Welcome to CS2110!

Learning about:
- OO, abstract data types, generics, Java Collections, ...
- Reasoning about complex problems, analyzing algorithms we create to solve them, and implementing algorithms with elegant, easy-to-understand, correct code
- Testing; Reasoning about correctness
- Data structures: linked lists, trees, hash tables, graphs, etc.
- Recursion
- Algorithmic complexity
- Parallelism — threads of execution

Usefulness of 2110

This summer I’m working in particle physics, making simulations of some of the background signal we’d expect to see in our detector for an experiment run in the particle accelerator. What I’m working on is a clustering algorithm to put together energy deposits from several quantized points in the detector to learn what the initial particle’s energy and position was. After some thought, I decided the best first sweep over this data would be to do a depth first search starting about a high energy deposition in the calorimeter. It works great, and my PI was very excited about the results!

Usefulness of 2110

I am working at a startup in Silicon Valley this summer … that does subscription-based financial management and billing among other things. It has been pretty incredible the amount I’ve learned from your class that relates to this internship and I have definitely decided to pursue computer science (I was initially engineering physics).
Is CS2110 right for you?

- Knowledge of Java not required
  - Only ~30% of you know Java — others know Matlab, Python ...
  - Requirement: comfort with some programming language, on the level of CS1110 (Python based) and CS1112 (Matlab based).
  - Prior knowledge of OO not required.
- We assume you do not know Java!
- If you know Java, the first 3 weeks will be easier for you but you STILL have to learn things, probably unlearn what you learned

Homework!

- Homework 1. Read article Why Software is So Bad. Link: Course website -> Lectures notes (Lecture 1)
- Homework 2. Get Java, Eclipse, DrJava on your computer.
- Homework 3. Spend some time perusing the course website. Look at course information, resources, links, etc.
- Homework 4. BEFORE EACH LECTURE/RECITATION: download pdf form of the slides, bring to class and look at them during lecture. We project not only PPT but also Eclipse and other things. Having PPT slides in paper form or on your laptop/tablet can help you during the lecture.

Lectures

- TR 10:10-11am, Statler auditorium
  - Attendance mandatory

- ENGRD 2110 or CS 2110?
  - Same course! We call it CS 2110 in online materials
  - Non-engineers sign up for CS 2110
  - Engineers should sign up for ENGRD 2110

Sections (Recitations)

- T 12:20 4 sections:
- T 1:25 2 sections:
- T 2:30 2 sections:
- T 3:35 1 section:
- W 12:20 2 sections:
- W 01:25 2 sections:
- W 02:30 2 sections:
- W 07:30 1 section:

- Some time EARLY, visit StudentCenter and change your section to even out the numbers
- Attendance mandatory
- Sometimes flipped: you watch videos beforehand, come to recitation and do something
- Sometimes review, help on homework, new material
- No permission needed to switch sections, but do register for whichever one you attend

CS2111

- An "enrichment" course
- Help students who might feel overwhelmed by CS2110
- Gives more explanation of core ideas behind Java, programming, data structures, assignments, etc.
- Taught by Birell, Gries, Sampson, 1 credit S/U
- Only for students who also take CS2110
- Only requirement: Attend weekly lecture

I would just like to thank you for taking the time to hold CS2111 this year. You have no idea how the class helped and impacted a lot of us. I would never had "survived" CS2110 without your generous share of your knowledge. I appreciated your time.

Academic Excellence Workshops

- Two-hour labs: students work together in cooperative setting
- One credit S/U course based on attendance
- Time and location TBA
- Visit Olin 167 and ask about AEWs
- See website for more info: www.engineering.cornell.edu/academics/undergraduate/curriculum/courses/workshops/index.cfm
Piazza

- Click link on our "links" web page to register
- Incredible resource for 24 x 7 help with anything
- We keep an eye on it and answer questions. YOU can (and will) too. Visit the Piazza often.

Resources

- JavaHyperText. Course website: Link on Links or Resources page
- Java resource: online materials at Oracle JDK web site
- VideoNote: indexed videos of our lectures. http://www.videonote.com/cornell. Log in with netid
- Book: Frank M. Carrano, Data Structures and Abstractions with Java, 3rd ed., Prentice Hall. It is OPTIONAL!
  - 2nd edition is okay. E-book not required
- Share textbook. Need access to it from time to time
- Copies on reserve in Engineering Library

Obtaining Java and Eclipse

- Follow instructions on our Resources web page
  - Make sure you have Java JDK 1.8, if not download and install. We explain how on the web page.
  - Then download and install the Eclipse IDE
- Test it out: launch Eclipse and click “new>Java Project”
  - This is one of a few ways Java can be used
  - When program runs, output is visible in a little console window

DrJava IDE

- IDE: Integrated Development Environment
- DrJava is a much simpler IDE, few features
- We use it only to demo Java features and programming concepts. Has an “interactions pane”, which allows trying things without requiring a complete Java program. Great tool!
- DON’T use it for course assignments –use Eclipse
- Download jar file from links page of course website

Coursework

- 7–8 programming assignments (37%)
- Two prelims (14% and 16%)
- Final exam (30%)
- Course evaluation (1%)
- Work in recitations (1-3%)
Formula will change as course progresses and we make changes in assignments, give quizzes, etc.

Assignments: a real learning experience

- Teams of one or two
  - A0 and then A1 will be posted soon on the CMS
  - Finding a partner: choose your own or contact your TA. Piazza can be helpful.

One way to do an assignment. Wait until the day before it is due.

Results: Frustration, anger, impatience, long lines in consulting room. No fun.
Not a good educational experience
Assignments: a real learning experience

One way to do an assignment:
Read the handout immediately.
Work on it every (other) day.
Ponder. Look things up. Get help in
consulting room, with no lines, or
office hours. Fun, hard work, a
great learning experience.

Piano lessons:
Practice Daily?
Or put off practicing
until an hour before
weekly lesson?

Academic Integrity… Trust but verify!

- 98% of you are honest and don’t try to cheat
- We use artificial intelligence tools to check each
  homework assignment, so catch the other 2%
  - The software is accurate!
  - It tests your code and notices similarities between code
    written by different people
- Sure, you can fool this software
  - … but it’s easier to just do the assignments
  - … and if you try to fool it and screw up, you might fail the
    assignment or even the whole course.

Types in Java

References in JavaHyperText
type
variable
variable declaration
primitive types. A lot of detail
constants: look under Literal
assignment. Has a 2.5-minute video
casting
expression
precedence
unicode (character representation)

Type: Set of values
together with operations on them.

Type integer:
values: ..., −3, −2, −1, 0, 1, 2, 3, ...
operations: +, −, *, /, unary −

God’s integers!
Can represent them
in many ways —
decimal, binary, octal, maybe as
strokes ||||
(that’s 4)
Do you know how
your computer
represents them?

The integers as the basis

Leopold Kronecker [1823-1891], Prussian mathematician,
Argued that arithmetic and analysis should be
founded on the whole numbers (integers):
Die ganzen Zahlen hat der liebe Gott gemacht,
alles andere ist Menschenerwerk.
The beloved God made the whole numbers,
everything else is the work of man.

He insisted on the constructibility of math
objects. Real numbers—do they really exist?
You can’t compute most of them because they have
an infinite number of digits.

Type: Set of values
together with operations on them.

Matlab and Python are weekly typed:
One variable can contain at different
times a number, a string, an array, etc.
One isn’t so concerned with types.
Java strongly typed:
A variable must be declared before
it is used and can contain only values
of the type with which it is declared

Valid Python sequence:
x = 100;
x = ‘Hello World’;
x = [1, 2, 3, 4, 5];

Corresponding Java
int x;
x = 100;
x = “Hello”;

Illegal assignment:
“Hello” is not an int

Declaration of x:
x can contain only
values of type int
Weakly typed versus strongly typed

**Weakly typed:**
Shorter programs, generally. Programmer has more freedom, language is more liberal in applying operations to values.

**Strongly typed:**
Programmer has to be more disciplined. Declarations provide a place for comments about variables. More errors caught at compile-time (e.g., it’s a syntax error to assign a string to an int variable).

Note: weak and strong typing not well defined; literature has several definitions.

Type: Set of values together with operations on them.

Java Type int:
- values: $-2^{31}$.. $2^{31}$-1
- operations: +, -, *, /, %, unary –

$b \% c : \text{remainder}$ when b is divided by c.

$67 \% 60 = 7$

Java designers decided on this Principle: Primitive operations on type int should yield an int.

Most-used ‘primitive’ types

**int:** values: $-2^{31}$.. $2^{31}$-1
- operations: +, -, *, /, %, unary –

$b \% c : \text{remainder}$ when b is divided by c.

$67 \% 60 = 7$

Write values in “scientific notation”

**double:** values like: -22.51E6, 24.9
- operations: +, -, *, /, %, unary –

Use single quotes for type char.

‘n’ is new-line char

**char:** values like: ‘V’ ‘S’ ‘n’
- operations: none

**boolean:** values: true false
- operations: ! (not), && (and), || (or)

Can’t use integers as booleans!

About ‘primitive’ type int

Java Principle: A basic operation of type int must produce an int

Integer.MAX_VALUE: name for max int value: $2^{31}$-1: 2147483647
Integer.MAX_VALUE + 1 is $-2^{31}$: -2147483648 WRAP-AROUND

Primitive number types

<table>
<thead>
<tr>
<th>Integer types:</th>
<th>byte</th>
<th>short</th>
<th>int</th>
<th>long</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>2 bytes</td>
<td>4 bytes</td>
<td>8 bytes</td>
<td>usual operators</td>
</tr>
</tbody>
</table>

Real types:
- float 4 bytes 8 bytes
- double 8 bytes

-22.51E6 24.9

Use these to save space.

Have an array of 1,000,000 integers in range 0..7?

Use a byte array rather than an int array.

Don’t worry about this in next 7-8 weeks. Use int and double.

Casting among types

(int) 3.2 casts double value 3.2 to an int

any number type casts any number expression

narrow may be automatic cast wider

byte short int long float double

must be explicit cast, may truncate

(int) is a unary prefix operator, just like –

$-3$ evaluates to 3

$-(\text{int}) 3.2$ evaluates to -3
char is a number type!

Declaration: gives name of variable, type of value it can contain

```plaintext
int x;
```
Declaration of `x`, can contain an `int` value

```plaintext
double area;
```
Declaration of `area`, can contain a `double` value

```plaintext
int[] a;
```
Declaration of `a`, can contain a pointer to an `int` array. We explain arrays much later

Unicode: 16-bit char repr. Encodes chars in just about all languages. In java, use hexadecimal (base 16) char literals:

```
\u0041 is 'A'
\u0042 is 'B'
\u0056 is 'V'
\u0024 is '$'
```

See www.unicode.org

Assignment statement

Much like in other languages —need `;` at end:

```plaintext
<variable> = <expression>;
```

<table>
<thead>
<tr>
<th>int x;</th>
</tr>
</thead>
<tbody>
<tr>
<td>x=10;</td>
</tr>
<tr>
<td>... other code</td>
</tr>
<tr>
<td>x=x+1;</td>
</tr>
</tbody>
</table>

Have to declare `x` before assigning to it.

| int x=10; |
| ... other code |
| x=x+1; |

Can combine declaration with an initializing assignment. Shorthand for a declaration followed by an assignment.

```plaintext
int x=10;
... other code
x=x+1;
```

Assignment statement type restriction

Every expression has a type, which depends on its operators and the types of its operands in a natural way.  

**Rule**: In `x=e;`, type of `e` has to be same as or narrower than type of `x`. Reason: To avoid possibly losing info without the programmer realizing it.

```plaintext
double y = 5 + 1;
```
The value of 5+1 is automatically cast from type `int` to type `double`.

```plaintext
int x=25.5 + 1;  
```
Illegal: The exp value is of type `double`.

```plaintext
int x=(int)(75.5 + 1);
```
You can cast to `int` explicitly. 76 will be stored in `x`.

A function in Matlab, Python, and Java

```plaintext
function s = sum(a, b)  
    % Return sum of a and b  
    s= a + b;
```
Matlab

```plaintext
def sum(a, b):
    """ return sum of a and b""
    return a + b
```
Python

```plaintext
/ ** return sum of a and b */
public static double sum(double a, double b) {
    return a + b;
}
```
Java

```plaintext
Specication in comment before function
Declarations of parameters a and b
```