



Course Design Document

IS103 Computational Thinking (CT)

Version 1.0

10 October 2011

“Computational thinking confronts the riddle of machine intelligence: What can humans do better than computers? What can computers do better than humans? Most fundamentally it addresses the question: What is computable?”

Prof. Jeannette Wing, Carnegie Mellon University

Table of Content

1. Versions History	4
2. Overview	5
Synopsis	5
Prerequisites.....	5
Objectives	5
Basic Modules	5
Instructional Staff.....	5
3. Output and Assessment Summary	6
(1) Class Participation (10%)	6
(2) 1 Mid-Term Exam (20%)	6
(3) 3 Assignments (30%).....	6
(4) 1 Final Exam (40%).....	6
4. Group Allocation for Assignments	6
5. Classroom Planning	6
5.1 Course Schedule Summary	6
5.2 Weekly Plan.....	6
6. Resources	7
7. Tooling	7
8. Learning Outcomes	7

1. Versions History

Version	Change Description	Author	Date
1.0	Initial release for use in 2011 Term 2	Lau Hoong Chuin	10-Oct-2011

2. Overview

Synopsis

Computational Thinking equips students to tackle complex computational problems; it trains students to design solutions to solve those problems using a computer program. It draws upon concepts from mathematics and computer science – more precisely, discrete mathematics, data structures and algorithm design.

This course will hone students' analytical skills as they are challenged to think abstractly and computationally. Their minds will be open to the wonders of computing, as they go behind the scene to unravel the fundamental analytics that empower Google, consulting agencies and service companies.

Prerequisites

NIL

Objectives

By taking this course, the student will:

- discover the science of computing (or *How to think like a Computer Scientist*)
- model problems and learn practical problem-solving techniques to tackle complex computational problems (beyond what a spreadsheet is capable of solving)
- apply problem-solving techniques to develop more elegant and efficient programs
- learn to write programs to represent and manipulate with complex data objects
- understand the challenge of **scale**, not only in dealing with large data sets, but also in appreciating the nature of computing and computability

Basic Modules

Building Blocks in Discrete Mathematics	3 weeks
Fundamental Data Structures	4 weeks
Introduction to Algorithm Design and Analysis	4 weeks

Instructional Staff

- Course Coordinator: Assoc. Prof. Lau Hoong Chuin (hclau@smu.edu.sg)
- Instructors: Lau Hoong Chuin, supported by Mok Heng Ngee (hnmok@smu.edu.sg)
- TA: Lindawati (lindawati.2008@phdis.smu.edu.sg)

3. Output and Assessment Summary

- (1) Class Participation (10%)
- (2) 1 Mid-Term Exam (20%)
- (3) 3 Assignments (40%)
- (4) 1 Final Exam (30%)

4. Group Allocation for Assignments

Assignment No	How groups are formed?	No of Students in a group
Assignment 1	Individual	1
Assignment 2	Individual	1
Assignment 3	Team	3

5. Classroom Planning

There is one 3-hour classroom session per week, divided into two sessions with a 15 minutes break in between.

5.1 Course Schedule Summary

Wk	Theme	Events
1	What makes a computer scientist different from a biologist, a lawyer, or an economist?	
2	Of pins and balls – counting and computing	
3	Logic as foundation of thinking and computing	Due: Assignment 1
4	The elegance and power of recursion in computing	
5	How does a computer represent lists and sequences?	
6	How does a computer represent hierarchy and structure?	Due: Assignment 2
7	Modeling social and enterprise networks on a computer	
8	Recess Break	
9	MIDTERM EXAM Introduction to the world of computational complexity	Mid-term Exam
10	How to search efficiently using a computer?	
11	Exploiting the power of decomposition and parallelism	
12	Exploiting the power of heuristic reasoning	Due: Assignment 3
13	Reasoning about a highly-connected cyber and physical world	
15	FINAL EXAM	Final Exam

5.2 Weekly Plan (see Section 6 on acronyms)

Wk	Topic	Reference
1	Introduction , Abstract Data Types	PC, Chp 4
2	Discrete Mathematics I (Counting)	SDB, Chp 1.1-1.4
3	Discrete Mathematics II (Logic and Proof)	SDB, Chp 3

4	Discrete Mathematics III (Recursion and Recurrence)	SDB, Chp 4.1-4.2 PC, Chp 3.1-3.2
5	Data Structures I (List, Stack and Queue)	PC, Chp 5, 7, 8
6	Data Structures II (Trees)	PC, Chp 11
7	Data Structures III (Graphs and Networks)	PC, Chp 14.1-14.2
8	Recess Break	
9	MIDTERM EXAM (2 hours) + Introduction to Computational Complexity	PC, Chp 10.1
10	Algorithm Design I (Searching, Recursion)	SDB, Chp 4.3 PC, Chp 3.3-3.5, 6.1
11	Algorithm Design II (Sorting, Divide and Conquer)	PC, Chp 10.2
12	PROJECT PRESENTATION (1.5 hours) Algorithm Design III (Greedy Algorithms)	Suppl. Materials
13	Algorithm Design IV (Graph Algorithms)	PC, Chp 14.3-14.4
15	FINAL EXAM	

6. Resources

- Course Package
 - [SDB] Stein, Drysdale and Bogart, Discrete Mathematics for Computer Scientists, Pearson, 2010 (Chapters 1.1-1.4, 3, 4.1-4.3)
 - [PC] Prichard and Carrano, Data Abstraction and Problem Solving with Java: Walls and Mirrors, 3/E, 2011 (Chapters 3-8,10-11,14)
- Supplemental materials
 - To be provided
- Interesting web links:
 - CMU Centre for Computational Thinking <http://www.cs.cmu.edu/~CompThink/> and CT course <http://www.andrew.cmu.edu/course/15-122/index1.html>
 - Dartmouth College CT course: <http://www.cs.dartmouth.edu/~cs3/index.html>
 - Google on CT <http://www.google.com/edu/computational-thinking/>
 - Computer Science for Fun - <http://www.cs4fn.org/fundamentals/index.php>

7. Tooling

The course is mostly conceptual so that it can be available to students without programming background. However, students will be exposed to tools (such as JCF: Java Collections Framework and JDSL: Java Data Structures Library). There will be in class demos and hands-on sessions of these tools.

8. Learning Outcomes

	IS103– Computational Thinking		Course-specific core competencies which address the Outcomes	Faculty Methods to Assess Outcomes
1	Integration of business & technology in a sector context			
	1.1 Business IT value linkage skills			
	1.2 Cost and benefits analysis skills			
	1.3 Business software solution impact analysis skills			

2	IT architecture, design and development skills			
	2.1 System requirements specification skills	YY	<ul style="list-style-type: none"> Translate problems into computational problems formally 	Grade Assignments, Mid-Term and Final Exams and Class Participation
	2.2 Software and IT architecture analysis and design skills	YY	<ul style="list-style-type: none"> Identify and design appropriate representation and solution to solve computational problems Explain issue with scale and computational efficiency, space-time tradeoff 	
	2.3 Implementation skills	Y	<ul style="list-style-type: none"> Translate representation and solution design into Java code (by tweak existing Java code) 	
	2.4 Technology application skills	YY	Acquire the following skills: <ul style="list-style-type: none"> recursive thinking abstraction and decomposition heuristic reasoning design with simplicity and elegance 	
3	Project management skills			
	3.1 Scope management skills			
	3.2 Risks management skills			
	3.3 Project integration and time management skills			
	3.4 Configuration management skills			
	3.5 Quality management skills			
4	Learning to learn skills			
	4.1 Search skills	Y	<ul style="list-style-type: none"> Independently investigate other problem-solving techniques not taught/discussed in class (through supplementary readings) Self-learn the use of software tools (JDSL, Java Collections) and use them for in class exercises and project 	Grade Class participation and Project
	4.2 Skills for developing a methodology for learning			
5	Collaboration (or team) skills:			
	5.1 Skills to improve the effectiveness of group processes and work products			
6	Change management skills for enterprise systems			
	6.1 Skills to diagnose business changes			
	6.2 Skills to implement and sustain business changes			
7	Skills for working across countries, cultures and borders			
	7.1 Cross-national awareness skills			
	7.2 Business across countries facilitation skills			
8	Communication skills			
	8.1 Presentation skills	Y	<ul style="list-style-type: none"> Prepare and deliver effective technical presentations, particularly on problem definition and solution design in formal yet intuitive fashion, explain algorithm design concisely and convincingly 	Grade and give feedback to participation in in-class exercises Grade Team Project presentation
	8.2 Writing skills	YY	<ul style="list-style-type: none"> Express problem definition, data structures, algorithm design formally 	Grade Assignments and Team Project report

Y This sub-skill is covered partially by the course

YY This sub-skill is a main focus for this course