CSSE 220 Day 8

Arrays, ArrayLists,
Wrapper Classes, Auto-boxing,
Enhanced for loop, Array Copying

Check out *ArraysListsAnd2D* from SVN

Questions?

- Reading
- Pascal's Triangle Assignment
- Anything else?

Exam Coming!

- Thursday, Jan 8 (See schedule page!)
- Topics from *Big Java* Ch. 1–9
- Will include:
  - A paper part (40%)
    - logic, short answer, fill-in-the-blank,
    - Code to read and trace
    - Small code snippets to write
  - A programming part (60%)
  - a few small programs, unit tests provided
- Review in-class Wednesday, Jan 7
  - Bring questions
  - I won't anything prepared but am happy to cover whatever you want, including working examples

Q1

Allowed exam resources

- Written part: One two-sided sheet of paper that can be read without special magnifying devices
- Computer part
  - Textbook
  - Java API documentation
  - Code that is on your computer or in your SVN repository before the exam begins.
  - All of my on-line course materials
  - No Google searches, IM, email, etc.
- (Both parts)No headphones

Array Types

- Syntax: `ElementType[] name`
  - Note different bracket placement than in C
- Examples:
  - A variable: `double[] averages;`
  - Parameters: `public int max(int[] values) {…}`
  - A field: `private Investment[] mutualFunds;`

Allocating Arrays

- Syntax: **new** *ElementType*[*length*]
- Creates space to hold values
- Sets values to defaults
  - **0** for number types
  - **false** for boolean type
  - **null** for object types
- Examples:
  - `double[] polls = new double[50];`
  - `int[] elecVotes = new int[50];`
  -
```
boolean [] prime = {false, false, true,true,
                    false, true, false, true};
```

Don't forget this step!

Q2

Reading and Writing Array Elements

- Reading:
  - `double exp = polls[42] * elecVotes[42];`

Reads the element with index 42.

Sets the value in slot 37.

- Writing:
  - `elecVotes[37] = 11;`

- Index numbers run from 0 to array length – 1
- Getting array length: `elecVotes.length`

No parens, array length is (like) a field

Q3,4

Arrays: Comparison Shopping

Arrays…	Java	C	Python
have fixed length	yes	yes	no
are initialized to default values	yes	no	?
knowtheir own length	yes	no	yes
trying to access "out of bounds" element stops the program before worse things happen	yes	no	yes

Live Coding

- Enhance the Fraction class
  Arrays of Fractions

What if we don't know how many elements there will be?

- ArrayLists to the rescue
- Example:

Element type

  - `ArrayList<Fraction> fracs = new ArrayList<Fraction>();`

Variable type

Constructs new, empty list

Adds new element to end of list

```
fracs.add(new Fraction(2,3));
```

- **`ArrayList`** is a *generic class*
  - Type in <brackets> is called a *type parameter*

Q5,6

ArrayList Gotchas

- Type parameter can't be a primitive type
  - Not: `ArrayList<int> runs;`
  - But: `ArrayList<Integer> runs;`
- Use **`get`** method to read elements
  - Not: `runs[12]`
  - But: `runs.get(12)`
- Use **`size()`** not **`length`**
  - Not: `runs.length`
  - But: `runs.size()`

Lots of Ways to Add to List

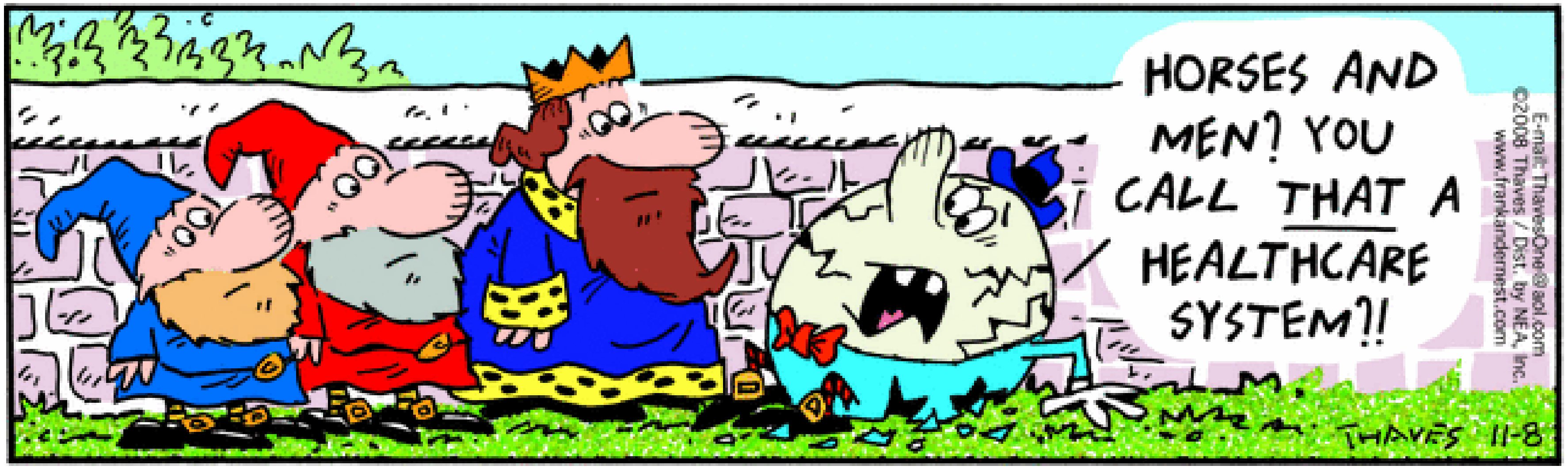
- Add to end:
  - `victories.add(new WorldSeries(2008));`
- Overwrite existing element:
  - `victories.set(0,new WorldSeries(1907));`
- Insert in the middle:
  - `victories.add(1, new WorldSeries(1908));`
  - Pushes elements at indexes 2 and higher up one

- Can also remove from a specific position in the list:
  - `victories.remove(victories.size()  - 1)`

Live Coding

- ArrayLists of Fractions

Cartoon of the Day


HORSES AND MEN? YOU CALL THAT A HEALTHCARE SYSTEM?!
E-mail: ThavesOne@aol.com
©2008 Thaves / Dist. by NEA, Inc.
www.frankandernest.com
THAVES 11-8

So, what's the deal with primitive types?

- Problem:
  - ArrayLists only hold objects
  - Primitive types aren't objects
- Solution:
  - *Wrapper classes*—instances are used to "turn" primitive types into objects
  - Primitive value is stored in a field inside the object

Primitive	Wrapper
byte	Byte
boolean	Boolean
char	Character
double	Double
float	Float
int	Integer
long	Long
short	Short

Q7

Auto-boxing Makes Wrappers Easy

- Auto-boxing: automatically enclosing a primitive type in a wrapper object when needed
- Example:
  - You write: `Integer m = 6;`
  - Java does: `Integer m = new Integer(6);`

  - You write: `Integer ans= m * 7;`
  - Java does:

```
int temp = m.intValue() * 7;
Integer ans = new Integer(temp);
```

Auto-boxing Lets Us Use ArrayLists with Primitive Types

- Just have to remember to use wrapper class for list element type

- Example:

  -
    ```
    ArrayList<Integer> runs =
            new ArrayList<Integer>();

    runs.add(9); // 9 is auto-boxed
    ```

  -
    ```
    int r = runs.get(0); // result is unboxed
    ```

Enhanced For Loop and Arrays

- Old school

```
double scores[] = …
double sum = 0.0;
for (int i=0; i < scores.length; i++) {
    sum += scores[i];
}
```

- New, whiz-bang, enhanced for loop

```
double scores[] = …
double sum = 0.0;
for (double sc : scores) {
    sum += sc;
}
```

Say “in”

- No index variable
- Gives a name (**sc** here) to each element

Enhanced For and ArrayLists

-
  ```
  ArrayList<State> states = …
  int total = 0;
  for (State st : states) {
     total += st.getElectoralVotes();
  }
  ```
- Enhanced **for** also works with arrays

Q8

Copying Arrays

- Assignment uses reference values:

  -
    ```
    double[] data = new double[4];
    for (int i=0; i < data.length; i++) {
          data[i] = i * i;
    }
    double[] pieces = data;
    ```
- Can copy whole arrays:

  -
    ```
    double[] pizzas = (double []) data.clone();
    ```

All objects have a clone() method. Its return type is Object, so we have to cast it.

Q10-13

Copying Part of an Array

- Use built-in function:
  - `System.arraycopy(fromArray,fromStart,`
    `                        toArray,toStart,count);`
- Copies
  - **count** values from **fromArray**,
  - beginning at index **fromStart**,
  - copying into array **toArray**,
  - beginning at index **toStart**