Academic Year	AY1819		**Semester**	2	
Author(s)	Li Fang (asfli@ntu.edu.sg) Lin Shang-Wei (shang-wei.lin@ntu.edu.sg) Nicholas Vun (aschvun@ntu.edu.sg) Adopted for MA1008 by Lee Yong Tsui (mytlee@ntu.edu.sg) Sameer Alam (sameeralam@ntu.edu.sg)				
Course Code	MA1008				
Course Title	Introduction to Computational Thinking				
Pre-requisites	NIL				
Pre-requisite for					
No of AUs	3				
Contact Hours	Lectures/ TEL (Online Videos)	13	Example Class (Seminars/Hands-on Exercises)	26	
Proposal Date	7 January 2019				

Course Aims
Computational thinking (CT) is a problem solving process with the aid of computer; i.e. formulating a problem and expressing its solution in such a way that a computer can effectively carry it out. It includes a number of characteristics, such as breaking a problem into small and repetitive ordered steps, logically ordering and analyzing data and creating solutions that can be effectively implemented as algorithms running on computer. As such, computational thinking is essential not only to the Computer Science discipline, it can also be used to support problem solving across all disciplines, including math, science, engineering, business, finance and humanities. The aim of this course is hence to take students with no prior experience of thinking in a computational manner to a point where they can derive simple algorithms and code the programs to solve some basic problems in their domain of studies. In addition, the course will include topics to appreciate the internal operations of a processor, and raise awareness of the socio-ethical issues arising from the pervasiveness of computing technology.
Intended Learning Outcomes (ILO)

Upon the successful completion of this course, you shall be able to:

1. Describe the internal operation of a basic processor, how a program is executed by a computer and computing trends.
2. Code basic programs based on the programming language used in the course
3. Formulate a problem and express its solution in such a way that a computer can effectively carry it out. (i.e. equip you with CT skills)
4. Apply the CT concepts on case studies/problem-based scenarios through hands-on practice of the CT processes.

Course Contents

	Topics	LAMS (Hours)	Example Class (2-hr session)
0	**Course Overview and Concepts of Computational Thinking** Solving complex problem using computer **-** enables the student to work out exactly what to tell the computer to do.	0.5	Familiarization with Python programming environment (1 week)
1	**Overview of Programming Languages** High level programming languages (Python, C, Java)	0.5	
2	**Basic internal operation of computer** Basic computer organization (Processor, Memory, I/O) and how a computer execute a program (Machine instructions)	1	Simple Python programs (1 week)
3	**Basic program structure: control constructs and data types** Concepts of data types, variables; Pseude code and flowcharts; Sequences, Selection (if/else), iteration (for/while loop);	3	Basic data types and control constructs (2 weeks)
4	**CT concept - Abstraction** Problem formulation - reducing something to a very simple set of characteristics to only focusing on the most relevant to the problem. Concept of functions/libraries, file and exception handling and data structure	2	Data Structures, Library functions, user-defined functions (3 weeks)
5	**CT concept - Decomposition** Break a complex problem into smaller and more manageable parts/steps, such that each of these smaller problems can then be looked at individually	1	More on functions, file & exception handling (1 week)
6	**CT concept – Pattern recognition** Looking for similarities among and within problems, which also enable re-use knowledge of previous similar problems	1	Project – planning (1 week)

7	**CT concept – Algorithm** Reformulating the problem into series of ordered steps through Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources. (Some common/useful examples: Sorting and searching)	2	Project – Coding (1 week)
8	**Limit of computing** Analysis of Algorithm Complexity to determine how much resources (space and time) are needed to execute an Algorithm in order to achieve code optimization.	1	Project – Coding (1 week)
9	**Computing Trends** Cloud, Edge and Fog computing, Quantum Computers	0.5	Project – coding and Submission (1 Week)
10	**Social-Ethical Issues and Ramifications of Computing** Fairness, Privacy, Sharing, Hacking, software Piracy, Data Protection, Cyberbullying and trolling, Fake news, digital divides, IP/Copyright	0.5	
		13	26

Assessment (includes both continuous and summative assessment)

Component	Course LO Tested	Related Programme LO or Graduate Attributes	Weightage	Team/Individual	Assessment Rubrics
1.TEL MCQs	1,2,3,4,	a, b, c, d	5%	Individual	See appendix 1
2.Written quizzes (2)	2,3,4,	a, b, c	35%	Individual	See appendix 1
3.Hands-on exercises & assessment (Working Program & 2 programming Tests)	2,3,4,	a, b, c, d, e, l	30%	Individual	See appendix 2
4.Mini Project assessment	2,3,4,	a, b, c, d, e, i, j, l	30%	Individual	See appendix 3
Total			100%		

Mapping of Course SLOs to EAB Graduate Attributes

Course Student Learning Outcomes	Cat	EAB's 12 Graduate Attributes*												EAB's ME/AE requirements	
		(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	(k)	(l)	ME	AE
MA1008 Introduction to Computational Thinking	Core	●	●	●	●	◐				◐	○		◐	●	●
Overall Statement	This course aims to take students with no prior experience of thinking in a computational manner to a point where they can derive simple algorithms and code the programs to solve some basic problems in their domain of studies. In addition, the course will include topics to appreciate the internal operations of a processor, and raise awareness of the socio-ethical issues arising from the pervasiveness of computing technology													Apply to both ME and AE programmes	
1. Describe the internal operation of a basic processor, how a program is executed by a computer and computing trends.	a, b, c, d														
2. Code basic programs based on the programming language used in the course	a, b, c, d, e, i														
3. Formulate a problem and express its solution in such a way that a computer can effectively carry it out. (i.e. equip you with CT skills)	a, b, c, d, e, i, j, l														
4. Apply the CT concepts on case studies/problem-based scenarios.	a, b, c, d, e, l														

Legend:

●	Fully consistent (contributes to more than 75% of Student Learning Outcomes)
◐	Partially consistent (contributes to about 50% of Student Learning Outcomes)
○	Weakly consistent (contributes to about 25% of Student Learning Outcomes)
Blank	Not related to Student Learning Outcomes

*The graduate attributes as stipulated by the EAB, are:

(a) **Engineering knowledge**: Apply the knowledge of mathematics, natural science, engineering fundamentals, and an engineering specialisation to the solution of complex engineering problems.

(b) **Problem Analysis**: Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

(c) **Design/development of Solution**s: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.

(d) **Investigation**: Conduct investigations of complex problems using research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

(e) **Modern Tool Usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

(f) **The engineer and Society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

(g) **Environment and Sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for the sustainable development.

(h) **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

(i) **Individual and Team Work**: Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

(j) **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

(k) **Project Management and Finance**: Demonstrate knowledge and understanding of the engineering and management principles and economic decision-making, and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

(l) **Life-long Learning**: Recognise the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Formative feedback

For online tasks, immediately after you submitted the answers, you will see your scores, your answers, the correct answers, feedback on your incorrect answers, and explanations for the correct answers. For online and laboratory quizzes, individual feedback will be provided to students through evaluation of their submissions. Quiz answers will be discussed in the example class. you will also see the average scores of the other students in the same cohort.

For lab assessments, you will be given verbal feedbacks during your demonstrations of the circuits.

Learning and Teaching approach

Approach	How does this approach support students in achieving the learning outcomes?
TEL (Online Video)	Topics will be delivered as a series of online videos lectures, and you will also be provided reference reading materials for self-study to achieve LO 1 to LO 4
Example Class (Face-to-face)	The Example class will be used as seminar sessions for students to clarify the contents of the online topic, as well as hands-on sessions to equip students with practical knowledge on coding, and on the design and implementation of a mini project to achieve LO 1 to LO 4.

Reading and References

The computational thinking aspect will not use any specific text book. The following books and websites will be used as reference materials. But for Python programming, Item 1 is the textbook.

1. The Practice of Computing using Python; William Punch and Richard Enbody, Pearson, 2017.
2. Introduction to Computation and Programming Using Python : With Application to Understanding Data; (2nd Ed) John V. Guttag, MIT Press Ltd, 2016.
3. https://edu.google.com/resources/programs/exploring-computational-thinking/

Course Policies and Student Responsibilities

As a student of the course, you are required to abide by both the University Code of Conduct and the Student Code of Conduct. The Codes provide information on the responsibilities of all NTU students, as well as examples of misconduct and details about how students can report suspected misconduct. The university also has the Student Mental Health Policy. The Policy states the University's commitment to providing a supportive environment for the holistic development of students, including the improvement of mental health and wellbeing. These policies and codes concerning students can be found in the following link.

http://www.ntu.edu.sg/SAO/Pages/Policies-concerning-students.aspx

Academic Integrity

Good academic work depends on honesty and ethical behavior. The quality of your work as a student relies on adhering to the principles of academic integrity and to the NTU Honor Code, a set of values shared by the whole university community. Truth, Trust and Justice are at the core of NTU's shared values.

As a student, it is important that you recognize your responsibilities in understanding and applying the principles of academic integrity in all the work you do at NTU. Not knowing what is involved in maintaining academic integrity does not excuse academic dishonesty. You need to actively equip yourself with strategies to avoid all forms of academic dishonesty, including plagiarism, academic fraud, and collusion and cheating. If you are uncertain of the definitions of any of these terms, you should go to the academic integrity website for more information. Consult your instructor(s) if you need any clarification about the requirements of academic integrity in the course.

Course Instructors

Instructor	Office Location	Phone	Email
Lee Yong Tsui	N3.2-02-87	67905493	mytlee@ntu.edu.sg
Sameer Alam	N3.2-02-17	67906906	sameeralam@ntu.edu.sg

LAMS Instructors

LI FANG			asfli@ntu.edu.sg
LIN SHANG WEI			shang-wei.lin@ntu.edu.sg
NICHOLAS VUN			aschvun@ntu.edu.sg

Planned Weekly Schedule

Week	Topic	Course LO	Readings	Tutorial Class Activities
1	Computational Thinking Concepts, Programming languages	3,4	On-line Video	Familiarisation with the Python programming environment
2	Basic internal operation of computer	1	On-line Video.	Simple Python exercises
3	Basic program structure: Case Study, Pseudo code and	2	On-line Video.	Programming with expressions and

	flowchart,			data types
4	Basic program structure: Data type, Variable, sequence, logic and comparison operation	2	On-line Video.	Programming with conditional control
5	Basic program structure: Selection and Iteration	2	On-line Video	Programming using loops; Quiz 1
6	Procedural abstraction: function and library	2,3	On-line Video	data structures, list string & dictionary
7	Data abstraction: Data structure	2,3	On-line Video.	Programming with functions; Programming assessment 1
8	Decomposition Case study	2,3,4	On-line Video	More programming with functions
9	Pattern recognition Case study	2,3,4	On-line Video	Programming with files and exceptions; Quiz 2
10	Algorithms Sorting algorithm	2,3,4	On-line Video	Mini project planning
11	Algorithm design Searching algorithm	2,3,4	On-line Video	Mini project coding; Programming assessment 2
12	Algorithm Complexity Analysis, Big-0 concept	2,3,4	On-line Video	Mini project coding
13	Computing trends and Ethical considerations	1	On-line Video	Mini project coding and submission