

CSCI 136
Data Structures &
Advanced Programming

Fall 2017

Instructors

Bill Jannen & Bill Lenhart

Administrative Details

- Class roster: Who's here?
 - And who's trying to get in?
- Handout: Class syllabus
- Lecture location: SSL 030B
- Lab: Wed 12-2 or 2-4
- Lab location: TCL 217a (Lenhart) & 216 (Jannen)
- Lab entry code: XXXXXXXX (ask us if you forgot!)
- Course Webpage:

<http://cs.williams.edu/~cs136/index.html>

Today's Outline

- Course Preview
- Course Bureaucracy
- Java refresher—Part I

Why Take CSI 36?

- To learn about:
 - Data Structures
 - Effective ways to store and manipulate data
 - Advanced Programming
 - Use structures and techniques to write programs that solve interesting and important problems
 - Basics of Algorithm Analysis
 - Measuring algorithm complexity
 - Determining algorithm correctness

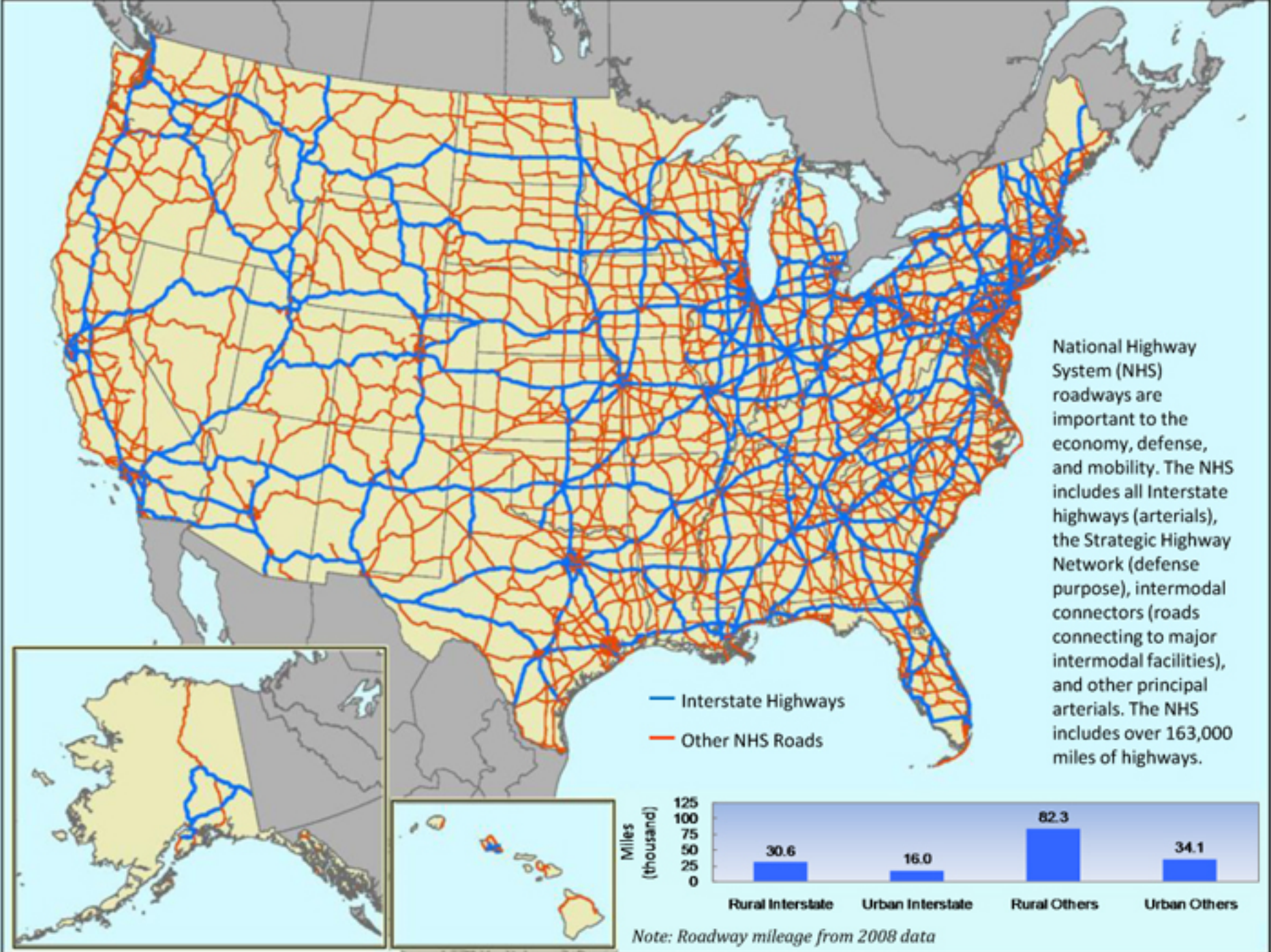
Squad* Goals

- Identify basic data structures
 - list, stack, array, tree, graph, hash table, and more
- Implement these structures in Java
- Learn how to evaluate and visualize data structures
 - Linked lists and arrays both represent lists of items
 - Different representations of data
 - Different algorithms for manipulating/accessing/storing data
- Learn how to design larger programs that are easier to modify, extend, and debug
- **Have fun!**

Common Themes

1. Identify data for problem
2. Identify questions to answer about data
3. Design data structures and algorithms to answer questions *correctly* and *efficiently* (Note: not all correct solutions are efficient, and vice versa!)
4. Implement solutions that are robust, adaptable, and reusable

Example: Shortest Paths in Networks



Finding Shortest Paths

- The data: road segments
 - Road segment: Source, destination, length (weight)
- The question
 - Given source and destination, compute the shortest path from source
- The algorithm: Dijkstra's Algorithm
- The data structures (spoiler alert!)
 - Graph: holds the road network in some useful form
 - Priority Queue: holds not-yet-inspected edges
 - Also uses: Lists, arrays, stacks, ...
- A quick demo....

Course Outline

- Java review
- Basic structures
 - Lists, vectors, queues, stacks
- Advanced structures
 - Graphs, heaps, trees, dictionaries
- Foundations (throughout semester)
 - Vocabulary
 - Analysis tools
 - Recursion & Induction
 - Methodology

Syllabus Highlights

- How to contact us
 - Bill Lenhart (TPL 304)
 - Office hours: Tues & Thurs 1:30-3:30pm and by appointment
 - <mailto:wlenhart@williams.edu>
 - Bill Jannen (TCL 306)
 - Office hours:
 - <mailto:jannen@cs.williams.edu>
- Textbook
 - Java Structures: Data Structures in Java for the Principled Programmer, $\sqrt{7}$ Edition (by Duane Bailey)
 - Take one: You're already paying for it!
- Weekly labs, problem sets, mid-term & final exams....

Honor Code and Ethics

- College Honor Code and Computer Ethics guidelines can be found here:
 - <https://sites.williams.edu/honor-system/>
 - <https://oit.williams.edu/policies/ethics/>
- You should also know the CS Department computer usage policy.
 - <https://csci.williams.edu/the-cs-honor-code-and-computer-usage-policy/>
 - If you are not familiar with these items, please review them.
- We take these things very seriously...

Your Responsibilities

- Come to lab and lecture on time
- Read assigned material before class and lab
 - Bring textbook to lab (or be prepared to use PDF)
 - Bring paper/pen(cil) to lab for brain-storming, ...
- **Come to lab prepared**
 - Bring design docs for program
 - I Prof + I TA == help for you: take advantage of this
- Do NOT accept prolonged confusion! Ask questions
- Your work should be your own. Unsure? Ask!
- Participate

Accounts and Passwords

- Before the first lab
 - Check your CS Mac Lab account password
 - If you don't have an account, see Mary Bailey
 - If you forgot your password, see Mary Bailey
- Mary manages our systems. She will be available
 - Today (Sept. 8): 10:00–11:30 am
 - Mon. (Sept. 11): 10:00–11:30 am & 3:00–4:30 pm
 - Tues. (Sept. 12): 10: 10:00–11:30 am & 3:00–4:30 pm
 - Wed. (Sept. 13): 10: 10:00–11:30 am
 - Her office is in the 3rd floor CS lab (TCL 312)
- Get this sorted out before lab on Wednesday!

Why Java?

- There are lots of programming languages...
 - C, Pascal, C++, Java, C#, Python
- Java was designed in 1990s to support Internet programming
- Why Java?
 - It's easier (than predecessors like C++) to write correct programs
 - Object-oriented – good for large systems
 - Good support for abstraction, extension, modularization
 - Automatically handles low-level memory management
 - Very portable

Why Not BlueJ?

- Learn to use Unix
 - Command-line tools
 - Emacs standard unix-based editor
- Emphasis will move from user interface programming to data structuring and efficient algorithm design
- Take advantage of opportunity to become Unix-savvy!

Java Review (Crash Course)

Java

- Variable types
 - Primitive: int, double, boolean, ...
 - Object (class-based): String (special), Point, JButton, ...
 - Arrays
- Statements
 - `int x = 3; // declare & initialize x`
 - `x = x + 2;`
 - `if (x > 3) { ... } else { ... }`
 - `while (x < 2) { ... }`
 - `for (int i = 0; i < x; i++) { ... }`
- Comments
 - `//this is a comment`
 - `/* so is this */`

Primitive Types

- Provide numeric, character, and logical values
 - 11, -23, 4.21, 'c', false
- Can be associated with a name (*variable*)
- Variables must be declared before use

```
int age;           // A simple integer value
float speed;      // A number with a 'decimal' part
char grade;      // A single character
bool loggedIn;   // Either true or false
```

- Variables can be initialized when declared

```
int age = 21;
float speed = 47.25;
char grade = "A";
bool loggedIn = true;
```

Array Types

- Holds a collection of values of some type
- Can be of any type

```
int[] ages;           // An array of integers
float[] speeds;       // An array of floats
char[] grades;        // An array of characters
bool[] loggedIn;     // Either true or false
```

- Arrays can be initialized when declared

```
int[] ages = { 21, 20, 19, 19, 20 };
float[] speeds = { 47.25, 3.4, -2.13, 0.0 };
char[] grades = { "A", 'B', "c", "C" };
bool[] loggedIn = { true, true, false, true };
```

- Or just created with a standard default value

```
int[] ages = new int[15]; // array of 15 0s
```

Simple Sample Programs

- Hello.java
 - Write a program that prints “Hello” to the terminal.
 - Now let’s run it.
- Of Note:
 - `public static void main(String[] args){...}`
 - `System.out` is of type `PrintStream`
 - `javac` and `java` commands

Sample Programs

- Sum0-5.java
 - Programs that adds two integers
- Of Note:
 - System.in is of type ReadStream
 - Scanner class provides parsing of text streams (terminal input, files, Strings, etc)
 - args[] is passed to main from the OS environment
 - args[] contains command-line arguments held as Strings
 - Integer.valueOf(...) converts String to int
 - Static values/methods: in, out, valueOf, main