therefore, we can write generic methods on Comparable objects. For example, in the Arrays class:

static void	sort(Object[] a, int fromIndex, int toIndex) Sorts the specified range of the specified array of objects into ascending order, according to the natural ordering of its elements.

Example of using Arrays.sort

```
import java.util.Arrays;

public class StringSort {

   public static void main(String[] args) {
      String [] toons = {"Mickey", "Minnie", "Donald",
                         "Pluto", "Goofy"};
      Arrays.sort(toons);
      for (String s:toons)
         System.out.println(s);
   }
}
```

Output:

```
Donald
Goofy
Mickey
Minnie
Pluto
```

Measuring program effciency

- What kinds of things should we measure?
- CPU time
- memory used
- disk transfers
- network bandwidth

- Mostly in this course, we focus on the first two, and especially on CPU time.

- How can we measure running time?
- System.currentTimeMillis
- Run Sieve example.
- When do we really care about efficiency?
- Lots more on this after the break

Minesweeper teams and repositories

- I had to make a couple of adjustments, due to people who had to drop the course.
- This included dissolving the team of three in Section 01 and adding a team of three in Section 02.
- Check out your repository

Repsitory name	student 1	student 2	student 3
minesweeper1	allencw	wenzel	
minesweeper2	andersar	cranemd	
minesweeper3	bacaep	brumbams	
minesweeper4	baekj	reillywj	
minesweeper5	bergencb	berrygl	
minesweeper6	blankeaz	burnsjl	
minesweeper7	dunnmp	vanderkl	
minesweeper8	fehribrm	sullivsd	
minesweeper9	junkersa	leroycw	
minesweeper10	williakl	middlemp	
minesweeper11	pientars	pridalmj	
minesweeper12	salisbjm	watersbt	
minesweeper13	bottjd	thomass2	
minesweeper14	gerthwd	pickdp	
minesweeper15	goodrijk	snivelee	
minesweeper16	hansenrl	merseljp	
minesweeper17	hilljd	tamal	
minesweeper18	iversopn	bussinjr	
minesweeper19	jennemj	buetowbp	
minesweeper20	kotsybja	wisejl	
minesweeper21	kriesbsd	yimah	
minesweeper22	nowickpj	watersdc	kaiserja
minesweeper23	skaggskd	thiememd	