CSCI 136
Data Structures & Advanced Programming

Spring 2018

Instructors

Bill Jannen & Jon Park

Today's Outline

- Course Preview
- Course Bureaucracy
- Crash Course in Java – Part 1

Why Take CS136?

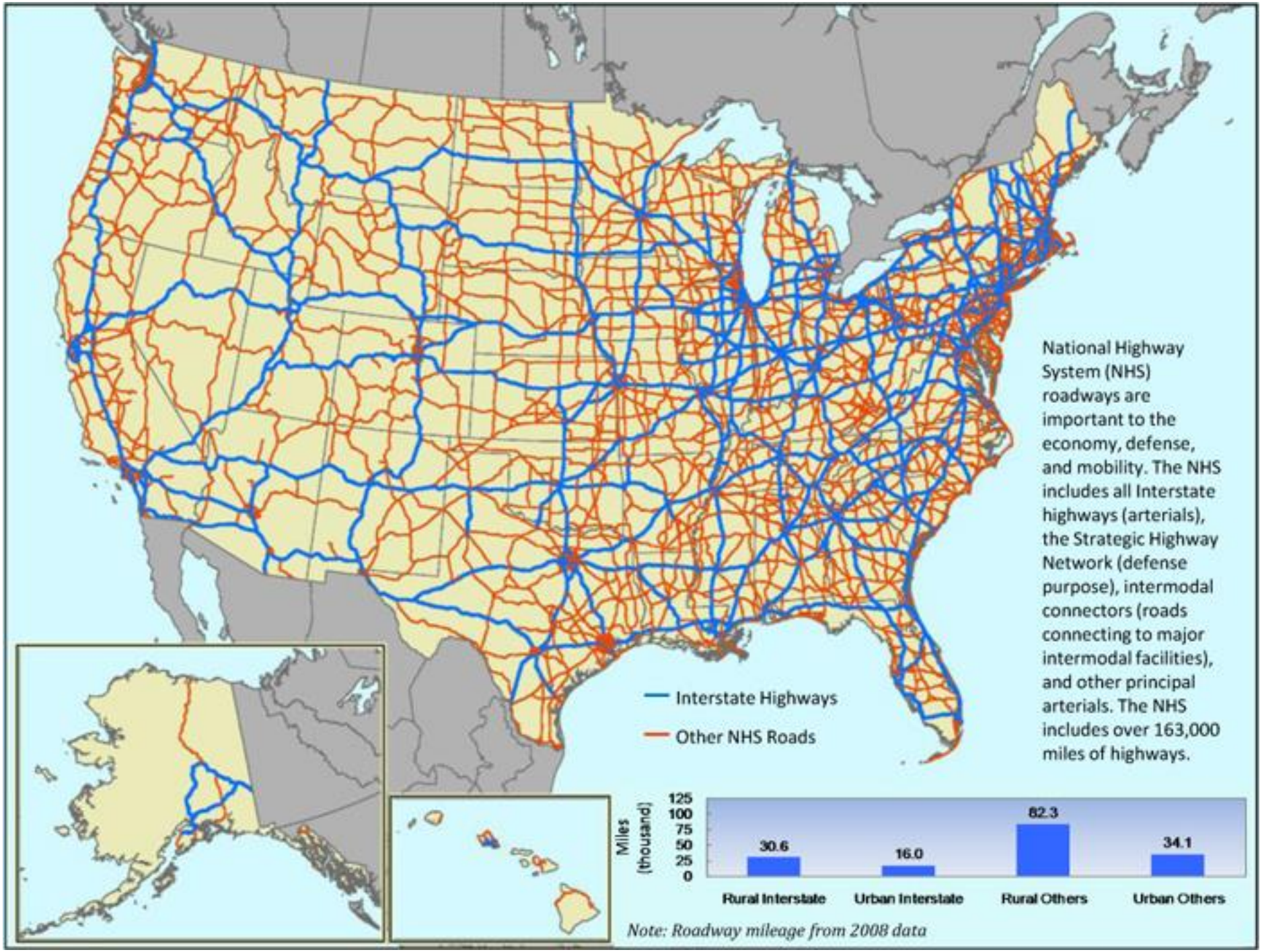
- To learn about:
  - Data Structures
  - Advanced Programming
  - Basics of Algorithm Analysis

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*
4. Implement solutions that are robust, adaptable, and reusable


National Highway System (NHS) roadways are important to the economy, defense, and mobility. The NHS includes all Interstate highways (arterials), the Strategic Highway Network (defense purpose), intermodal connectors (roads connecting to major intermodal facilities), and other principal arterials. The NHS includes over 163,000 miles of highways.
Interstate Highways
Other NHS Roads
Miles (thousand)
125
100
75
50
25
0
30.6
16.0
82.3
34.1
Rural Interstate
Urban Interstate
Rural Others
Urban Others
Note: Roadway mileage from 2008 data

Finding Shortest Paths

1. Identify data for problem
   - Road segment: Source, destination, length (weight)
2. Identify questions to answer about data
   - compute the shortest path
3. Design data structures and algorithms to answer questions *correctly* and *efficiently*
   - Graph: holds the road network in some useful form
   - Also uses: Lists, arrays, stacks, priority queue, ...
   - Dijkstra's Algorithm
4. Implement solutions that are robust, adaptable, and reusable

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

Administrative Details

- Course Webpage:

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

Lab entry code: x-x-x-x-x-x (memorize now!)

[TAP:QEGWV] Java Savvy?

- What is your level of proficiency in Java?
  A. Tabula rasa (I am in a “blank state”)
  > B. Beginner (I have played around with it)
  > C. Proficient (I have had at least 1 on/off-line course)
  D. Expert (~~I’m here to teach you, Jon!~~)
  E. Whatever

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…

Unsure? Ask!

Your Responsibilities

- Read assigned material before class and lab
- Be on time (class and lab)
- **Come to lab prepared**
  - Bring design docs for program
  - Bring textbook to lab (or be prepared to use PDF)
  - Bring paper/pen(cil) to lab for brain-storming, …
- Do NOT accept prolonged confusion! Ask questions
  - 1 Prof + 1TA : Help us help you!
  - ~13 students : Help one another! (But, your work should be your own.. Unsure? Ask!)

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. Her office is in the 3rd floor CS lab (TCL 312). She'll be available at:
  - Today (Feb 2): 9:30–11:15am, 1:15-2:30pm
  - Mon. (Feb. 5): 10:00–11:30am & 2:00–4:00pm
  - Tues. (Feb. 6): 9:00-11:00am & 3:00–4:30pm
  - Wed. (Feb 7): 9:00am–11:00 am
- Get this sorted out before lab on Wednesday!

Why Java?

- It's easier (than predecessors like C++) to write correct programs
  - Automatically handles low-level memory management
  - Object-oriented – good for large systems
  - Good support for abstraction, extension, modularization
- Very portable

Why Not BlueJ (or other IDE)?

- Learn to use Unix
  - Command-line tools
  - Emacs standard unix-based editor
- Take advantage of opportunity to become Unix-savvy!

Crash Course in Java

Spring 2018

Instructors

Bill Jannen & Jon Park

Simple Sample Program

- Hello.java
  - program that prints “Hello!” to the terminal.

file name = class name

- write code + save
- compile (javac Hello.java)
- run (java Hello)

Crash Course in Java

- Variables "content word" school nice ... x
  - Operators "function word" is +
  - Expressions "phrase" very nice x+3
  - Statements "sentence" x = x + 3;

Variable Types

- Primitive Types:
  - boolean, char, byte, short, int, long, float, double
- Objects :
  - arrays
    - Holds values of a single type
  - (class-based) Objects
    - Can hold information (fields)
    - Can specify behaviors (methods)

Variables

- Variables must be declared before use

```
int age;
char grade;
bool loggedIn;
int[] grades;
Student x;
```

- Variables can be initialized when declared

```
int age = 21;
char grade = "A";
bool loggedIn = true;
Student x = new Student();
int[] grades = new int[5];
int[] grades = { 21, 20, 19, 19, 20 };
```

[TAP] Array Initialization

```
1. int[] grades;
2. int[] grades = new int[5];
3. int[] grades = { 0, 0, 0, 0, 0 };
```

- Which of the above are equivalent?
  - A. 1, 2, 3
  - B. 1, 2
  - C. 2, 3
  - D. They are all different
  - E. Whatever

Crash Course in Java

- Variables
- Operators
- Expressions
- Statements

(General) Operators

- Unary
  - Arithmetic: +, -, ++, -- (prefix and postfix)
  - Logical: !
- Binary
  - Arithmetic: +, -, *, /, %
  - Relational: ==, !=, <, <=, >, >=
  - Logical: &&, ||
  - Assignment: =, +=, -=, *=, /=, %=
- Ternary
  - booleanCondition ? value1 : value2