CS 113 - Introduction to Computer Science I  
Course Syllabus, Fall 2013

New Format
The new format has lecture and lab components (recitations):
• One lecture plus one recitation per week.
• All students attend the same lecture class
• Students partitioned into smaller groups attend recitation

Lectures + Recitation

Lecture Class Meeting Time: Wednesday: 11:30 am – 12:55 pm  
Room: GITC 1400  
Instructor: Dr. Narain Gehani

There are 3 recitation sections attached to the above lecture class. You will be in one of them:

• Recitation Section 001 Meeting Time: Monday: 2:30 pm – 3:55 pm  
  Room: GITC 2315C (Dr. Michael Baltrush)

• Recitation Section 003 Meeting Time: Monday: 11:30 am – 12:55 pm  
  Room: GITC 2315C

• Recitation Section 005 Meeting Time: Friday: 1:00 pm– 2:25 pm  
  Room: GITC 2315C

Instructor Contact Information

• Instructor: Dr. Narain Gehani  
  Office: GITC 4304  
  Email: gehani@njit.edu  
  Office Hours: Wednesday: 1:00 pm – 2:30 pm.

TA Contact Information

• TA: TBA  
  Office: TBA  
  Email: TBA  
  Office Hours: TBA

Overview
This course is a comprehensive introduction to the Java programming language teaching writing, testing and debugging of programs. Topics covered fundamental Java object-oriented
programming. Topics covered are primitive data types, variables, assignments expressions and operators, control statements, recursion, design and use of classes, arrays, and I/O. Other topics covered are testing and debugging and writing programs that work reliably. The course guides students to the development of comprehensive Java applications.

**Textbook**


Please ignore the graphics portions at the end of each chapter. Graphics are not part of this course. Similar is the case with applets.

**Reference Material**

Java’s strength comes from the large number of libraries. Language is relative simple – but made complex from the large library facilities For details of Java library facilities, please refer to

```html
docs.oracle.com/javase/6/docs/api/overview-summary.html
```

**Programming Environment**

We will use jGRASP an integrated development environment (IDE) that supports Java, for program development. This environment is supported at NJITs lab classrooms. To acquire this environment for personal use, you can download it from

```html
www.jgrasp.org
```

You will also need to install Java development kit (JDK). Follow Java (JDK) download instructions on the jGRASP download page – go to Oracle website and download Java.

```html
www.oracle.com/technetwork/java/javase/downloads
```

You can also develop and run Java programs by accessing the Java compiler (`javac`) and interpreter (`java`) directly from the Windows Command Window. You need to edit Windows Environment variables PATH to point to the BIN directory in the Java software folder

**Prerequisites**

CS100 – Roadmap to Computing or equivalent

**Course Policies**

- Attendance is mandatory.
- Moodle (moodle.njit.edu/) will be used for course communication. Please keep checking Moodle.
- Homework assignments must be submitted in hard copy.
- Homework assignments will not be accepted late except for special circumstances (such as jury duty or medical problem), for which you must provide documentation.
• All submitted work (including exams) must include your name and student ID.
• Plagiarism will result in zero credit for the assignment and/or an XF grade in the course.
• Cell phones must be turned off during class.
• Students will be informed of any modifications of the syllabus during the semester.

Material covered

• Introduction to programming and Java programming language
• Data and Expressions
• Using Classes and Methods
• Decisions and Loops
• Arrays and I/O
• Objects-Oriented Programming
  a. Object-Oriented Design
  b. Defining Classes and creating Objects
  c. Defining methods
  d. Inheritance
• Testing and Debugging
• Exceptions
• Recursion

Goals

Upon completing this course, students will be able to specifically (in the context of Java):

• implement methods,
• design and implement classes,
• define objects and work with them,
• understand arrays and work with both one and two dimensional arrays,
• understand, properly design and use inheritance,
• understand exceptions and the different ways of handling them.
• understand definition of recursion and apply technique to simple problems,

Learning Outcomes

Objective

Teach students Java programming skills & concepts along with an emphasis on object-oriented programming via lectures, recitations (including hands-on software development in class), and assignments.

Outcome

Students will be able to work with a customer (someone who wants software/program to be developed) to understand the problem to be solved, design and write the (Java) program working by themselves or in a team.

Specifically, having determined the program to be written, students will be able to write Java programs that include programming concepts / Java facilities declarations, conditional and loop
statements, define classes, methods (including recursive methods), handle errors, and debug/test programs for correct behavior.

Students will also be prepared with Java programming skills (as above) for advanced courses.

**What do the Assignments Accomplish?**

By doing assignments,
1. a student will develop problem solving expertise
2. write Java programs putting to practice the programming concepts / Java facilities learned,
3. learn object-oriented programming, and
4. develop team working skills (in case of team projects)

Writing Java programs will involve the use of declarations, conditional and loop statements, define classes (including inheritance), methods (including recursive methods), handle errors, and debugging / testing programs for correct behavior.

**Performance**

Assignments and the exams aim to assess
1. Java programming skills in the context of the use of Java facilities (as mentioned above) to solve problems, and
2. understanding of programming concepts.

**Evaluation**

The evaluation will be based on the following course requirements:

- Attendance 5%
- Homework 30%
- Midterm Exam 32%
- Final Exam 33%

**Exam Policies**

You must bring a photo ID to all exams. Students with special needs are advised to make arrangements with Disability Services.

There are no makeup exams. If you miss a midterm because of a documented special circumstance you may receive an imputed grade based on the other midterm.

If you believe that you deserve more credit than you have been awarded on a particular exam problem, you may request, within 48 hours of the exam being returned, that it be regraded. Your entire exam will be regraded, which may result in points being added or subtracted.

Exams do not require any portable electronic devices, such as cell phones or calculators, and all such devices must be put away and turned off during the exam.

**University Code on Academic Integrity**
Read the University Code on Academic Integrity (njit.edu/academics/integrity.php). All work that you represent as your own must be your own. Work done by others must be given proper credit.

**Tentative Weekly Coverage of Material**

The following table shows approximately how much time may be devoted to each topic and the corresponding reading from the suggested textbook. **Actual class lectures may vary in pace and order.** Recitations will supplement lectures.

<table>
<thead>
<tr>
<th>Week (Approx)</th>
<th>Lecture</th>
<th>Recitation</th>
<th>Reading from book</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Introduction to programming and Java programming language</strong></td>
<td></td>
<td>Ch. 1 – focus on Java and program development</td>
</tr>
<tr>
<td></td>
<td>Programming Languages</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compiled vs. interpreted languages</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Getting Java and JGrasp</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compiling and running first program</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Java Program Structure – classes and objects</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The first java Program</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Writing reliable programs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Writing human readable programs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><strong>Nuts &amp; Bolts of Java programs</strong></td>
<td></td>
<td>Ch. 2</td>
</tr>
<tr>
<td></td>
<td>Identifiers</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Literals</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Types</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variables &amp; Constants</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Assignment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><strong>Nuts &amp; Bolts of Java programs ...</strong></td>
<td></td>
<td>Ch. 2</td>
</tr>
<tr>
<td></td>
<td>Expressions &amp; Operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><strong>Nuts &amp; Bolts of Java programs ...</strong></td>
<td></td>
<td>Ch. 5</td>
</tr>
<tr>
<td></td>
<td>Statements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><strong>Arrays</strong></td>
<td></td>
<td>Ch. 8</td>
</tr>
<tr>
<td>5</td>
<td><strong>Object-Oriented Programming</strong></td>
<td></td>
<td>Ch. 3, 4</td>
</tr>
<tr>
<td>5</td>
<td><strong>Printing output &amp; Wrapper Classes</strong></td>
<td></td>
<td>Ch 2, 3</td>
</tr>
<tr>
<td>6-7</td>
<td><strong>Defining Classes</strong></td>
<td></td>
<td>Ch. 4</td>
</tr>
<tr>
<td>7</td>
<td><strong>Finding + Swapping +</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week</td>
<td>Topic</td>
<td>Chapter(s)</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>7-8</td>
<td>More Conditionals &amp; Loops</td>
<td>Chapter 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Midterm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Object-Oriented Design</td>
<td>Ch. 7</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>More about classes &amp; objects</td>
<td>Ch 4, 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Static Variables/Methods</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Special Variable this</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reading Code –Rational Number class</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Interfaces</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Method Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Parameter Passing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Method Overloading</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Testing &amp; Debugging</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>More about Arrays</td>
<td>Ch. 8</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>More data structures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>More about classes – Inheritance, Abstract Classes</td>
<td>Ch 7 &amp; 8</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Recursion, Searching, Sorting</td>
<td>Chp 12, 10</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Exceptions</td>
<td>Ch. 11</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stepwise Refinement</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Time Permittin</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Even More Data Structures</td>
<td>Ch.13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Final Exam</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>