CT 100 Computational Thinking
Fall 2018

Section: 01 9:55am – 10:50am, MWF Wing 6

Instructor: Dr. Mao Zheng
Wing 217, Tel: 785-6808, Email: mzheng@uwlnx.edu

Office Hour: 11:00am-11:50am on Mondays, Wednesdays
10:00am – 11:00am on Tuesdays, Thursdays
Or set an appointment by email

Course Website: http://www.cs.uwlax.edu/~zheng/CT100Fall18/

TextBook: “Computational Thinking for the Modern Problem Solver”, David D.

Grading Policies:
Exam 1 & 2 -- regular quizzes -- 25% each
Final Exam (cumulative) -- -- 30% Fri. Dec. 14@ 2:30pm
Labs, projects & quizzes -- -- 20%

Course Description:
Without computers most of the important discoveries of the last two decades
would not have been possible. Without computers our everyday lives would
be very different. The purpose of CT 100 is to dive into the minds of
computer scientists – to explore their tools, techniques and thought
processes and see how these are widely relevant for our world today.
Computational thinking is the application of algorithms and manipulation of
information in ways that can assist you in go further, regardless of your
chosen professional.

Attendance and Participation:
In-class activities and assignments are critical to be successful in this course.
You are expected to attend every class meeting, be attentive and participate
appropriately.

Late Labs/Projects, Missed Quizzes / Exams
Labs or projects are due on the dates indicated when they are given. For
extenuating circumstances that impact your ability to meet deadlines or
participate in class activities, you are responsible for alerting me as soon as
possible.

Lab or projects will not accepted after the due date, no make-up test or exam
will be given except for University-excused absences.
Grading Scale

<table>
<thead>
<tr>
<th>Letter Grade</th>
<th>Percentage Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>93-100%</td>
</tr>
<tr>
<td>AB</td>
<td>89-92%</td>
</tr>
<tr>
<td>B</td>
<td>83-88%</td>
</tr>
<tr>
<td>BC</td>
<td>79-82%</td>
</tr>
<tr>
<td>C</td>
<td>70-78%</td>
</tr>
<tr>
<td>D</td>
<td>60-69%</td>
</tr>
<tr>
<td>F</td>
<td>59% - below</td>
</tr>
</tbody>
</table>

Our Legal Obligations to You

Sexual Misconduct

As an employee of the University of Wisconsin-La Crosse, I am a mandated reporter of sexual harassment and sexual violence that takes place on campus or otherwise affects the campus community. This means that if I receive detailed or specific information about an incident such as the date, time, location, or identity of the people involved, I am obligated to share this with UWL’s Title IX Coordinator http://www.uwlax.edu/affirmative-action/ in order to enable the university to take appropriate action to ensure the safety and rights of all involved. For students not wishing to make an official report, there are confidential resources available to provide support and discuss the available options. The contact in Student Life is Ingrid Peterson, Violence Prevention Specialist, (608) 785-8062, ipeterson@uwlax.edu. Please see http://www.uwlax.edu/sexual-misconduct for more resources or to file a report.

Religious Accommodations

Per the UWL Undergraduate and Graduate Catalogs
http://catalog.uwlax.edu/undergraduate/aboutuwlax/#accommodation-religious-beliefs

“any student with a conflict between an academic requirement and any religious observance must be given an alternative means of meeting the academic requirement. The student must notify the instructor within the first three weeks of class (within the first week of summer session and short courses) of specific days/dates for which the student will request an accommodation. Instructors may schedule a make-up examination or other academic requirement before or after the regularly scheduled examination or other academic requirement.”
Students with Disabilities

Any student with a documented disability (e.g. ADHD, Autism Spectrum Disorder, Acquired Brain Injury, PTSD, Physical, Sensory, Psychological, or Learning Disability) who needs to arrange academic accommodations must contact The ACCESS Center (165 Murphy Library, 608-785-6900, ACCESSCenter@uw lax.edu) and meet with an advisor to register and develop an accommodation plan. In addition to registering with The ACCESS Center, it is the student’s responsibility to discuss their academic needs with their instructors.

You can find out more about services available to students with disabilities at The ACCESS Center website: [http://www.uwlax.edu/access-center](http://www.uwlax.edu/access-center)

Veterans and Active Military Personnel

Veterans and active military personnel with special circumstances (e.g., upcoming deployments, drill requirements, disabilities) are welcome and encouraged to communicate these, in advance if possible, to me. For additional information and assistance, contact the Veterans Services Office. [http://www.uwlax.edu/veteran-services/](http://www.uwlax.edu/veteran-services/). Students who need to withdraw from class or from the university due to military orders should be aware of the military duty withdrawal policy [http://catalog.uwlax.edu/undergraduate/academicpolicies/withdrawal/#military-duty-withdrawal-university](http://catalog.uwlax.edu/undergraduate/academicpolicies/withdrawal/#military-duty-withdrawal-university).
Learning Outcomes:

By the end of this course, you will be able to:

- **General Computation**
  - be familiar with the history of the modern digital computer from the 1950’s
  - be able to apply Moore’s law as a measure of computing advances
  - understand the limitations and advantages of digital devices by comparison to analog
  - appreciate the computer as a repository of data
  - understand that data is represented as sequences of bits
  - know the relative sizes of byte, kilobyte, megabyte, terabyte and petabyte
  - be able to translate decimal numbers to binary and binary to decimal
  - understand that data has alternative forms (numeric, textual, code, image, sound)
  - understand that different data forms use different, largely incompatible encodings
  - be familiar with the concept of data compression and the difference between lossy and nonlossy compression
  - understand different forms of computer storage and the relative size and performance differences between memory and secondary storage
  - understand the process of program compilation and execution (source code and executables)

- **Logic**
  - understand that one bit of data may also store a Boolean value (true or false) be able to analyze a Boolean expression involving the operations of NOT, AND, OR, IMPLIES and EQUIVALENCE
  - be able to express factual English sentence in the form of propositional logic
  - be familiar with logical expression applications in expressing software requirements be able to express database queries at the level of propositional logic

- **Problem Solving**
  - understand the nature of software development as a problem solving activity
  - be able to apply divide and conquer strategies to basic problems
  - be able to develop story boarding for solving real world scenarios
• understand the problem solving approach of prototyping and be able to apply prototyping
• to implementing simple sequential algorithms o be able to translate simple story boards into algorithms involving objects and a solution as a sequence of method calls

• Control
  o understand the concept of software and program execution
  o be able to write simple programs involving sequences of methods applied to different objects
  o understand that algorithms involve choices and choices take the form of selections involving logical conditions
  o understand that algorithms often involve repetition understand how algorithms are modularized

• Models of Computation
  o know the definitions of basic digraph terminology, including nodes, vertices, arcs, cycles, walks, and paths. understand the basics of using graphical abstractions as models of computation know the form and function of flow chart elements for imperative statements, selection and repetition be able to express simple algorithms using flow charts
  o understand the concepts of computational state, events and actions be able to model simple, sequence algorithms of ten or less states

• Organizing Data
  o be able to select meaningful names and understand why this is importance for such things as URLs, files, variables, and so forth.
  o understand the role of unique naming and the concept of case sensitivity be able to craft path names for files organized into file trees
  o understand the concept of direct access and indexed data retrieval
  o understand the concept of sequential, linked, data retrieval know the difference between linear and non-linear data organizations understand the concept of data organized in tree structures and applications in classification and analysis. understand the concept of 2-dimensional data layout and tabular information retrieval

• Code is Data / Data is code
  o understand the von Neumann principle that computers can be used to store both data and code
be able to craft simple spread sheet
understand the concept of 2-dimensional data layout and tabular information retrieval
understand the relationship of discrete functions and tables
appreciate the role of patterns in software development and the use of software for pattern matching of strings of Computation

• Performance and Limits of Computation
recognize that performance can be characterized by account of operations that are performed
understand the distinct between best case and worst case analysis and the importance of the latter
understand the limitations of benchmarking vs. analytic analysis
to be able recognize the difference between tractable (polynomial) and brute force (exponential)
understand that many security mechanisms rely upon algorithmic performance recognize that not all things are actually computable (Halting problem)

• Verification and Validation
to understand that software correctness only makes sense with respect to specifications to be able to analyze a small programming problem, the outcome of which is a set of software requirements
to understand that proving program correctness is sometimes a possibility, but not often a reality.
to understand that software correctness is usually accomplished by way of software tests to realize the limitations of software testing to be able to craft test cases using simple boundary condition analysis

• Concurrency
to understand that software correctness only makes sense with respect to specifications to understand the distinction between sequential and concurrent execution.
to learn how sorting can be accomplished concurrently with assorting network.
to understand the problem of data integrity in a concurrent environment and some common solutions to data corruption avoidance.
to understand the potential for dead lock and strategies for avoiding deadlock.
Security

- to understand the basic tenants of security - integrity, confidentiality, and availability. to realize that complete mitigation is rarely possible, but that a certain measure of security
- is possible using mechanisms for deterrence, deflection and detection/recovery. To know basic security terminology, including vulnerability, asset, security system, mitigation, virus, spoofing, spam, sniffing, DOS and CERT.
- to understand the proper use of security mechanisms including anti-virus software, passwords, encryption, file backup and firewalls. to understand the potential or dead lock and strategies for avoiding deadlock.
CT 100 Course Outline

What is CT? (Chapter 1)
- History of computation
- Computational Thinking

Data (Chapter 2)
- Information and data
- Data encoding (binary & decimal)
- File organization and web browsing
- Intro to programming

Logic (Chapter 3)
- English semantics and logic
- Logical expressions
- Decisions in algorithm

Modeling (Chapter 6)
- Directed graphs
- Activity diagrams
- State diagrams

Problem Solving (Chapter 4 & 5)
- Divide and conquer strategies
- Problem solving
- Programming

Data Organization (Chapter 7)
- Indexing
- Linking

Spreadsheets – a tool for 2D programming (Chapter 8)

String Programming
- Strings
- Regular expression pattern matching

Performance (Chapter 10)
- Benchmarks
- Counting analysis
- Algorithms that are impractical or non-computable
Concurrency (Chapter 11)
  • Sorting networks
  • Conflicts and deadlock

Security (Chapter 12)