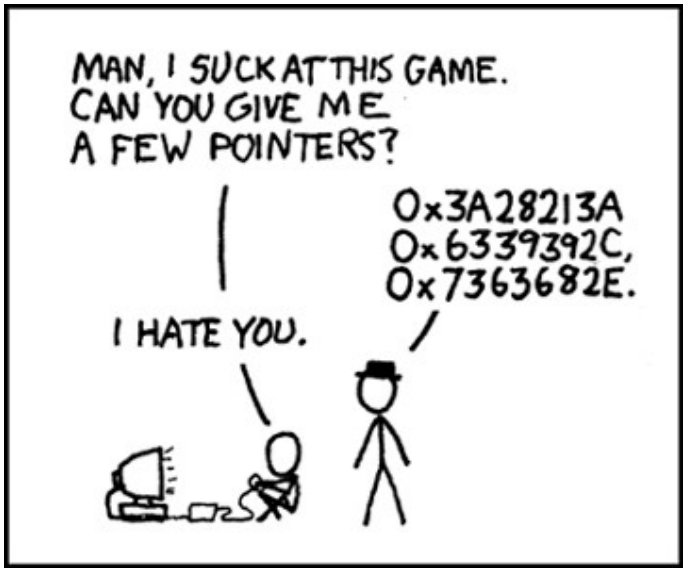
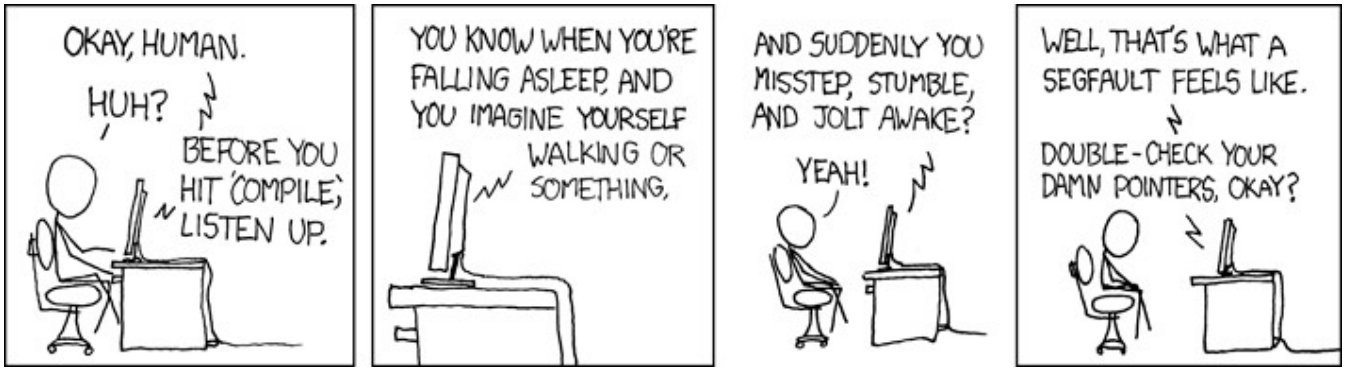
CS 240
Fall 2015
Section 004

Alvin Chao, Professor





Today

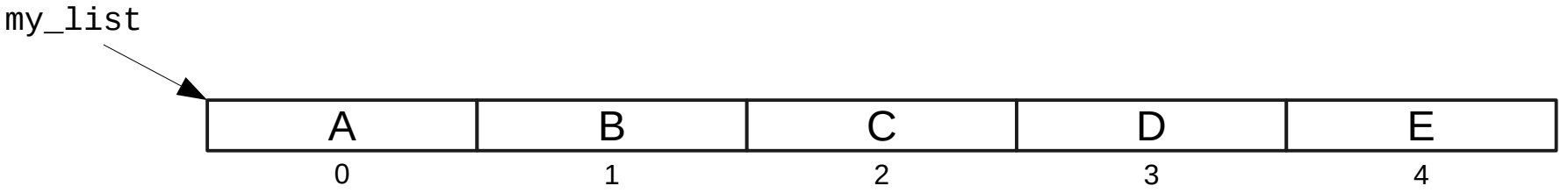
- Course overview – Data Structures / Algorithms
- Course policies
- The C language

Motivation

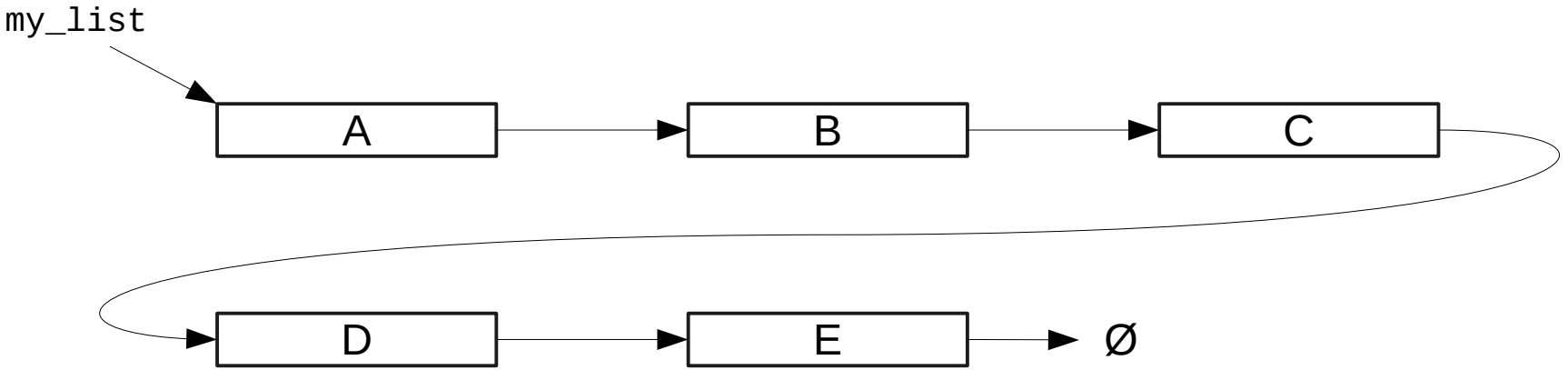
- Computers are digital
  - Data is stored in binary format (1's and 0's)
  - Assembly programming uses this directly
  - This is tedious and error-prone!
- High-level languages (like Java) help
  - Variables (where are the bits located?)
  - Data types (what do the bits mean?)
  - But they don't inherently solve all problems

Basic Example

- Arrays vs. Linked Lists – Arrays expensive insert/delete and fixed size.



vs.



Overview

- Data structures
  - Abstractions for organizing data on a computer
    - “How is your data organized?”
  - Generally, the goal is to be as efficient as possible
  - Topics: arrays, linked lists, stacks, queues, trees, graphs, maps, hash tables

Overview

- Algorithms
  - Step-by-step procedure to solve a problem
    - “How exactly do I do ____?”
  - Generally, the goal is to be as efficient as possible
  - Topics: recursion, asymptotic analysis, Big-O notation, searching, sorting

Overview

- Lots of crossover
  - Most data structures are designed to help solve particular types of problems
  - Many algorithms take advantage of particular data structures

Game: Spit-Not-So

SPIT
NOT
SO
FAT
FOP
AS
IF
IN
PAN

Rules:

- Players take turns claiming words (one circles, the other crosses out)
- The first player to claim three words containing the same letter wins

Can you devise a strategy that always wins?

Game: Spit-Not-So

- Core insight: the *organization* of our information can make a **HUGE** difference in how easy it is to perform tasks using that data!

Key Concepts

- Abstract data types
  - Generic classes of data structures
  - Often provided with a language's standard library
  - Two goals: efficiency and reusability
- Algorithm analysis
  - Measuring efficiency (what metrics to use?)
  - Comparing algorithms (which is faster?)
  - Empirical (test it!) and analytical (math!)

Goals

- Describe, implement, and analyze common data structures and algorithms
- Choose appropriate structures and algorithms to solve real-world problems
- Secondary goal: become familiar with a programming language that is not Java

Syllabus & Course Website

- This is the main course website:

  `http://educ.jmu.edu/~chaoaj/cs240fall2015`

- Lots of useful stuff:
  - Syllabus (Read it! Especially the parts marked in **bold**)
  - Calendar (Will be updated regularly)
  - Assignments
  - Resources
- Bookmark it and check back often

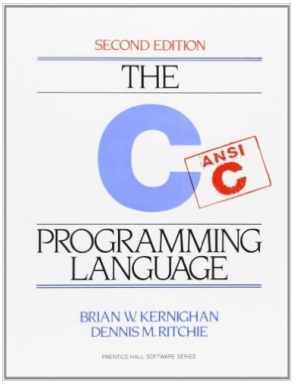
Online Systems

- Canvas
  - Grades and online quizzes
  - Project submissions
  - Private files (i.e., project solutions)
- Piazza (accessed via Canvas)
  - Q&A and discussions
- Make sure you can access all of these!

Textbook(s)

- Required textbook: “Open Data Structures”
  - Open-source textbook by Pat Morin
  - Available as a PDF from `http://opendatastructures.org/`
  - Available as a coursepack from JMU bookstore
  - Available as a paperback online (see link in syllabus)

- Recommended book: "The C Programming Language"
  - Brian Kernighan and **Dennis Ritchie** (creator of C)
  - Available on Safari Books through the library



Course Policies

- Reading quizzes are due by 9am the day they are to be covered
  - One-time exception for today's quiz (due tomorrow)
- Class attendance is **highly** recommended
  - If the class periods are not worth attending, tell me so that I can make them better!
- Slides will be posted on the website
  - Don't waste time writing down stuff from the slides
- Please silence your cell phones during class
  - Be respectful with laptop and tablet usage

Course Policies

- Submit programming projects as specified in the project description
  - No thumb drives, CDs, or emails (unless requested)
- Project grading will be based on automated test results
  - ***You will not have access to all grading tests before the deadline***
  - ***Some portion of the grade (usually 10-20%) will be granted for style and documentation based on a manual inspection of the code***
- Late submissions up to 72 hours will receive a 10% penalty per 24 hr period

Course Policies

- The JMU Honor Code applies on ALL assignments
  - We **will** use software to detect plagiarism
  - Violations may be sent to the honor council
  - See relevant section in the syllabus
- All submitted code must be YOUR work entirely
  - You may work in groups to discuss assignments (in fact, I encourage this), but do NOT share code!

Course Grades

Homework Assignments	30%
Programming Projects	25%
Midterm Exams	30%
Final Exam	15%

Course Policies

- Exams will be held in ISAT 248
  - Final exam times are on the website
- If you ask for a re-grade, I may re-grade the entire assignment
  - This applies to homework and projects, too
- If you have to miss a due date or exam because of an excused absence, let me know ASAP

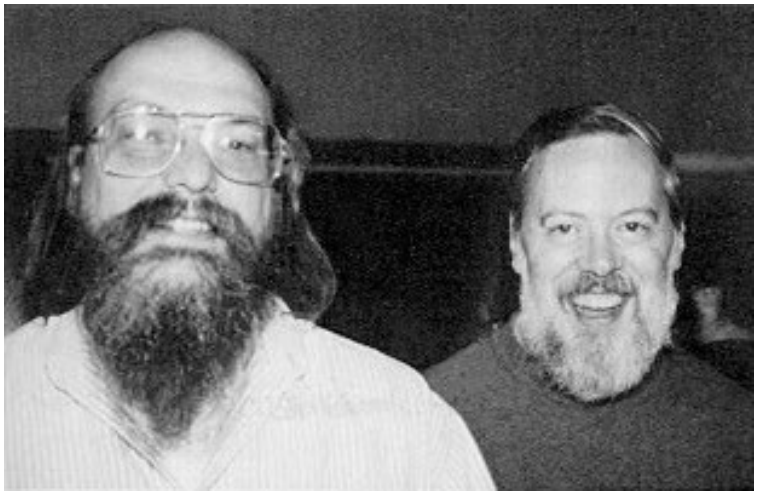
Involvement Opportunities

- JMU Unix Users' Group
  - Club of Unix/GNU/Linux fans who gather for fun and tech talk
  - Tutorial series throughout semester
    - Linux InstallFest on Wed, Sept. 2 at 7pm in ISAT/CS 259 (nTelos Room) – repeated on Tue, Sept. 8 (same time and place)
    - HIGHLY RECOMMENDED for CS 240 students
- ACM Competitive Programming Club
  - Club of students who enjoy programming and problem solving
  - Meetings and contests throughout semester
    - First meeting on Fri, Sept. 4 at 2:30pm in ISAT/CS 143
    - Repeats every Friday (same time and place)
    - Can be taken as a course for one credit if you wish (register for CS 280)
    - HIGHLY RECOMMENDED for CS 240 students

Questions?

The C Language

- Invented by Dennis Ritchie in the early 1970s at Bell Labs for use during the development of the Unix operating system
  - Based on "B" by Ken Thompson
  - Kernighan interview: `https://www.youtube.com/watch?v=de2Hsvxaf8M`
- Like Java, it is a general purpose, imperative, structured programming language with static scoping and typing
  - Unlike Java, it is NOT object-oriented or automatically garbage-collected
- Usually compiled to native machine code
- Has been in use for nearly fifty years
- Ubiquitous
- Pointers!



Ken Thompson (left)
Dennis Ritchie (right)

C Example

```c
#include <stdio.h>

int main()
{
    int value = 1;
    for (int x = 0; x <= 10; x++) {
        printf("2 ^ %d = %d\n", x, value);
        value *= 2;    // calculate next power of 2
    }
    return 0;
}
```

Colors:
keywords
local variables
string literal
comment

Why C?

- Core language is very simple
  - Small number of distinct language constructs
- Language is standardized and compilers are highly mature
  - We will be using the C99 standard this semester
  - Mature compilers => fast programs!
- Still widely used (especially in systems and embedded software)
  - #2 in 2015 TIOBE rankings (14.7%)
  - There is a very high likelihood that you will run into C code at some point in your career
- Aligns well with CS 240 goals
  - Manual memory management model (no hidden deallocations)
  - Statically typed (compiler helps you w/ debugging)
  - Clear distinction between data and pointers

C Style Guides

- There are many different style guides for C
  - https://en.wikipedia.org/wiki/Indent_style
  - My preferred style is the GNU Style – also used by other JMU instructors
  - Dr Lam and others prefer the 1TBS ("the one true brace style," based on K&R)
  - From K&R C book: "Pick a style that suits you, then use it consistently."
  - The key is to make sure that others can easily **read** and **understand** your code
- Keep your code clean
  - If I have trouble reading it, I will deduct points
  - If your code style and/or whitespace is inconsistent, I will deduct points
- Include documentation
  - If I have no clue what you're trying to do, I will deduct points

Learning C

- We will spend a couple of weeks learning C
  - Hybrid lecture/labs and dedicated lab time
  - "Demo" programming project (PA0)
  - C homework assignment (completing certain labs)
  - Major programming project (PA1)
- You should be spending time outside class learning C as well
  - Quickly learning a new language is a very important CS skill
  - There are MANY tutorials available online
  - We will not be able to cover everything about C in the labs
  - I recommend setting aside some time each evening to experiment—30 minutes to an hour

Homework

- Complete course survey by tomorrow (on Canvas)
- Complete first reading quiz by tomorrow (on Canvas)
- If you have a personal computer, install a C compiler and a text editor (see "Resources" page on website)
  - Alternatively, go into ISAT 250 and familiarize yourself with the software on the lab computers
  - Change the code snippet from the earlier slide to calculate Fibonacci numbers rather than powers of two
- Consider attending Linux InstallFest on Wednesday
  - They will help you set up a development environment for this class

Contacting Me

- Questions? Try Piazza!
- Email: `chaoaj`
  - I will attempt to respond as quickly as possible, but do not expect a response in under 24 hours
- Office: ISAT 264 or Massanutten 293
  - Office hours T/H 2:30-3:30
  - Appointments preferred outside office hours

Good luck!

- Have a great semester!