CS102
Introduction to
data structures, algorithms,
and object-oriented
programming
Computer Science II covers:
- fundamental data structuring concepts, including abstract data types (e.g., stacks, queues, lists, trees, hash tables, and graphs)
- fundamental algorithms (e.g., searching, sorting, minimum spanning tree, and shortest paths)
- introduction to analysis of algorithm complexity
- input and output to files and streams

At the same time, students are introduced to the principles of object-oriented programming using Java.
Textbooks

• **Required text:**

  *Introduction to Programming Using Java,* David J. Eck ([http://math.hws.edu/javanotes.](http://math.hws.edu/javanotes.))

• **Recommended text:**

  *Algorithms, 4th Edition,* Sedgewick and Wayne
  (available on-line: [http://www.aw.com/onto](http://www.aw.com/onto))

Other introductory Java books are available for checkout from the instructor, plus there are many on-line Java tutorials.
Course Requirements

Students are required to:

1. complete *all* programming assignments over the course of the semester (8 to 10),
2. take a midterm and a final exam,
3. do paper-and-pencil quizzes (timing at the discretion of the professor), and
4. complete *all* weekly laboratory assignments (8 to 10).
Assignment Lateness Policy

All programming assignments are due at midnight on the date specified.

Late assignments will receive a 5% penalty for every day they are late (including weekend days because submission is electronic through Moodle).

No assignment that is overdue by more than 8 days will be accepted, except when a valid excuse is presented. No assignment will be accepted after solutions have been posted.
Valid Excuses & Approval of Late Submission

Valid excuses must come through Baldwin Health Services or the Dean of Studies office.

If you have a reason to hand an assignment in late, but you don't have a valid excuse from one of the above sources, make arrangements prior to the due date with your instructor or your score will accrue the daily penalty described on the previous slide.
Grading

Final grades will be computed as follows:

• 30% Assignments

• 15% Labs and Class participation

• 25% Midterm exam

• 30% Final exam (regularly scheduled time)
Program Grading

Programs will be graded as follows:

- 60% for correctness (program compiles and runs)
- 20% for documentation (comments explaining program)
- 10% for following directions (i.e., formatting output and general program operation)
- 10% for programming style and design (use of constants, modularity, information hiding, procedural abstraction)
Academic Integrity

All of your code must be original, written by you.

You may discuss general approaches with other students or help them find bugs in their code.

However, any copying (electronic or otherwise) of another person's code or code fragments violates academic integrity.

*Any student(s) suspected of going beyond the limits of academic integrity will be referred to the Dean of Students and the Academic Panel.*
Course Moodle Page

• All information about the course, syllabus, assignments, schedule, etc. is on our course Moodle page. *Check it frequently.*

• Check your e-mail frequently for other course announcements.
Office Hours & Contact Information  
Section 01 - Jenny Walter

**Office hours:** Mon & Wed, noon to 2 pm  
Thu, 2 to 4 pm

**Office:** SP 306

**E-Mails:** jewalter@vassar.edu

**Office phone:** ext. 7449  
(leave message if no answer)

Reading assignment for 1st week: Chapters 1 and 2 of book at http://math.hws.edu/javanotes.
Scheme or Racket, *functional* languages are designed to

♦ organize algorithms as collections of interacting *functions*, where the output of one function is input to another.

Java is an *object-oriented* language, which is designed to

♦ organize algorithms as collections of interacting *objects*, and

♦ create *reusable, portable* code with a hierarchical *class* structure.
Writing a Java Application

STEP 1:
- Create a file containing the text definition of a class (i.e., the code)
  - Program must start with class file name.
  - File must have a “.java” suffix.
  - The prefix of the file name must be same as class name inside the file.
    E.g., Widget.java should contain definition of class “Widget”

NOTES
- Several classes may combine to form a complete program, called an application, but at least one class per application must contain the main method (execution starts here).

    public static void main(String[] args) { ... }
STEP 2:
- Run the program through the Java compiler (javac), which translates the code into **Java Byte Code**
  - Machine-independent representation, enables portability among platforms
  - Byte code is stored in a file with same prefix as class definition file, with suffix “.class”
    - E.g., output from successful compilation for HelloWorld.java would be HelloWorld.class
  - Normally have to edit file and recompile several times to fix syntax errors.

STEP 3:
- Run the byte code interpreter (java) on your byte code file
  - Program runs, possibly taking input during its operation and producing output
At the command line, you type:

```
javac HelloWorld.java
```

```
java HelloWorld
```

Your code file

Input

Output
How to compile and run a Java application file in Linux

```bash
$ ls
HelloWorld.java

$ javac HelloWorld.java

$ ls
HelloWorld.class
HelloWorld.java

$ java HelloWorld
Hello World!

$
Common Linux OS Commands

- **ls**: (small L and S) show filenames in current directory

- **cd <dirname>**: change current active session to subdirectory <name>...with no angle brackets. (Note that typing cd alone returns current active session to your home directory).

- **cp <filename> <newfilename>**: Copy the contents of file <filename> to file <newfilename> (creates or writes over <newfilename>).

- **cp -r <dirname> <newdirname>**: Copy the contents of directory <dirname> to directory <newdirname> (creates or copies over <newdirname>).
Common Linux OS Commands

- **more <filename>** : scrolls contents of file <filename> to the screen in one pass only. Hit spacebar to see next screen full.

- **less <filename>** : displays contents of file <filename> to the screen, allowing cursor movement within file. Type “q” to exit.

- **man <command>** : displays help file for specified <command>.

- **enscript <filename>** : print contents of <filename> to printer in SP 307 in portrait orientation.

- **enscript -2r <filename>** : print contents of <filename> to printer in SP 307 in landscape mode with 2 columns per page.
DrJava IDE

• There are many powerful development software packages (e.g., eclipse, netbeans, intelliJ, blueJ). We will work with a simple editor called DrJava, freely available on the web. We will also start by focusing on compiling and running programs at the command line.
CS Network Accounts

• If you do not have a CS account from last year, or if you can’t remember your password, see Jerry, our Sys Admin.

• At the first lab this afternoon, we will have you log in to your accounts and create and run a Java program.
DrJava

• This IDE is available on all the networked CS machines.

• It is also freely downloadable from a site you will find if you google DrJava.

• If you have trouble installing DrJava, see me or one of the coaches. The install contains the Java JDK and a terminal application.