CSE142: Computer Programming I

Come up and meet the staff!

We're not just here for your viewing pleasure.

We can engage in very lifelike conversation.

CSE142 Computer Programming I

Overview

Martin Dickey & Steve Wolfman

Spring Quarter 2001

Based on slides by R. Anderson, H. Perkins, J. Zahorjan, *et al.*

CSE142: Computer Programming I

What is the power of a class this large?

How loud can a class of 250 people be?

How smart can a class of 250 people be?

How insightful can a class of 250 people be?

How tight can a class of 250 people be?

Today's Outline

The World of Computers

The World of Problems

The World of Programming

The Eternal Question: Why Are We Here?

Course Organization and Mechanics

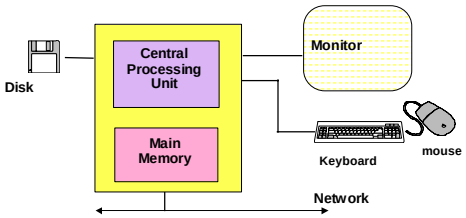
The World of Computers

A computer is a general-purpose machine for manipulating data.

Computers execute instructions in a *deterministic* fashion.

- execute a set sequence of commands
- guided by current state and inputs

The Modern Computer



But in a way, this is a lie…

Why?

What makes computers so great?

Speed:

– What is `53*7`? Raise your hand when you know.

A Pentium III can do that in one billionth of a second!

Size:

– Early computers "filled a room." How big is that?

Modern computers fit everywhere from the palm of your hand to the ruff of a cat's neck!

Precision:

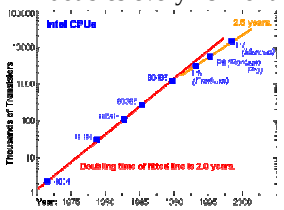
– How often does a computer make a mistake?

Disks: ~one error in one billion bits…
I can't even remember what I had for breakfast.

A-7

Is the revolution over?

"Moore's Law" says # of transistors per chip doubles every 18 months… is it over?



[Graph from George Watson, U. Delaware]

Gordon Moore made the observation now called Moore's Law over 35 years ago!

A-8

World of Computers: Travel Brochure

The modern computer is:

– incredibly small, fast, and efficient

– "deterministic":

always responds the same to the same commands

– capable of a variety of input and output

– connected to the world

– *nothing until a programmer tells it what to be!*

A-9

Today's Outline

The World of Computers
The World of Problems
The World of Programming
The Eternal Question: Why Are We Here?
Course Organization and Mechanics

A-10

The World of Problems

A vast array of problems can be solved with computers:

– mathematical calculations, simulations, graphics, data analysis, robotics control, trend prediction, scheduling, word processing, presentation development (thank you, computer)

Unfortunately, these problems are rarely expressed as sequences of bits.

We'll take a whack at the problems in blue!

A-11

Today's Outline

The World of Computers
The World of Problems
The World of Programming
The Eternal Question: Why Are We Here?
Course Organization and Mechanics

A-12

The World of Programming

Programmers stand between the world of problems and the world of the computer.

What is programming?

- take a problem
- analyze the problem
- envision a way to solve the problem
- take a computer
- encode the solution
- test & "debug"
- maintain the solution

A-13

What is a Program?

A program is a set of instructions to a computer.

Computers are *general-purpose* machines.

- i.e., just about useless without a program

Programs turn them into *special-purpose* devices capable of solving specific problems!

A-14

What is programming like?

Programming is like...

- solving "word problems in math"
  Yes, we also translate problems into a formal solution. Symbol manipulation is an integral part.
- writing a recipe
  Yes, we also require careful decomposition of instructions and planning ahead for execution.
- teaching a four-year old to calculate the wing resistance of a 747 over the telephone
  No. Four year olds don't take instructions nearly as literally as computers.

A-15

What skills does programming take?

Programming is a mixture of high-level creativity and low-level details.

Good programmers must learn to

- be creative problem solvers
- be meticulous artisans
- think about computers at many levels
- be incredibly patient!!

A-16

Machine Languages, High Level Languages

Computer hardware carries out instructions written in a *machine language*

- 1's and 0's – hard to understand or write!

A *high level language* is a notation that humans can understand and use more easily

A *compiler* translates a high level language to a machine language that can be executed

A-17

Evolution of Languages

Languages change as the need and technology for them changes…

- Machine Language - 1940's
- Fortran, Lisp - 1950's
- Cobol, Algol, APL, PL/I - 1960's
- Basic, Pascal, C - 1970's
- Smalltalk, C++, Modula, Ada, Prolog - 1980's
- Java 1990's

C is a bit retro but forms the basis of the most popular modern languages.

A-18

Back to the Future: CSE142JJ

Do you have that retro malaise?

Here's Ben Dugan with the cure for you!

Today's Outline

The World of Computers

The World of Problems

The World of Programming

The Eternal Question: Why Are We Here?

Course Organization and Mechanics

So, why are we here?

Computers are everywhere (ubiquitous?) and getting faster every year.

Many of the problems that need solving today can be solved (or assisted) by computers.

The skills of programming enable us to create new solutions and better use existing ones.

Plus, programming rocks!

Should you be here? (1 of 3)

"I am a complete novice to programming."

Prior programming experience is *not* required.

Still, *programming* a computer is very different from *using* one.

Being comfortable (or even expert) with applications is not the same as programming!

Should you be here? (2 of 3)

"I already know C and basic programming."

You *can* go directly to CSE 143.

- 142 credit available if you do well in 143
- Go there *today* to check it out: Guggenheim 224, 2:30 pm MWF
- *This* course may bore you but will still be time-consuming. You'll have to do things "our way."

Should you be here? (3 of 3)

"I am a woman or minority."

Should this stop you from being here?

NO!! Don't let it!

Many CS programs have skewed proportions of women and minorities. This is a problem.

But… computer science is for *everyone*!

Don't believe me?

Giants of CS (1 of 2)

From: http://www.agnesscott.edu/lriddle/women/love.htm

Ada Byron, Lady Lovelace (1815-1852)

- lived and died before computers
- brought up as mathematician (by her mother)
- predicted many of the feats of modern computers
- wrote the first computer program

The language "Ada" is named after her.

A-25

Giants of CS (2 of 2)

From: http://www.turing.org.uk/turing/

Alan Turing (1912-1954)

- British mathematician and codebreaker
- excellent cyclist
- invented the Turing Machine
- invented the Turing Test
- was openly gay

The most distinguished award in CS is the Turing Award.

A-26

Today's Outline

The World of Computers
The World of Problems
The World of Programming
The Eternal Question: Why Are We Here?
Course Organization and Mechanics

A-27

What is this class about?

UW Catalog Description:

Basic programming-in-the-small abilities and concepts. Highlights include procedural and functional abstraction with simple built-in data type manipulation. Basic abilities of writing, executing and debugging programs.

A-28

What does that mean?

Types of programs: small, mostly stand-alone
Techniques: C language, basic software engineering
Key concepts:

- the *tools* of programming
- the *rules* of programming
- the *way* of the programmer

A-29

The Tools of Programming

Techniques, ideas, terminology, and constructs that make programming possible and effective

- problem analysis
- divide-and-conquer
- "conditionals", "iteration", "recursion", etc.
- test-case design

A-30

The Rules of Programming

The constraints and syntax of the C language and of computers in general

- expressing programming constructs
- understanding and debugging errors
- familiarity with "artificial" languages

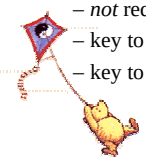
We use the C programming language.
It's widely used and industrial strength.
And… sometimes it's completely inscrutable.

A-31

The Way of the Programmer

Style and patterns of thought that make programs successful and comprehensible.

- *not* "necessary" parts of the program
- *not* required by the language
- key to becoming a truly proficient programmer
- key to creating elegant, beautiful code



The Tao of the Programmer?

http://www.just-pooh.com/tao.html

A-32

The Nitty-Gritty

Lectures three times per week:
guided thoughts and reflection on the material

Quiz section once a week:
Q&A, active learning, group work, quizzes

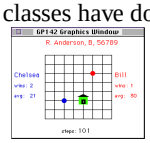
Programming projects:
In the lab or at home (with proper equipment)
Individual effort (not group work!)
Include questions beyond programming problem

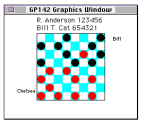
Two midterm exams plus a final

A-33

Homework can be Fun!

Past classes have done nifty games.





We'll do that, too. But, we'll also…

- play with robots
- solve a Car Talk puzzler
- pave the way for the release of Tron 2.0

A-34

More on Quiz Section

Designated sections

"low-background": for students without previous programming experience

"high-background": for students with considerable experience

All sections have identical assignments, tests, and grading criteria

Can request section swap in Wed. lecture

Please memorize your name, student ID#, quiz section ID, and your TA's name!

A-35

Final Exam
Wed., June 6, 2001

Times and rooms, but not the day, are likely to be different from the on-line Time Schedule (will be announced when we know the details)

With permission you can move to the exam period other than the one you are scheduled for.

If you have a problem with both times contact course administrator as soon as times are announced.

It will not be possible to take the final on any other day.

A-36

What to Expect

Is this a tough course?
- Contents are challenging
- Projects can be time-consuming
- Cramming won't work -- must keep up

Grades:
- Class average just below 3.0
- Always some 4.0's, always some 0.0's

Fun? Absolutely!

A-37

Can't get in?

All you can do is keep trying
- *No* waiting list, *no* lottery, ***no* entry codes**
- Matriculated ugrads have priority over grads and non-matriculated ugrads
- Historically many drops in the 1st two weeks

What to do while you wait?
- come to class, pick a section and go, do the work
- see the ugrad advisors in Sieg 114 (but *not* for entry codes!)

A-38

Advice

Keep up with the course day-by-day

Seek help early and often:
- TA, instructor office hours
- Lab consultants (IPL)
- Undergrad advisors in Sieg 114

  Some special tutoring is available

Consider joining a "low-background" section if you're new to programming

A-39

Course Staff: Here to Help You Succeed!

Instructors: You can go to either instructor's office hours

TAs:
- Teach sections & grade homework
- You can go to any TA's office hours

Lab staff: Operator (front-desk), CSE 142 Consultants
- there to help you get past stumbling blocks

Course administrator: Special arrangements, fix bookkeeping problems, claim abandoned work, etc.

A-40

Use Office Hours or…

Your TAs and instructors will *die* of boredom.

The facts about office hours:
- 63% of office hours by volume are used to hone "trash can basketball" skills
- 11% of TAs sniff glue to pass the time
- 3% bring inflatable buddies to stave off insanity
- 100% are awesome, dedicated, and love helping you!

No real TAs were harmed to create this clip art. (Do real TAs wear ties?)

A-41

Textbook and Materials

Text: "Problem Solving and Program Design in C" - Hanly and Koffman
- 3rd edition
- "self-check" and "quick-check" exercises highly recommended…some will be on quizzes! Verbatim!

Course Packets: slides, reference material
- Many students bring this to every lecture to take notes (recommended)
- Buy at: Professional Copy & Print, 4200 U. Way

A-42

CSE142 Web Site
http://www.cs.washington.edu/142/

Tattoo this URL on your forearm!

Messages from class mailing list (read often)

Homework projects: instructions, downloads, turn-in

Lecture schedule, slides, and current reading

Cool stuff like hall of fame and recorded lectures

Exam information, office hours, lab schedules

Tips, hints, et cetera, et cetera, etc.

A-43

Mailing List and Newsgroups
cse142@cs.washington.edu
uwash.class.cse142.bboard

Announcements, tips, hints, place to ask questions and get answers

"cse142-announce" mailing list for announcements from course staff

General discussions on the newsgroup

Learn to use the newsgroup today! And…

Tattoo its name on the other arm!

A-44

IPL: Intro. Programming Lab

Mary Gates Hall (MGH) 334

Pentium PC's running Windows
- Microsoft Visual C++ Version 6.0
- Web browsers
- Electronic mail

CSE142 consultants (posted hours)

Visit today!

A-45

Computing at Home

Stay connected with Web and e-mail

Get a compiler - MSVC++ 6.0 recommended
UW Bookstore has "Standard" edition for <$50.

Windows 9x/NT/2000 + MSVC is the official platform… some support for others

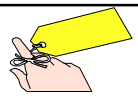
Do first project in IPL!
just to become familiar with it

Help on computing at home is on 142 web site

Expect a few headaches (but worth it!)

A-46

Tutorials



Optional tutorials, this week
Hands-on sessions in the IPL to get you familiar with the system
- Windows 95/98/NT, Web browser, basic MSVC, ...
- Meant for people unfamiliar with the software
  - No advanced stuff
- Can do assign. 0 (esp. part B) during tutorial
- Seating: first come, first served
- Length: about 1 hour

Location: IPL, MGH
Time: TBA

A-47

Homework #0

Due in 2 parts: This Friday(!) and Sunday/Monday

Read Chapter 1 and handouts.

Go to IPL and start learning the system. Be sure and read section 1.2 before going to lab.

Start playing with the other software tools.

There's lots to read during the quarter: Start going & keep going!

A-48

Final Thoughts

Fill out section swap form and hand in as you leave *if* you want to switch between high-low-regular background section *at your currently scheduled time*

Remember tutorials in the IPL

Get started!!

A-49

QOTD (Question of the Day)

Every lecture will end with a question.

One of the three questions will be on the quiz.

(we might change it just a bit on the quiz)

You can work on these with *anyone as much as you like*! In fact, we encourage you to!

(but don't just copy… work together!)

But… no notes at the quiz.

A-50

Today's QOTD: When Compilers Were People

Imagine you have a sister in grade school.

She's just learning to multiply.

She knows how to add numbers.

Compile for her the problem of multiplying six by nine into a problem that only requires adding numbers.

6 * 9 ➔ *addition problem*

A-51