CSCI 121 — Course Information

Lectures: Professor Eric Roberts
E-mail: esroberts@reed.edu
Office: Greywood 117
Phone: 503-517-4883
Hours: Tuesdays, 9:00 A.M.–NOON
Wednesdays (on days when projects are due), 7:00–11:59 P.M.

Labs: Professor Mark Hopkins
E-mail: hopkinsm@reed.edu
Office: Library 382
Phone: 503-517-4735
Hours: Mondays 4:00 to 6:00 P.M.
Wednesdays 4:00 to 5:00 P.M.

Course description

CSCI 121: Computer Science Fundamentals I
Full course for one semester. An introduction to computer science, covering topics including elementary algorithms and data structures, functional and procedural abstraction, data abstraction, object orientation, logic, and the digital representations of numbers. Emphasis is on mathematical problems and calculations and on recursive algorithms and data structures. The course includes a significant programming laboratory component where students will solve computational problems using a high-level language. The mechanisms for processing and executing programs will be surveyed. Prerequisite: three years of high school mathematics. Lecture-laboratory.

Lectures

Lectures are scheduled for Monday, Wednesday, and Friday at 3:10 P.M. in PSYCH 105. The schedule of lectures appears on the last two pages of this handout. In addition to lecture, you must also sign up for a weekly 80-minute laboratory, led by Mark Hopkins.

Web page and software

The web page for CSCI 121 is https://people.reed.edu/~esroberts/csci121. All the materials and course announcements will be posted on this website. We recommend using a Python package called Anaconda, which includes an integrated development environment called Spyder, which you can download from the web site.

Course materials

The text for this class is a reader entitled Programming in Python, which is available in the Reed Bookstore for purchase at the cost of production. The reader is also available on the web site, but you will want to have a printed copy for exams, which are open book but disallow the use of online sources. We will distribute additional material in the form of class handouts.
Examinations

CSCI 121 will hold an in-class midterm examination on Friday, October 11, and a final examination from 9:00 A.M.–12:00 NOON on Wednesday, December 18, which is the time slot scheduled by the registrar.

Assignments and projects

CSCI 121 requires regular assignments that fall into two categories: problem sets and projects. Problem sets are shorter and focus on specific topics. Projects are larger and more integrative, ensuring that you can put different concepts together to create an interesting application.

Problem sets are graded using numeric scores, just as they are in most science and math classes. To emphasize a holistic assessment of the assignment as a whole, programming projects are graded on the following scale:

++ An absolutely fantastic submission of the sort that will only come along a few times during the semester, if at all.
+ A submission that exceeds our expectations. The program must reflect additional work beyond the requirements or get the job done in a particularly elegant way.
 ✓+ A submission that satisfies all the requirements for the assignment—a job well done.
 ✓ A submission that meets the requirements for the assignment, but which is either stylistically weak or has a few minor problems.
 ✓– A submission that falls significantly short of the requirements for the assignment.
 – A submission with still more serious problems but that nonetheless shows some effort and understanding.
 –– A submission that shows little effort and does not represent passing work.

Due dates and late days

All assignments—five problem sets and five projects—are due at 11:59 P.M. on Wednesdays. Assignments that come in after 11:59 will be considered late. The due dates for the assignments appear in the course calendar at the end of this handout.

Because each of you will probably come upon some time during the semester where so much work piles up that you need a little extra time, every student begins the semester with two free “late days,” where “day” is defined as a calendar day. Thus, if your assignment was due on Friday but turned in on Saturday, that assignment would be one day late. After your late days for the semester are exhausted, programs are assessed a late penalty of one category point per late day used (a ✓+ turns into a ✓, and so forth). Late days are valuable, and it pays to keep some around for the harder assignments toward the end of the term.

In special circumstances such as extended medical problems or other unforeseeable emergencies, extensions may be granted beyond the late days. To request an extension, send email to either Eric or Mark no later than 24 hours before the program is due.
Contests
In addition to the projects, we will also schedule two programming contests at different points during the term. The point of these contests is to give you a chance to show creativity and initiative beyond what is formally required by the course. Rules for each of the contests will be distributed in class when they are announced.

To encourage greater participation in the contests, we offer two additional incentives. First, every reasonably serious entry gets you one virtual ticket in a random drawing for a special grand prize at the end of the semester. The more contests you enter, the more chances you have. Winning runner-up prizes or honorable mentions in a contest give you additional chances. The random drawing will take place at the beginning of the review section for the final exam.

This semester, you have an additional opportunity to win virtual tickets. Although students last year found several errors in the reader as I was writing it, I expect that some mistakes slipped through the cracks. If you find an error in the reader, send it to esroberts@reed.edu. If you’re the first person to report a problem that turns out to be real, you get a ticket for the random drawing.

Grading
The most important component of the course in terms of learning is the problem sets and programming projects. To give you a strong incentive to complete them, it is important that assignments count for a significant proportion of the grade. At the same time, exams are necessary both to see whether you have retained mastery of the material and as a defense against the possibility that some of you will get too much help on the assignments, either appropriately from the teaching assistants or inappropriately by copying code from other students.

At most universities and colleges, computer science courses have had more than their fair share of cheating cases. Fortunately, violations of the Honor Principle were not a serious problem in CSCI 121 last year, which was for both of us our first year at Reed. We hope and expect that all of you will continue to maintain that same high level of academic integrity, but we will also continue to keep a close eye on the assignments. If we find a significant amount of cheating, we reserve the right to give more weight to the exams in the calculation of the final grade.
### Lecture schedule

<table>
<thead>
<tr>
<th>Monday</th>
<th>Wednesday</th>
<th>Friday</th>
</tr>
</thead>
</table>
| September 2  
Labor Day  
(no class) | 4  
Course overview  
Introducing Python | 6  
Arithmetic expressions  
Functions  
String data |
| 9  
Control statements  
Boolean data  
Algorithms | 11  
The Portable Graphics Library  
Simple graphics | 13  
Graphical applications  
Decomposition |
| **Read: Chapter 2** | **Read: Sections 3.1–3.4**  
**Due: Problem Set #1** | **Read: Sections 3.5–3.8** |
| 16  
The mechanics of functions  
Libraries and interfaces  
The `random.js` library | 18  
Interactive graphics  
Responding to mouse events | 20  
Timer-based animation |
| **Read: Sections 4.1–4.6** | **Read: Sections 5.1–5.4**  
**Due: Problem Set #2** | **Read: Section 5.5** |
| 23  
The `GObject` hierarchy | 25  
Recursive functions | 27  
Binary representation  
Representing characters |
| **Read: Section 5.6** | **Read: Section 4.7**  
**Due: Project #1 (Breakout)** | **Read: Section 6.1** |
| 30  
A simple machine model  
References and addresses | October 2  
Strings in Python  
Common string patterns | 4  
String applications  
The `english.py` library |
| **Read: Toddler handout**  
**Due: Graphics contest** | **Read: Sections 6.2–6.4**  
**Due: Problem Set #3** | **Read: Section 6.5** |
| 7  
Introduction to lists  
Operations on sequences | 9  
Debugging strategies | 11  
Midterm exam |
| **Read: Sections 7.1–7.2** | **Read: Sections 1.7–1.8**  
**Due: Project #2 (SpellingBee)** | 14  
The recursive paradigm  
The Towers of Hanoi |
| 16  
Searching and sorting  
Selection sort | **Read: Sections 8.1–8.2**  
**Due: Problem Set #4** | **Read: Sections 8.3–8.5** |
<table>
<thead>
<tr>
<th>Monday</th>
<th>Wednesday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>October 28</td>
<td>30</td>
<td>November 1</td>
</tr>
<tr>
<td>Using lists for tabulation</td>
<td>Multidimensional arrays</td>
<td>Records and tuples</td>
</tr>
<tr>
<td>List comprehensions</td>
<td>Image manipulation</td>
<td>Defining an Employee class</td>
</tr>
<tr>
<td><strong>Read:</strong> Sections 7.3–7.4</td>
<td><strong>Read:</strong> Sections 7.6–7.7</td>
<td><strong>Read:</strong> Sections 9.1–9.2</td>
</tr>
<tr>
<td><strong>Due:</strong> Problem Set #5</td>
<td><strong>Due:</strong> Problem Set #5</td>
<td><strong>Due:</strong> Problem Set #5</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>Rational numbers</td>
<td>Inheritance</td>
<td>Extending the graphics classes</td>
</tr>
<tr>
<td>Overloading operators</td>
<td>The Employee hierarchy</td>
<td>Data decomposition</td>
</tr>
<tr>
<td><strong>Read:</strong> Section 9.3</td>
<td><strong>Read:</strong> Sections 10.1–10.2</td>
<td><strong>Read:</strong> Sections 10.3–10.5</td>
</tr>
<tr>
<td><strong>Due:</strong> Project #3 (ImageShop)</td>
<td><strong>Due:</strong> Project #3 (ImageShop)</td>
<td><strong>Due:</strong> Project #3 (ImageShop)</td>
</tr>
<tr>
<td>11</td>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td>Cryptography</td>
<td>Dictionaries and maps</td>
<td>Sets in mathematics</td>
</tr>
<tr>
<td>The Enigma machine</td>
<td>Hashing</td>
<td>Python’s set class</td>
</tr>
<tr>
<td><strong>Read:</strong> Enigma handout</td>
<td><strong>Read:</strong> Sections 11.1–11.2</td>
<td><strong>Read:</strong> Section 11.3</td>
</tr>
<tr>
<td><strong>Due:</strong> Project #3 (ImageShop)</td>
<td><strong>Due:</strong> Problem Set #6</td>
<td><strong>Due:</strong> Section 11.3</td>
</tr>
<tr>
<td>18</td>
<td>20</td>
<td>22</td>
</tr>
<tr>
<td>Data files</td>
<td>Abstract data types</td>
<td>Data-driven programs</td>
</tr>
<tr>
<td><strong>Read:</strong> Section 7.5</td>
<td><strong>Read:</strong> Sections 12.1–12.3</td>
<td><strong>Read:</strong> Section 12.4</td>
</tr>
<tr>
<td><strong>Due:</strong> Problem Set #6</td>
<td><strong>Due:</strong> Project #4 (Enigma)</td>
<td><strong>Due:</strong> Section 12.4</td>
</tr>
<tr>
<td>25</td>
<td>27</td>
<td>29</td>
</tr>
<tr>
<td>Overview of Adventure!</td>
<td>Amazing algorithms</td>
<td>Thanksgiving break</td>
</tr>
<tr>
<td><strong>Read:</strong> Adventure handout</td>
<td><strong>Read:</strong> Enigma handout</td>
<td>(no class)</td>
</tr>
<tr>
<td>December 2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Linked lists</td>
<td>Recursive data models</td>
<td>Graphs</td>
</tr>
<tr>
<td>Stacks and queues</td>
<td>Tree structures</td>
<td>Graph algorithms</td>
</tr>
<tr>
<td><strong>Read:</strong> Enigma handout</td>
<td><strong>Read:</strong> Enigma handout</td>
<td><strong>Due:</strong> Enigma handout</td>
</tr>
<tr>
<td>9</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>Networking algorithms</td>
<td>Frontiers of computing</td>
<td><strong>Due:</strong> Project #5 (Adventure)</td>
</tr>
<tr>
<td>Google’s Page Rank</td>
<td><strong>Due:</strong> Adventure contest</td>
<td><strong>Due:</strong> Adventure contest</td>
</tr>
<tr>
<td><strong>Due:</strong> Project #5 (Adventure)</td>
<td><strong>Due:</strong> Project #5 (Adventure)</td>
<td><strong>Due:</strong> Project #5 (Adventure)</td>
</tr>
</tbody>
</table>