Fundamentals of Computer Science

Course syllabus

This course introduces computer science through three of its major fields: hardware systems (physical components, digital logic, and computer architecture), theory and algorithms (Boolean algebra, binary arithmetic, and theory of computation), and software systems (languages, compilers, computer graphics, operating systems, and computer networks.) Programming assignments are used as means to introduce and reinforce fundamental computing concepts, as well as computer programming skills that are useful beyond this course. The course provides elements now essential to understand and effectively interact with the information technology infrastructure of today’s world.

<table>
<thead>
<tr>
<th>SESSION</th>
<th>TITLE</th>
<th>Topics</th>
<th>ACTIVITIES</th>
<th>OPTIONAL ACTIVITIES/TOPICS</th>
</tr>
</thead>
</table>
| 1       | Course Introduction | • Course review  
• General aspects  
• Class style and policies  
• Learning material review.  
• Introduction to CS  
• Problem solving in CS | ➢ Evaluation of preliminary knowledge in Computer Science and Programming  
➢ Administrative tasks  
➢ Reading: “What is CS” by Eric Suh. | ➢ Icebreaker  
➢ Group discussion: Learning expectations from the course |
| Lab     | Interaction with programming environment | • Python through examples  
• Creation of simple programs  
• Instruction(s) for data output | ➢ Testing the programming environment.  
➢ “Hello world”  
➢ Add two numbers.  
➢ “The Avatar” (Making figures with characters) | ➢ Instruction(s) for data input  
➢ Use of variables  
➢ Free hands-on activity |
| 2       | Introduction to computer programming | • Instructions for data output and data input.  
• Variables and lists  
• Operators and hierarchy  
• Boolean expressions  
• Characters and strings  
• Conditional and iterative statements  
• Random number generation  
• Libraries  
• User defined functions | ➢ Question prompt  
➢ “What is your name?”  
➢ “What is your age?”  
➢ “The Writer” (madlib)  
➢ Geometric figures generated by loops (diagonals, back diagonals, triangles, rectangles, perimeters)  
➢ “The Fortune Teller” | ➢ “99 bottles of root beer”  
➢ “12 days of Christmas”  
➢ “Rock, paper, scissors.”  
➢ Common algorithms: Fibonacci, factorial, largest number, average  
➢ Flowcharts |
| Lab     | Binary numbers | ➢ Introduction to ASCII code  
➢ Introduction to binary numbers  
➢ Programming converters between binary and decimal numbers. | ➢ Conversions between binary and decimal numbers  
➢ Encoded messages: Exercise on ASCII and | ➢ Hexadecimal $\leftrightarrow$ Binary |
<table>
<thead>
<tr>
<th>Session</th>
<th>Title</th>
<th>Topics</th>
<th>Activities</th>
<th>Optional Activities/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>binary numbers</td>
<td>➢ Bitmapped characters (decimal to binary exercises)</td>
<td>➢ Demonstration of analog audio: vinyl recording.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>➢ Bitmapped characters decoders.</td>
<td></td>
<td>➢ Binary representation of decimal numbers with fractional part. (Fixed point vs. floating point.)</td>
</tr>
<tr>
<td>3</td>
<td>Data representation</td>
<td>Representation of numbers</td>
<td>➢ Demonstration of image quality according to amount of bits per pixel.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Representation of symbols and strings</td>
<td>➢ Demonstration of analog signals vs. digital ones.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Representation of images</td>
<td>➢ Demonstration of audio quality according to amount of bits per sample.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Representation of audio</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lab</td>
<td>Data manipulation</td>
<td>➢ Bitmapped-image viewer</td>
<td>➢ Reading: The Greatest Mystery in Modern Science, by Chazelle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>➢ Experiments to modify media files to observe impact on quality.</td>
<td>➢ Image editor to modify bits per pixel.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>➢ Audio editor to modify sampling rate and bits per sample.</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Fundamental ideas • Boolean algebra</td>
<td>Fundamental ideas in CS</td>
<td>➢ Examples of fundamental ideas.</td>
<td>➢ Reading: The Rope and Pulley Wonder, by Dewney.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Boolean algebra</td>
<td>➢ Boolean algebra symbols and examples.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Truth tables</td>
<td>➢ Relationship between Boolean algebra and binary system.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Binary system: place value system, operations, conversions</td>
<td>➢ Logic gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Digital circuits</td>
<td>➢ Binary arithmetical and circuits for addition</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Binary circuits to perform arithmetic operations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Session</td>
<td>Title</td>
<td>Topics</td>
<td>Activities</td>
<td>Optional Activities/Topics</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------</td>
<td>--------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>Lab</td>
<td>Digital circuits simulation I</td>
<td>• Introduction to MMLLogic (digital circuit simulator)</td>
<td>Simulation exercises:</td>
<td>Subtraction and two’s complement</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>➢ MMLLogic experiments</td>
<td>➢ Video on integrated circuit making.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>➢ Testing logic gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>➢ Verifying Boolean laws</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>➢ Implementing binary adders</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Boolean algebra and logic circuits II</td>
<td>• Combinational logic</td>
<td>Logic gates and multiplexors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Circuits and truth tables</td>
<td>➢ Binary arithmetic: half adder, full adder, 3-bit adder.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Circuits for arithmetic</td>
<td>➢ Fundamentals of ALU design</td>
<td></td>
</tr>
<tr>
<td>Lab</td>
<td>Digital circuits simulation II</td>
<td>• Binary circuits to perform arithmetic operations</td>
<td>MMLLogic experiments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Review (Sunday session)</td>
<td>Movie analysis</td>
<td>• Computational elements in Disney’s “Tron”</td>
<td>CGI, operating systems, CPU, bugs, virus, antivirus, software, simulation, etc.</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Computer architecture</td>
<td>• Computer parts review</td>
<td>CARDIAC (educational computer model)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Computer architecture fundamentals</td>
<td>➢ Assembly language exercises.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• von Neumann model: Stored program concept, data-driven machine, etc.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lab</td>
<td>Computer architecture</td>
<td>• Implementing CARDIAC virtual machine</td>
<td>Absolute value</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Making assembly programs for CARDIAC virtual machine</td>
<td>Integer division</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sign function</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Add first N naturals</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Numeric-intensive computing I</td>
<td>• Common series</td>
<td>Exercises on numerical series</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Numerical approximation by Taylor series</td>
<td>➢ Approximation of irrational numbers</td>
<td></td>
</tr>
<tr>
<td>Session</td>
<td>Title</td>
<td>Topics</td>
<td>Activities</td>
<td>Optional Activities/Topics</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>--------</td>
<td>------------</td>
<td>---------------------------</td>
</tr>
</tbody>
</table>
| **Lab** |       | ▶ Implementation of numerical approximation through iterative structures | ➢ Harmonic series  
➢ Zeno’s dichotomy  
➢ Approximation of PI, e, etc | |
| 8       | Advanced topics I | • Library research and readings | ➢ WWW: WATCH by Sawyer (Chapter 1)  
➢ The Rope-And-Pulley Wonder by Dewney.  
➢ The Turing Omnibus by Feynman  
➢ Computer Organization by Dewney  
➢ Why programs fail by Zeller  
➢ Computer Networks by Kurose  
➢ Computer Architecture by Hennesy  
➢ Intelligent machines by Alan Turing  
➢ The Pattern on the Stone by Hillis  
➢ Great Ideas in Computer Science by Biermann  
➢ A.I. for Games by Fenge | ➢ How to Program by Computer by Dromney  
➢ The Tinkertoy Computer by Dewney.  
➢ The Best of Byte  
➢ Program Design with Pseudocode by Bailey  
➢ A Hitchhiker Guide to Virtual Reality by McMenemy  
➢ The Advent of the Algorithm by Berlinsky  
➢ Creative Evolutionary Systems by Bentley  
➢ Nine Algorithms that Changed the Future by MacCormick  
➢ Fundamentals of Natural Computing by Nunes  
➢ The Limits of Computing by Walker  
➢ Why Programs Fail by Zeller |
| **Lab** | Computer generated imagery (CGI) | • Computer graphics  
• Memory and CGI  
• CGI evolution | ➢ Movie analysis: Computer generated Imagery (CGI) in Pixar’s short films | ➢ Raster vs. vector graphics.  
➢ Mathematics of 2D images  
➢ 3D transformations  
➢ Raytracing |
| 9       | Numeric-intensive computing II (Graphics) | • Graphic mode,  
• Coordinate systems,  
• Plotting formulas | ➢ Plotting linear and quadratic equations  
➢ Parametric curves  
➢ Image display | |
| 10      | Advanced topics II | • Turtle geometry  
• Recursion  
• Recursive definitions  
• Fractal geometry  
• Introduction to formal grammars | ➢ Exercises with the Logo programming language.  
➢ Picture generation through formal grammars | ➢ Classic geometric exercises solved with turtle geometry  
➢ Implementing recursive functions in Logo  
➢ Generating fractal shapes through recursion (Snowflake, Sierpinski, etc.)  
➢ Generating grammar-based skylines (“Manhattan skylines”) |
| General Review (Sunday session) | Advanced topics III | • Elements of formal grammars  
• Elements of graph theory | ➢ From formal grammars to computer languages  
➢ Graph theory apply to maze solving |  |
<table>
<thead>
<tr>
<th>Session</th>
<th>Title</th>
<th>Topics</th>
<th>Activities</th>
<th>Optional Activities/Topics</th>
</tr>
</thead>
</table>
| 11      | Theoretical Computer Science | • Cryptography  
• Automata  
• Cellular automata  
• Pseudo-random number generation | ➢ Caesar’s cipher  
➢ Abstract notion of graphs, automata and finite state machine  
➢ Conway’s Game of Life | ➢ Arduino controlled mobile-robot(C-Language)  
➢ Game on MIT App Inventor for Android (visual language) |
|         | Final project (introduction and prototyping) | • Discussion of final projects  
• Brainstorming  
• Initial planning  
• Integration of programming elements required  
• Research topic | ➢ General requirements  
➢ Decision-based game: adventure  
➢ Virtual pet  
➢ Computer electronics and microcontrollers  
➢ Mobile computing applications  
➢ Games involving simulation or random numbers. | ➢ Examples on complexity. |
| 12      | Theory of computation | • Computability  
• Turing machine  
• Formal definition of algorithm  
• Algorithm complexity  
• Operating systems and networking | ➢ Simple computers equivalent to Turing machines  
➢ Machine language and compilers  
➢ Halting problem.  
➢ Examples on complexity. | ➢ Feedback |
|         | Final project development (Hands-on sessions) | • Implementing project major features  
• Documenting project | ➢ Feedback | ➢ Project review |
| 13      | Final project development (Hands-on sessions) | • Implementing project details  
• Create project context | ➢ Preparation of: presentation, webpage, report, etc. | ➢ |
| 14      | Final project conclusion | • Oral report | ➢ Final assessment of student’s progress | ➢ |
| 15      | Course conclusion | • Course review. | ➢ | ➢ |

**Notes:** Each entry provides a general description on the main topic to be studied and activities to be performed. **Topics** indicate specific points to be discussed during the session. In general, **session** refers to in-classroom meetings during the morning. **Labs** are activities to be held in a computer laboratory, and are aimed at promoting student-paced learning and opportunities for individual feedback. Evening session will give students additional time to complete lab work, and also promote analysis of topics presented during the day. Development of personal endeavors will be encouraged during this time.