Welcome to CS 102: Data Structures

Course web page: http://cs.nyu.edu/courses/fall13/CSCI-UA.0102-001/
Teaching Assistant: Joanna Klukowska joannakl@cs.nyu.edu
Course communication: NYUClasses (via Academics tab on NYU Home)
Course-related communication, as discussed below, will be through NYUClasses. However, if you need to contact the instructor with a personal matter, please send an email with the subject line “NYU-CS-102”.

<table>
<thead>
<tr>
<th>Section 001, 002</th>
<th>Section 004</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructor: Eric Koskinen</td>
<td>Instructor: Daniel Schwartz-Narbonne</td>
</tr>
<tr>
<td>Lectures: Tue/Thur 15:30--16:45 WWH109</td>
<td>Lectures: Mon/Wed 14:00--15:15 WWH109</td>
</tr>
<tr>
<td>Office hours: Thur 11:00--12:00 WWH402</td>
<td>Office hours: Mon 15:30--16:30 WWH425</td>
</tr>
</tbody>
</table>

Goals
The goal of CS 102 is to teach fundamental concepts of programming that will enable you to solve interesting, challenging real-world problems with reliable, modular programs that can be tested, extended, shared with others, and combined effectively with other programs. You will learn about:

- data types and data abstraction
- how data is represented in memory
- how to decompose complex programming problems into manageable subproblems
- how and when to use elementary data structures such as arrays, lists, trees, and maps
- different approaches to structuring programs (object-oriented, imperative, functional)
- communication between programs and their environment
- how to test and fix programs (unit testing, debugging).

By the end of this course, you should have:

- The ability to write larger (~1000 lines) programs
- Fluency in program design
  - test-driven programming
  - modular decomposition
  - working with different programming idioms
- A “toolbox” of data-structures and algorithms you can use as you build programs
- Fluency in core Java
- Firm grasp of the basic principles of Computer Science

Prerequisite
The prerequisite for this course is Introduction to Computer Science (CSCI-UA.0101). This course is a more challenging and rigorous extension to 101. If you had difficulty with 101, please discuss with us whether 102 is appropriate for you.
Programming languages

In this course, implementations will be focused on Java. All programming elements of homework and exam questions will be in Java. You are required to be familiar with Java (i.e. taken CS101). This course also offers the opportunity to learn a functional language, as represented by Ocaml. From time-to-time you may receive bonus points by completing bonus questions that involve Ocaml. You are encouraged to attempt these questions because we think you will find that Ocaml often permits you to express data-structures and algorithms in a manner that is more intuitive and natural than Java.

Textbooks

We will be using the following textbook for Java:


For Ocaml, we recommend the following texts (available free online)

- *Introduction to Objective Caml*. By Hickley.
- The tutorials at [http://ocaml.org/tutorials/](http://ocaml.org/tutorials/)
- *Developing Applications with Objective Caml*. By Chailloux, Manoury and Pagano.

Communication

All class electronic communication will occur through NYUClasses (on the Academics tab of NYUHome). This includes announcements, assignments, discussions, etc. You will need to check this web site regularly.

**Forum.** If you have a question about the class, it should be made on the NYUClasses web site. This provides the maximum opportunity for both the course staff and your classmates to help you, and ensures that all students will have access to the replies. You should only directly email the course staff for matters which require confidentiality.

Grading

Course grades will be derived from a combination of

- Homework (40%)
- Midterm Exam (20%)
- Final Exam (40%)

Bonus points if your homework submissions make us laugh (and are otherwise correct).
Assignments
There will be programming assignments throughout the semester, approximately one every 10 days, posted to the course web page. These assignments will be posted to NYUClasses. You will also submit your completed assignments through NYUClasses.

Exams
There will be a mid-term examination on 9 October. The date of the final exam is per the registrar. Contact us by Friday September 6th if you have a conflict with these dates.

Tools
We recommend, but do not require, the use of Eclipse for all programming assignments. However, assignments will be compiled, tested, and graded using the command-line tools available on the CIMS linux computers. If your code does not work correctly on these computers, you will not receive credit for that portion of the assignment. It is highly recommended that you test your code on one of these machines before submitting. We will request course accounts for you from the CIMS IT staff. Be sure to be registered for this class by Friday, September 6th at 4pm. Details about how to log into CIMS computers will be sent out once the accounts are set up. All submissions will be collected through NYUClasses.

Lateness policy
Homework is due on the day and time listed at the beginning of the assignment. Homework submitted after that time will be considered late. Late submissions will not be accepted except as described below.

We recognise that unexpected events may prevent you from completing your assignments on time. Each student will therefore be given five (5) slip days, which they can use at any point during the course. Each slip day excuses one 24 hour period of lateness on an assignment. For example, assume you have five remaining slip days, and an assignment due on Tuesday at midnight. Unfortunately, due to [Illness / computer crash / excess of cute cat pictures on the internet], you are unable to finish the assignment until Wednesday. You could use one of your spare slip days to submit the assignment late, leaving you with four for the remainder of the semester. To submit an assignment after the deadline, sacrificing a slip day, please email your assignment directly to the instructor:

- Subject: “NYU-CS-102 Assignment [n] Late Submission”
- Body: “Submission is [x] days late, leaving me with [y] slip days from my previous [z].”
- Attachment: Include your completed assignment as a zip file attachment.
Academic Integrity
You are expected to comply with NYU's academic integrity guidelines, as outlined on http://www.cs.nyu.edu/webapps/content/academic/undergrad/academic_integrity.
Collaboration in encouraged; however, all code that you write, and all problem sets that you submit, must represent your own work.

- **You must type in and edit your own code.**
  - Copying all or part of someone else's file is not allowed.
  - Allowing someone else to copy all or part of a file of yours, either explicitly or implicitly by leaving your code unprotected, is not allowed.
  - *Every keystroke you submit must have been done by you*
  - Pointing out a problem in someone else's code is OK; fixing it for them is not

- **You must credit your collaborators**
  At the beginning of each assignment:
  - List every person who helped you
  - Describe what they did.

- **Use your best judgement.**
  - Use judgement about asking or answering questions of other students. For example, if you are supposed to implement Algorithm X that is described in the book, and you don't understand Algorithm X, then you can ask another student to explain it to you. However, if you are supposed to come up with your own algorithm to solve a problem, try to help your colleagues to reach the answer without doing the work for them.
  - Remember that the purpose of the homework is to help you and your classmates learn. Students who spend little time on the homework invariably do poorly on exams and end up with a poor final grade.

Assignment 0 (3 marks, due Friday Sept 13 17:00)
1. Install Eclipse and ocaml as per http://www.seas.upenn.edu/~cis120/current/ocaml_setup.shtml
2. Read the following tutorials from http://ocaml.org/tutorials/. Test out the sample programs.
   a. Basics
   b. Structure of Ocaml programs
   c. Data Types and Matching
3. Write and execute a program which outputs the text “hello world”
   a. In Java
   b. In Ocaml
4. **Briefly** (1 paragraph) describe your computing background, what programming languages you have experience with, and one interesting thing about you.