

CSSE 220 Day 9

Two-dimensional arrays,
Copying arrays,
Software Engineering Techniques

Check out *TwoDArrays* from SVN

Questions?

```
public class TicTacToe {
    private final int rows;
    private final int columns;
    private String[][] board;
```

Two-dimensional arrays

```
/**
 * Constructs a 3x3 TicTacToe board with all squares blank.
 */
```

```
public TicTacToe() {
    this.rows = 3;
    this.columns = 3;
```

What is the value of `this.board[1][2]` immediately after this statement executes?

```
this.board = new String[this.rows][this.columns];
```

```
for (int r = 0; r < this.rows; r++) {
```

Could have used:
`this.board.length`

```
    for (int c = 0; c < this.columns; c++) {
```

```
        this.board[r][c] = " ";
```

Could have used:
`this.board[r].length`

```
    }
```

```
}
```

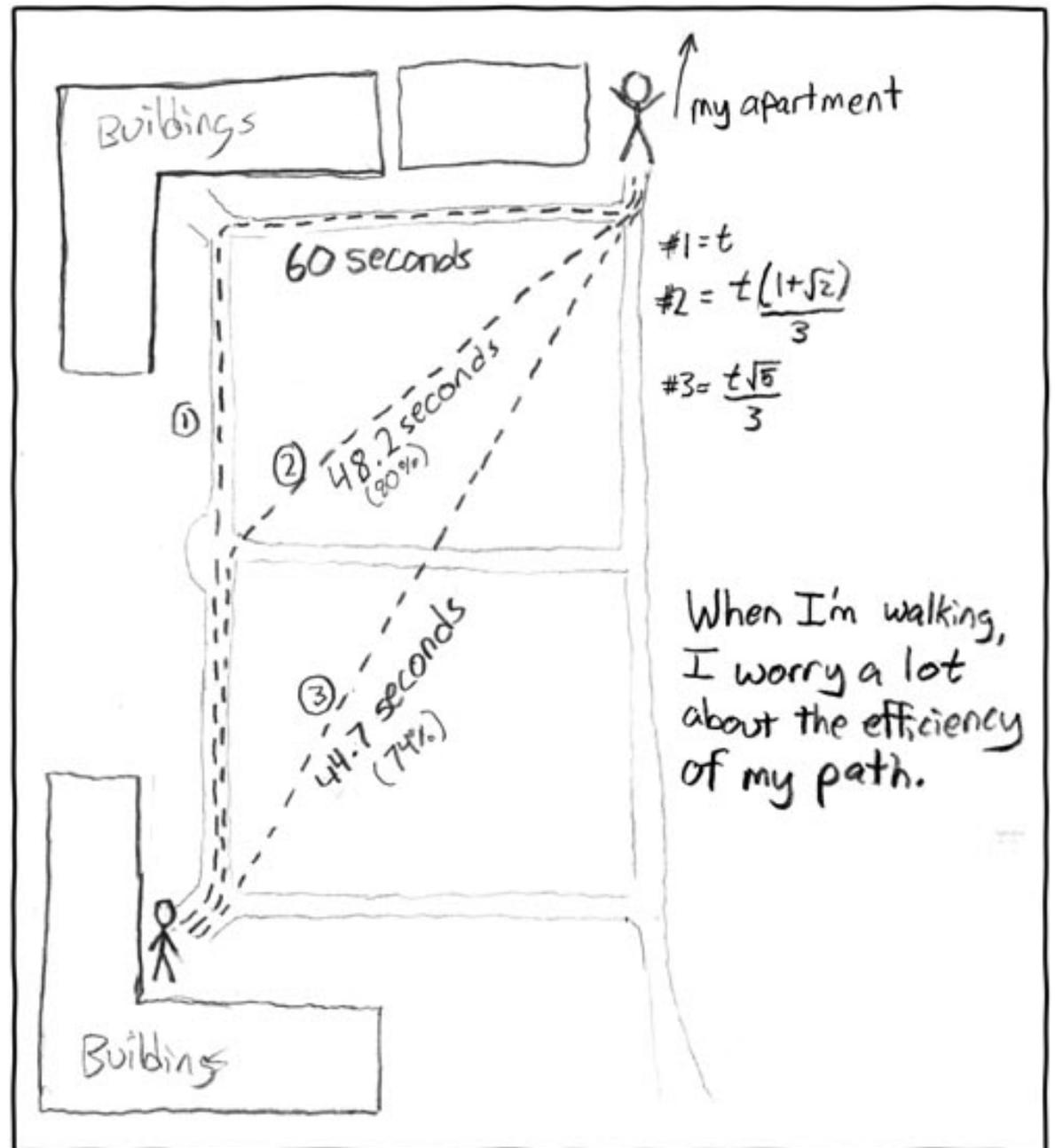
Note the (very common) pattern: loop-through-rows, for each row loop-through columns

Q1

Exercise

- Complete the TODO items in TicTacToe and TicTacToeTest
- » They're numbered; do 'em in order.

Interlude:



<http://xkcd.com/85/>

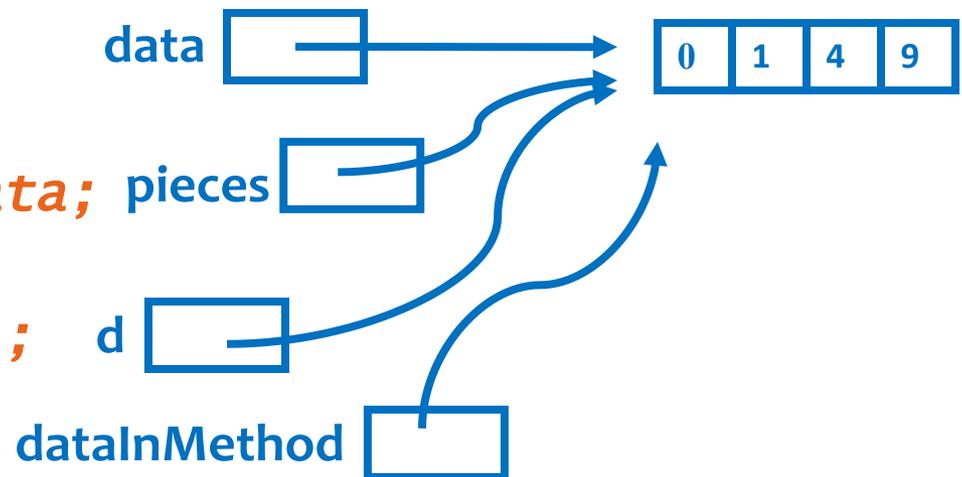
Copying Arrays – assignment

▶ Assignment uses *reference* values:

```
◦ double[] data = new double[4];  
for (int i = 0; i < data.length; i++) {  
    data[i] = i * i;  
}
```

```
◦ double[] pieces = data;
```

```
◦ foo.someMethod(data);
```



This makes the field a reference to (NOT a copy of) a list that exists elsewhere in the code. Think carefully about whether you want this or a clone (copy).

```
public void someMethod(double[] d) {  
    this.dataInMethod = d;  
    ...  
}
```

Copying Arrays – many ways

- ▶ You can copy an array in any of several ways:
 1. Write an explicit loop, copying the elements one by one

2. Use the *clone* method that all arrays have

```
newArray = oldArray.clone();
```

3. Use the *System.arraycopy* method:

```
System.arraycopy(oldArray, 0, newArray, 0,  
oldArray.length);
```

4. Use the *Arrays.copyOf* method:

```
newArray = Arrays.copyOf(  
oldArray, oldArray.length);
```

Starting position in *oldArray*

Starting position in *newArray*

Number of elements to copy

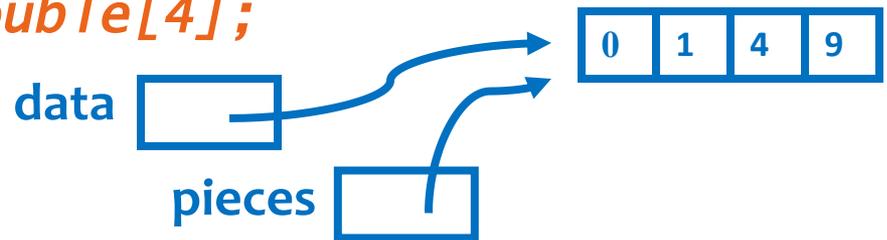
The key point is that all of these except possibly the first make *shallow copies* – see next slide

Copying Arrays – Shallow copies

- ▶ Can copy whole arrays in several ways:

- `double[] data = new double[4];`

- `...`
`pieces = data;`



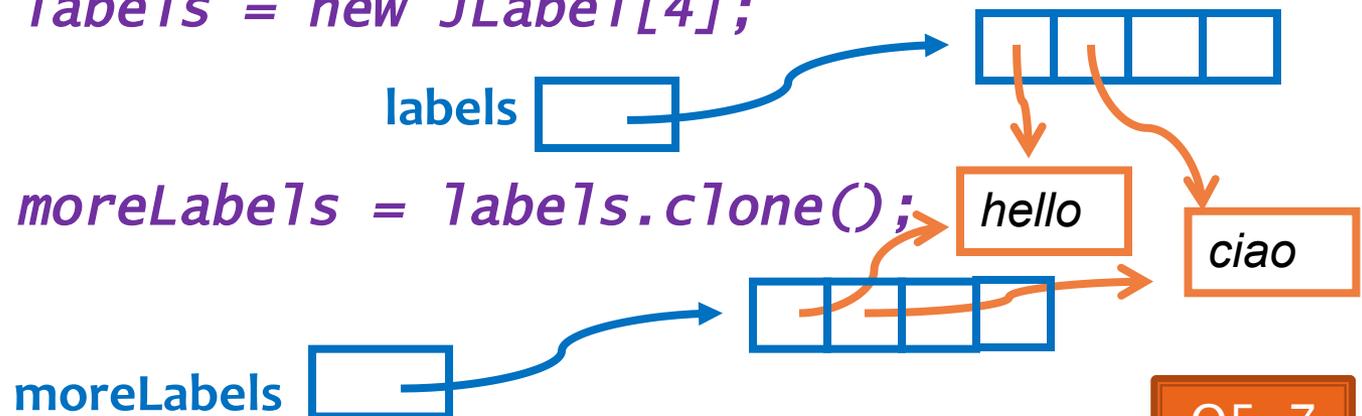
- `double[] pizzas = data.clone();`



- `JLabel[] labels = new JLabel[4];`

- `...`

- `JLabel[] moreLabels = labels.clone();`



Quality Tip – “Avoid parallel arrays”

- ▶ Consider an ElectionSimulator:
 - ▶ Instead of storing:
 - *ArrayList<String> stateNames;*
 - *ArrayList<Integer> electoralVotes;*
 - *ArrayList<Double> percentOfVotersWhoPlanToVoteForA;*
 - *ArrayList<Double> percentOfVotersWhoPlanToVoteForB;*
 - ▶ We used:
 - *ArrayList<State> states;*
and put the 4 pieces of data inside a State object
- ▶ Why bother?

Pick the Right Data Structure

- ▶ Array or ArrayList, that is the question
- ▶ General rule: use ArrayList
 - Less error-prone because it grows as needed
 - More powerful because it has methods
- ▶ Exceptions:
 - Lots of primitive data in time-critical code
 - Two (or more) dimensional arrays

Software Engineering Techniques

- ▶ Regression testing
- ▶ Pair programming
- ▶ Team version control

Regression Testing

- ▶ Keep and run old test cases
- ▶ Create test cases for new bugs
 - Like antibodies, to keep a bug from coming back
- ▶ Remember:
 - You can right-click the project in Eclipse to run all the unit tests

Pair Programming Video

- ▶ Let's watch the video together

Pair Programming

Becoming a
common interview
technique!

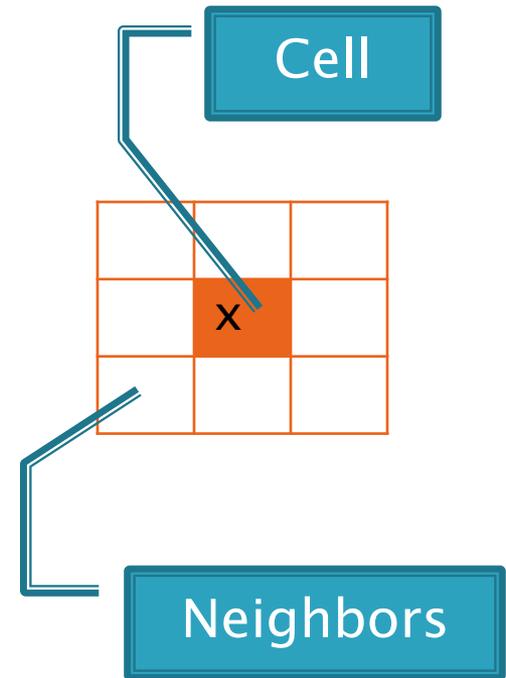
- ▶ Working in pairs on a single computer
 - One person, the *driver*, uses the keyboard
 - The other person, the *navigator*, watches, thinks, and takes notes
- ▶ For hard (or new) problems, this technique
 - Reduces number of errors
 - Saves time in the long run
- ▶ Works best when partners have similar skill level
 - If not, then student with most experience should navigate, while the other student drives.

Team Version Control

- ▶ **Always:**
 - Update before working
 - Update again before committing
 - Commit often and with good messages
- ▶ **Communicate** with teammates so you don't edit the same code simultaneously
 - Pair programming eliminates this issue

Game of Life

1. A new cell is born on an empty square if it has exactly 3 neighbor cells
2. A cell dies of overcrowding if it is surrounded by 4 or more neighbor cells
3. A cell dies of loneliness if it has just 0 or 1 neighbor cells



Work Time

- ▶ Work with your partner on the GameOfLife project
 - Get help as needed
 - The TODOs are numbered – do them in the indicated order.
 - *Follow the practices of pair programming!*
- ▶ ***Don't do any of the work without your partner!***
- ▶ Good exam prep.