CS 100, Computer Literacy
Course Policies – Spring 2015

Course Objectives: At the end of this course, students will be able to:

- Explain the key components and basic workings of a typical computer system.
- Explain in simple terms the fundamentals of Computer Science covered in the core curriculum, such as the basic ideas behind data structures and operating systems.
- Explain in simple terms a number of advanced computer science areas, such as Computer Graphics, Virtual Reality, Human-Computer Interaction, Databases, Networking, Artificial Intelligence, Image Processing, Natural Language Processing, and Algorithm Analysis.
- Implement basic programming concepts such as sequential, branching, and repetitive execution, both in general and in specific using pseudocode, flowcharts, and the Alice computing system.
- Explain important events, breakthroughs, and people from the history of computing, including classic problems in computer science.
- Intelligently discuss current events and news items relating to Computer Science as they appear in everyday life.

Instructor: John Bell, jbell@uic.edu, 921 SEO, 413-9054
http://www.cs.uic.edu/~jbell
Office Hours: May be TTh 12:00-1:30, W 10:30-11:30
   - See schedule on web
   - Open Door policy during other times.

Teaching Assistant: Tanima Chatterjee, tchatt2@uic.edu
Office Hours: May be ??? in ???
   - See web site for details

Prerequisite: None
Note that credit for this course cannot be applied towards any degree offered by the departments of Computer Science or Electrical and Computer Engineering

Credits: 3

Course Web Page: http://www.cs.uic.edu/~i100

Textbooks:

Required:
- Wanda P. Dann, Stephen Cooper, and Randy Pausch, “Learning to Program with Alice”, Pearson Prentice Hall,
**Planned Schedule:**

A specific day-by-day schedule has not been precisely determined. However the course will consist of the following major components, (not necessarily of equal duration or importance, or in this order.)

1. General concepts in Computer Science
   - Understand the terminology of computers, programming, and computer science.
   - Hardware - What are all the parts of a computer, and what do each of them do?
2. History of Computer Science
   - Major events, breakthroughs, developments, and the people who made them.
3. CS Concepts I – The Core CS Curriculum
   - A brief survey of the basic concepts covered in a CS Bachelor’s degree, such as:
     - Data Structures
     - Algorithms and their Analysis
     - Software Design / Software Engineering
     - Operating Systems
4. Simple computer programming
   - Developing a logically ordered list of instructions to be followed in pseudo-code, and diagramming it with flowcharts.
   - Simple program development with Alice.
   - Concepts include variables, algorithms, decision making, looping, functions, and object-oriented development.
   - How to compile and run a given program in Java and/or C++. May include making some simple editing changes to existing programs, but not original development.
   - Web page and/or Android development?
5. CS Concepts II – Advanced Topics in Computer Science
   - A brief survey of basic concepts typically covered in graduate / optional courses, such as:
     - Computer Graphics
     - Virtual Reality
     - Computer-Human Interactions
     - Databases
     - Computer Networks
     - Natural Language Processing
   - A brief survey of current research in Computer Science, at UIC and elsewhere.
6. Computer Science in the News – As appropriate

**MIDTERM EXAM I - FRIDAY 13 FEBRUARY. In class, location TBD.**
**MIDTERM EXAM II - FRIDAY 20 MARCH. In class, location TBD.**
**FINAL EXAM - WEDNESDAY 6 MAY, 1:00 to 3:00 P.M.**

A more detailed agenda of future events and record of past events will be maintained on the course web site, and updated as the course progresses. These web pages may contain links to supporting materials, but should NOT be considered a replacement to attending class and taking your own notes.
**Academic Integrity:**

Students are encouraged to study together and to help each other learn. When one student teaches another, both benefit from the experience.

However, it is a strict violation of class and university policy for any student to hand in any work that is not 100% their own creation. Therefore:

- All work on all exams and all homeworks must be individually performed by the student whose name appears on the paper.
- No student may give any other student any portion of their code or any other homework, either written down, electronically, or through any other means.
- Students are responsible for safeguarding the integrity of their work. This includes but is not limited to changing their passwords and keeping their computer accounts secure.
- Direct copying of code from any textbook or other source is strictly forbidden.
- Students may discuss homework problems, including background concepts and general solution strategies, but they are forbidden from discussing or sharing specific solutions. In particular, it is forbidden for any student to show any other student any portion of their computer programs or homework solutions for any reason, including debugging assistance. This means you must hand in your own homework. You are not allowed to see anyone else's work, or show your work to anyone. Failure to protect the privacy of your work may be a violation.
- All submitted programs will be analyzed using MOSS, to identify any unacceptable similarity to other students’ code or to previous or published solutions if applicable.
- In the case of extreme discrepancy between homework performance and exam performance (e.g. very high homework scores and very low exam scores), the instructor shall determine which scores more accurately reflect the students' true work.
- All violations will be reported directly to the Office of Student Judicial Affairs, http://www.uic.edu/depts/sja. First violations will be penalized with zero on the relevant assignment(s) **and** a penalty equal to the value of the assignment(s), **for all parties involved in the transgression**. Second or more serious violations may result in a failing grade, probation, suspension, or expulsion from the university. The violation will also be recorded on the permanent records of all students involved.
Grading Policy:

Numerical scores will be based upon the following contributions:

( 3 ) Exams ( 2 midterm, 1 final ) 20 points each
Programming and homework assignments 10 points each
Quizzes 5 points each
Lab Score 20 points
Total: Normalized to 100 point scale

Unless otherwise specified, all programming & homework assignments will carry equal weight. The exact number of such assignments will be determined as the course progresses.

Conversion of numerical scores to letter grades is a serious business, requiring careful consideration of every student's complete semester performance, and will not be considered until all scores are compiled at the end of the semester.

There are no predetermined grade guarantees. However it is expected that grades will follow the general pattern given below. Regardless of the numerical score, it will not be possible to pass this course without passing the exams, particularly the final exam, and completing most of the homework assignments.

The grade break for: will probably be somewhere around:

A / B 90
B / C 80
C / ? 70

Note that the final grade breaks may be either slightly below or slightly above the numbers given here.

Exam Policy:

- All mid-term exams will be given at night, so that students will have ample time to complete the exam.
- Exams will be written so that the average student will be expected to finish in about an hour, so time constraints will not be a factor.
- Any exam conflict needs to be brought to the instructor's attention for resolution before the regularly scheduled exam. Requests for make-up exams after the regularly scheduled exam will not normally be granted.
- Exams will be closed-book. One note sheet may or may not be allowed, to be determined. If allowed, the note sheet will be limited to one 8 ½ x 11 inch page, double sided, handwritten.
- All exams will be cumulative, with emphasis on material which has not been covered on previous exams.
• All material covered in class or in assigned reading or which should have been learned in the course of completing homework is fair game on exams. No more specific information will be provided as to exam content.
• Anyone who fails to stop working on their exam when time is called will receive a minimum of a 5 point late penalty.

Homework Grading Policies

Specific homework grading guidelines will be determined on a case-by-case basis. For programming assignments, it is expected that the points will break down roughly as follows:

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program compiles and runs</td>
<td>25%</td>
</tr>
<tr>
<td>Program handles simple, straightforward situations:</td>
<td>25%</td>
</tr>
<tr>
<td>Program handles more advanced and/or tricky situations:</td>
<td>25%</td>
</tr>
<tr>
<td>Program is efficiently written using good programming style:</td>
<td>25%</td>
</tr>
</tbody>
</table>

Notes:
1. The first 50% of any assignment should be relatively easy to get. Anyone who hands in a program that compiles, is documented, and shows a reasonable attempt to complete the assignment should get at least half the points. Scores less than 50% are for incomplete assignments or work that just isn't worth much at all.
2. Scores from 50 to 90 % are based on quality and performance, with most scores expected to be in the 70 to 90 range.
3. The last 10% should be hard to get. Scores over 90 should only go to notably excellent papers, with scores of 100 going only to perfect error-free work.

Special Considerations

• All programs must be turned in using Blackboard.
• Programs must compile and run properly on the UIC computer system (as specified by the TA for each assignment) to receive full credit. Programs which run on other systems but which do not run on the UIC system may be given a zero, at the grader’s discretion.
• Graders are under no obligation to grade any program that does not compile. However, they may choose to do so at their own discretion.
• Some assignments may also require a “readme” file. The purpose of a readme file is to explain what the program does and how it does it, so that any reasonably competent programmer can easily understand it. It is in your best interest to make the grader's job as easy as possible by submitting a readme file that is both easy to read and which clarifies your code.
• Any assignments involving the creation of web pages, (which are by nature dynamic), must be printed out at the time they are due and the hard copy handed in.
• All appeals for grading errors, no matter how justified, must be submitted within two weeks after the graded assignments are returned. No appeals for regrades will be heard after that time.
• All students are allowed 4 late days during the course of the semester: The first two late days are at no charge. The third costs a 10% late penalty. The fourth a 20% late penalty.
  o Late penalties are applied as a multiplier of 0.9 or 0.8. So a submission that would be worth 70 points normally would be worth 63 with a 10% penalty applied, or a 56 with a 20% penalty applied.
  o A late day is a 24-hour grace period (from the original due date & time), and is not divisible.
  o No more than one late day may be used on any given assignment.
  o Late days may not be bought, sold, traded, or bartered for. Unused late days have no value.