CS/ENGRD 2110 FALL 2014

Lecture 1: Overview and intro to types
http://courses.cs.cornell.edu/cs2110/2014fa

Welcome to CS2110!

2

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, graphs, etc.
- Recursion
- Algorithmic complexity
- Parallelism —threads of execution

Homework!

3

Homework 1. Read article Why Software is So Bad.
Link: Course website -> Lectures notes (Lecture 1)
Homework 2. Get Java and Eclipse on your computer
Homework 3. Spend some time perusing the course website.
Look at course information, resources, links, etc.

Homework 4. BEFORE EACH LECTURE: download the pdf form of the slides, bring them to class, and look through them during the lecture. We will be projecting not only PPT but also Eclipse and other things; having the PPT slides in paper form or on you laptop/tablet can help you during the lecture

What's CS 2110 about?

4

- Computational tools are "universal" but the key is to master computational thinking.
  - Looking at problems in ways that lead naturally to highly effective, correct, computational solutions
  - There are many ways to do anything, but some are far better than others
- Mastery of computational thinking will help you become a master of the universe!
- Great job prospects with high salaries…

Is CS2110 right for you?

5

- Knowledge of Java not required
  - Only ~30% of you know Java –others know Matlab, Python …
  - Requirement: comfort with some programming language.
    Prior knowledge of OO not required.
  - We assume you do not know Java!
- Don't take CS1110 just because you are worried that your high school programming experience won't do
- *Consider taking the honors version: CS2112, taught by Andrew Meyers. Given at same time. Switching between 2110 and 2112 during first 2 weeks is fine.*

Lectures

6

- 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 sign up for ENGRD 2110

Sections

7

- Like lecture, attendance is mandatory
- Sometimes review, help on homework
- Sometimes new material
- Section numbers are different for CS and ENGRD
- Each section led by member of teaching staff
- No permission needed to switch sections, but do register for whichever one you attend

Adding sections

8

T 12:20	3 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	1 section
W 07:30	1 section

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

Example. For T 12:20, instead of 48 students/section, we will have 33.

CS2111

9

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

Academic Excellence Workshops

10

- Two-hour labs: students work together in cooperative setting
- *One credit S/U course based on attendance*
- Time and location TBA
- See website for more info:

www.engineering.cornell.edu/academics/undergraduate/curriculum/courses/workshops/index.cfm

Piazza

11

- Click link on our 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

12

- Book: Frank M. Carrano, *Data Structures and Abstractions with Java*, 3rd ed., *Prentice Hall*
  - *Note: 2nd edition is okay*
  - Share textbook: fantastic idea. You do need access to it from time to time
  - Copies on reserve in Engr Library
- Additional material on Prentice Hall website
  - "e-Book" not required
- PPT slides (on course website and Piazza) outline all of OO in Java. Has index at beginning
- Great Java resource: online materials at Oracle JDK web site. Google has it indexed.

Obtaining Java

13

- Follow instructions on our Resources web page
  - Make sure you have Java JDK 1.7, 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

Eclipse IDE

14

- IDE: Integrated Development Environment
  - Helps you write your code
  - Protects against many common mistakes
  - At runtime, helps with debugging
- Follow Resources link to download and install

DrJava IDE

15

- 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.
- DON’T use it for course assignments –use Eclipse
- Free at www.drjava.org

Coursework

16

- 6–7 assignments involving both programming and written answers (35%)
- Two prelims (15% each)
- Final exam (30%)
- Course evaluation (1%)
- Possible surprise in-class quizzes (4%)

Formula will change as the course progresses and we make changes in assignments, give quizzes, etc.

Exams are most important aspect in determining final grade

Assignments

17

- 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.

Two kinds of assignment:
Vanilla: specific experience to learn and practice what’s being taught. We give exact instructions for doing it
Chocolate: Open-ended project done in 3 chunks
Parts of the design are left to you.
CS 2111 will give more help on it.

Academic Integrity… Trust but verify!

18

- We use artificial intelligence tools to check each homework assignment
  - The software is very accurate!
  - It tests your code and also 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

19

References in text and in JavaSummary

- type: A.14 slide 4
- variable: A.13 slide 7
- variable declaration: A.15 slide 7
- Primitive types, A.16, back inside cover slide 5
- Constants, A.17 slide 20
- Assignment, A.18-A.20 slide 8
- Casting, A.21 slide 6
- Expressions: A.22-A.23
- Precedences: A.24, back inside cover
- Unicode character codes, back inside cover

Type: Set of values together with operations on them.

20

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

21

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 Menschenwerk.

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.

God's integers!

Type: Set of values together with operations on them.

22

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

23

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.

24

Java Type int:

values: -2^{31} .. $2^{31}-1$

operations: +, –, *, /, %, unary –

b % c : *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**.

What, then, could be the value of Integer.MAX_VALUE + 1?

Most-used 'primitive' types

Inside back cover, A-6..7

25

int: values: -2^{31} .. $2^{31}-1$
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!

About 'primitive' type int

Inside back cover, A-6..7

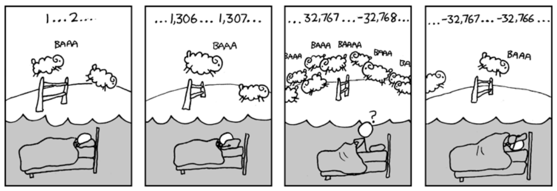
26

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

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

Inside back cover, A-6..7

27

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.

Casting among types

Page A-9, inside back cover

28

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

any number type

any number expression

narrow $\xrightarrow{\text{may be automatic cast}}$ wider

byte short int long float double

$\xleftarrow{\text{must be explicit cast, may truncate}}$

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

– – 3 evaluates to 3
– (**int**) 3.2 evaluates to –3

Char is a number type!

Page A-9, inside back cover

29

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' —Om, the sound of the universe
'\u5927' is '大' —大衛 is (I think) a transliteration
'\u885b' is '衛' of David into Chinese (Da Wei)

See www.unicode.org

Basic Variable Declaration

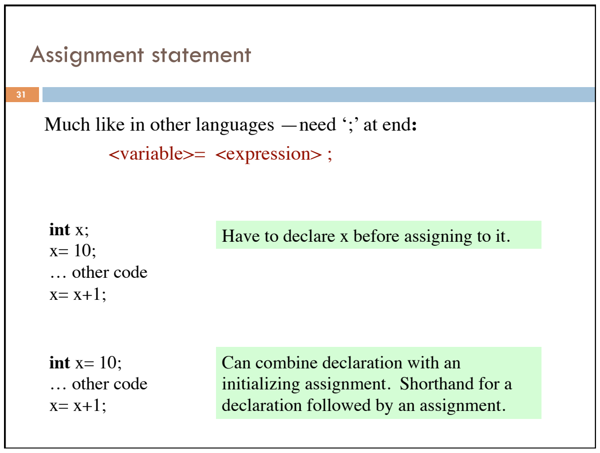
Page A-6

30

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 $\boxed{5}$ **int** area $\boxed{20.1}$ **double** a $\boxed{} \rightarrow$ **int**[]



Assignment statement

31

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

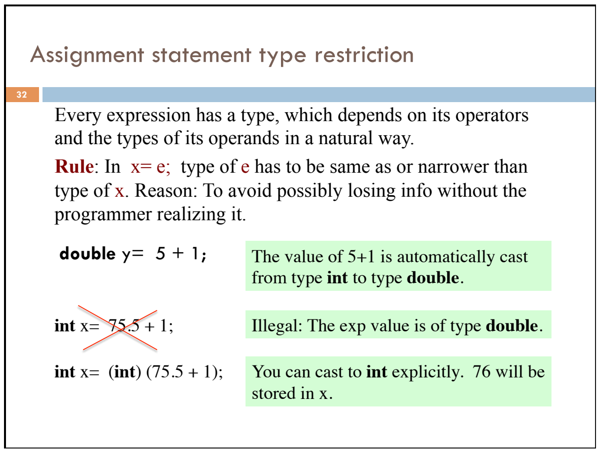
<variable>= <expression> ;

```
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

32

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;

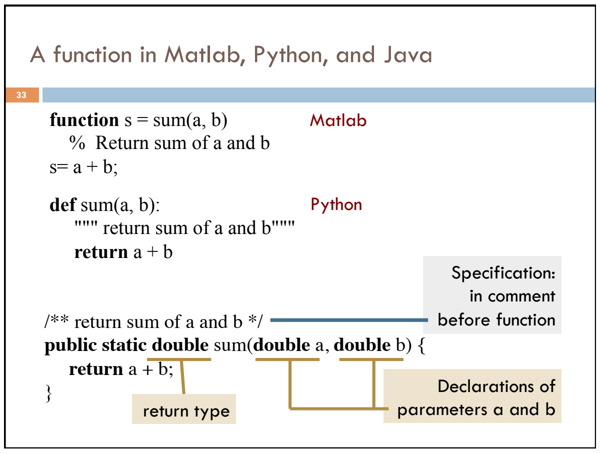
The value of 5+1 is automatically cast from type **int** to type **double**.

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.



A function in Matlab, Python, and Java

33

Matlab

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

Python

```
def sum(a, b):
    """ 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