

THERE IS ROOM IN THE BALCONY!

Romeo, Romeo, where art thou!

Up in the balcony, where it's cheaper!

CS/ENGRD 2110
SPRING 2019

Lecture 1: Overview and intro to types
<http://courses.cs.cornell.edu/cs2110/2019sp>

CS2110

- Object-oriented programming, reasoning about complex problems
- Testing; Reasoning about correctness
- Program development
- Algorithmic complexity, analyzing algorithms,
- Data structures: linked lists, trees, hash tables, graphs, etc.
- Programming paradigms: recursion, parallel execution

Email from July 2017!

I'm a rising junior studying ... I took 2110 in Sp 2016. At the time, I couldn't appreciate everything I was taught and how they were taught—I was mostly preoccupied with completing the assignments and passing. But I'm starting to realize how terrific a job you and the entire 2110 staff did teaching the material.

I'm at Verizon as a software engineering intern. Doing JUnit testing has become very routine; it was stressed heavily in 2110. I've had to use basically all my knowledge of Java and OO programming for my internship, and I can safely say that I would be screwed if I hadn't taken 2110.

While taking 2100, a lot of peers (including myself) were frustrated at how we had to master all the subtle nuances of Java. But now I confidently say that I am so thankful that I know all these concepts.

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 a clustering algorithm to put together energy depositions 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 some things, too!**

Welcome to CS2110!

OO Programming and Data Structures

405 Freshmen
158 Sophomores
058 Juniors
031 Seniors
039 Meng/Masters
025 PhD

Instructors:
Michael Clarkson
David Gries

Recitation leaders (TAs): 23
Consultants: 21

716 Total

Letter grade: 688
S/U grade: 22
AUDIT: 6

As of Fri evening, 18 Jan

Lectures

- TR 10:10-11 am, 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)

Some time EARLY, visit
StudentCenter and
change your section to
even out the numbers

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

- 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

Recitation this week

- Eclipse essentials
- Practice with common types
- DO BEFOREHAND:**
 - Install Java, Eclipse, DrJava (optional)
 - Watch tutorials on API & String

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.

Exams are most important aspect in determining final grade

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

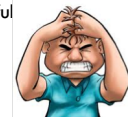
One way to do

an assignment:

Wait until the day before it is due.

Result: Frustration, anger, impatience, long lines in consulting room. No fun.

Not a good educational experience



Assignments: a real learning experience

13

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!

14



- 99% of you are honest and don't try to cheat
- We use artificial intelligence tools to check some assignments, so catch the other 1%
 - ▣ 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.

Resources

15

- **JavaHyperText**. Course website: Link on Links or Resources page
<http://www.cs.cornell.edu/courses/JavaAndDS/definitions.html>
- Java resource: online materials at Oracle JDK web site
<https://docs.oracle.com/javase/8/docs/api/index.html?java/lang/Object.html>

Piazza

16

- 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.



CS2111

17

- 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 Bracy, Gries, and a TA, 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 have "survived" CS2110 without your generous share of your knowledge. I appreciated your time.

Obtaining Java and Eclipse

18



- 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



19

- 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

Homework!

20

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.

Assignment A0

21

- Introduction to Java, Eclipse, and the assert statement
- Due 123123123123 January at 11:59pm
- Submit on CMS

AND
NOW

Relevant JavaHyperText Entries

23

type
 primitive type
 type, strong versus weak typing
 cast
 function function call
 procedure procedure call
 variable variable declaration
 expression
 assignment statement
 unicode

Type: Set of values
 together with operations on them.

24

Type integer:

values: ..., -3, -2, -1, 0, 1, 2, 3, ...

operations: +, -, *, /, unary -

Can represent them
 in many ways —
 decimal, binary,
 octal, maybe as
 strokes ||||
 (that's 4)

Do you know how
 your computer
 represents them?

Type: Set of values together with operations on them.

25

Java Type **int**:

values: $-2^{31} \dots 2^{31}-1$

operations: +, -, *, /, %, unary -

$b \% c$: remainder
when b is divided by c .
 $67 \% 60 = 7$

Integer.MAX_VALUE: name for max **int** value: $2^{31}-1$: 2147483647

What do you think happens when **Integer.MAX_VALUE + 1** is evaluated?
Talk to neighbors about possibilities. **Know Java? KEEP QUIET**

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

About int

26

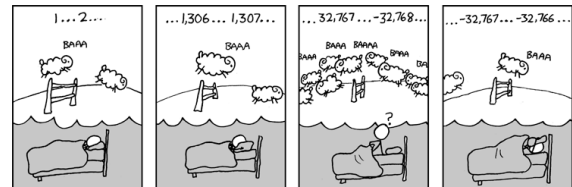
Java Principle: A basic operation of type **int** must produce an **int**

int: values: $-2^{31} \dots 2^{31}-1$, i.e.

operations: +, -, *, /, %, unary -

Integer.MAX_VALUE: name for max **int** value: $2^{31}-1$: 2147483647

Integer.MAX_VALUE + 1 is -2^{31} : -2147483648 **WRAP-AROUND**



Types in Java

27

Primitive Types

- Fully integrated into Java
- We'll cover these today!

Classes

- We'll talk about these on Thursday

Most-used 'primitive' types

28

int: values: $-2^{31} \dots 2^{31}-1$

operations: +, -, *, /, %, unary -

$b \% c$: remainder
when b is divided by c .
 $67 \% 60 = 7$

double: values like: -22.51E6, 24.9

operations: +, -, *, /, %, unary -

Write values in
"scientific notation"

char: values like: 'V' '\$' '\n'

operations: none

Use single quotes for
type char.
'\n' is new-line char

boolean: values: true false

operations: ! (not), && (and), || (or)

Can't use integers
as booleans!

Primitive number types

29

Integer types: **byte** 1 byte, **short** 2 bytes, **int** 4 bytes, **long** 8 bytes, usual operators

Real types: **float** 4 bytes, **double** 8 bytes, -22.51E6, 24.9, usual operators

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.

Type: Set of values together with operations on them.

30

Matlab and Python are **weakly typed**:

One variable can contain at different times a number, a string, an array, etc.
One isn't so concerned with types.

Valid Python sequence:
`x = 100;`
`x = 'Hello World';`
`x = (1, 2, 3, 4, 5);`

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

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

31

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

Basic variable declaration

32

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

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

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

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

`x` 5 `int` `area` 20.1 `double` `a` `int[]`

Assignment statement

33

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

`<variable> = <expression>;`

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

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.

Assignment statement type restriction

34

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.

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

~~`int x = 75.5 + 1;`~~ Illegal: The exp value is of type **double**.

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

Casting among types

35

`(int) 3.2` casts **double** value 3.2 to an **int**

any number type

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
— `(int) 3.2` evaluates to —3

char is a number type!

36

char is a number type: `(int) 'V'` `(char) 86`

Unicode repr. in decimal: 86

'V'

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 '\$'

`'\u0950'` is 'ॐ'

`'\u5927'` is '大'

`'\u885b'` is '衛'

—Om, the sound of the universe
—大衛 is (I think) a transliteration of David into Chinese (Da Wei)

See www.unicode.org

A function in Matlab, Python, and Java

37

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

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

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

Specification:
 in comment
 before function

return type

Declarations of
 parameters a and b