COURSE OUTLINE: PS0001

Course Title	**Introduction to Computational Thinking**		
Course Code	**PS0001**		
Offered	Study Year 1, Semester 1		
Course Coordinator	Thomas Peyrin (Assoc Prof)	thomas.peyrin@ntu.edu.sg	6513 2027
Pre-requisites	None		
Mutually exclusive	BG2211, CE1003, CE1103, CZ1003, CZ1103, MA1008, MS1008		
AU	3		
Contact hours	Tutorials: 39, Technology-enhanced Learning: 13		
For delivery from	AY 2023/24 semester 1		
Last revised	29 May 2023		

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 you from having no prior experience of thinking in a computational manner to a point where you can derive simple algorithms and code the programs to solve some basic problems in mathematics and science in general. 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

Upon successfully completing this course, you should 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 Python programming language.
3. Formulate a problem and express its solution in such a way that a computer can effectively carry it out.
4. Apply the CT concepts to case studies/problem-based scenarios.

Course Content

Computational Thinking Concepts, Programming languages

Basic internal operation of computer

Basic program structure: Case Study, Pseudo code and flowchart

Basic program structure: Data type, Variable, sequence, logic and comparison operation

Basic program structure: Selection and Iteration

Procedural abstraction: function and library

Data abstraction: Data structure

Decomposition Case study

Pattern recognition Case study

Algorithms Sorting algorithm

Algorithm design Searching algorithm

Algorithm Complexity Analysis Big-0 concept

Computing trends and Ethical considerations

Assessment

Component	Course ILOs tested	SPMS-MAS Graduate Attributes tested	Weighting	Team / Individual	Assessment Rubrics
Continuous Assessment					
Seminar					
Project	2, 3, 4	**2.** a, b, c **3.** a, b	25	both	See Appendix for rubric
Technology-enhanced Learning					
TEL participations and TEL MCQs	1, 2, 3, 4	**1.** a, b, c, d	20	individual	See Appendix for rubric
Multiple Choice Questions	1, 2, 3, 4	**1.** a, b, c, d **4.** a **5.** a	25	individual	
Mid-semester Quiz					
Programming test	2, 3, 4	**1.** a, c, d **2.** b	30	individual	See Appendix for rubric
		Total	**100%**		

These are the relevant SPMS-MAS Graduate Attributes.

1. Competence

a. Independently process and interpret mathematical theories and methodologies, and apply them to solve problems

b. Formulate mathematical statements precisely using rigorous mathematical language

c. Discover patterns by abstraction from examples

d. Use computer technology to solve problems, and to communicate mathematical ideas

2. Creativity

a. Critically assess the applicability of mathematical tools in the workplace

b. Build on the connection between subfields of mathematics to tackle new problems

c. Develop new applications of existing techniques

3. Communication

a. Present mathematics ideas logically and coherently at the appropriate level for the intended audience

b. Work in teams on complicated projects that require applications of mathematics, and communicate the results verbally and in written form

4. Civic-mindedness

a. Develop and communicate mathematical ideas and concepts relevant in everyday life for the benefits of society

5. Character

a. Act in socially responsible and ethical ways in line with the societal expectations of a mathematics professional, particularly in relation to analysis of data, computer security, numerical computations and algorithms

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 quizzes and laboratory quizzes, individual feedback will be provided to you through evaluation of your 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 the programming test, immediately after you submitted the answers, you will see your scores. General feedback will be provided to the students.

For project assessment, you will be given verbal feedbacks during your demonstrations of the program.

Learning and Teaching Approach

Tutorials (39 hours)	The Example class will be used as seminar sessions for you to clarify the contents of the online topic, as tutorial sessions to work on algorithm exercises, as well as hands-on sessions to equip you with practical knowledge on coding, and on the design and implementation of a mini project to achieve LO 1 to LO 4.
Technology-enhanced Learning (13 hours)	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

Reading and References

The course will not use any specific text book. The following books and websites will be used as reference materials.
1. The Practice of Computing using Python; William Punch and Richard Enbody, Pearson, 2017. ISBN: 9780134380315
2. Introduction to Computation and Programming Using Python : With Application to Understanding Data; (2nd Ed) John V. Guttag, MIT Press Ltd, 2016. ISBN: 0262529629
3. Learning Python (5th Ed), Mark Lutz, O'Reilly Media, 2013. ISBN: 1449355730
4. 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 you 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.
https://www.ntu.edu.sg/life-at-ntu/student-life/student-conduct

The programming test is a compulsory component of this course. Students who miss the test without valid reasons will receive a fail grade ("F") for the course.

If you are sick or have compassionate grounds, and unable to attend the programming test, you must:

1. Send an email to the instructor regarding the absence.
2. Submit the Medical Certificate* or death certificate to your home school.

* The Medical Certificate mentioned above should be issued in Singapore by a medical practitioner registered with the Singapore Medical Association. In this case, a make-up programming test will be arranged.

Please note that only ONE make-up programming test will be scheduled for this course, and the date will be announced later. In the event that you have a valid reason (approved medical leave/ short leave) for missing the make-up test, you may be eligible for an incomplete ("I") grade for the course if you can provide relevant supporting documents that are accepted by the school. Otherwise, a fail grade ("F") will be awarded for the course.

Academic Integrity

Good academic work depends on honesty and ethical behaviour. The quality of your work as a student relies on adhering to the principles of academic integrity and to the NTU Honour 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, 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
Thomas Peyrin (Assoc Prof)	SPMS-MAS-05-14	6513 2027	thomas.peyrin@ntu.edu.sg

Planned Weekly Schedule

Week	Topic	Course ILO	Readings/ Activities
1	Computational Thinking Concepts, Programming languages	3, 4	On-line Video Familiarization with Raspberry Pi (RPi) board, and Scratch Programming
2	Basic internal operation of computer	1	On-line Video IDE for Python on RPi
3	Basic program structure: Case Study, Pseudo code and flowchart,	2	On-line Video Python programming exercises
4	Basic program structure: Data type, Variable, sequence, logic and comparison operation	2	On-line Video Python programming exercises
5	Basic program structure: Selection and Iteration	2	On-line Video Python programming exercises
6	Procedural abstraction: function and library	2, 3	On-line Video Python programming exercises (Function)
7	Data abstraction: Data structure	2, 3	On-line Video Python programming exercises (Function)
8	Decomposition Case study	2, 3, 4	On-line Video Python Programming Exercises + Mini project – Flow Chart Design
9	Pattern recognition Case study	2, 3, 4	On-line Video Python Programming Exercises + Mini project – Coding

10	Algorithms Sorting algorithm	2, 3, 4	On-line Video Python Programming Exercises + Mini project – Coding and debugging
11	Algorithm design Searching algorithm	2, 3, 4	On-line Video Python Programming Exercises + Mini project – Coding and debugging
12	Algorithm Complexity Analysis Big-0 concept	2, 3, 4	On-line Video Python Programming Exercises + Mini project – Testing and assessment - Programming test
13	Computing trends and Ethical considerations	1	On-line Video Mini project – assessment

Appendix 1: Assessment Rubrics

Rubric for Seminar: Project (25%)

You will demonstrate 1 working program in the form of a mini project. The maximum score is 30.

Criteria	**Standards**		
	Fail standard (0-39%)	**Pass standard (40-80 %)**	**High standard (81-100 %)**
Demonstrate (including explanation) the use of CT concepts in the implementation of the project. (LO 2,3,4)	Demonstrated less than 40% of the functionalities according to the specifications.	Demonstrated 40% to 80% of the functionalities according to the specifications.	Demonstrated more than 80% of the functionalities according to the specifications.

Rubric for Technology-enhanced Learning: TEL participations and TEL MCQs (20%)

You will complete online MCQ directly testing your understanding of the course content (counting for 20% of the final grade)

Rubric for Mid-semester Quiz: Programming test (30%)

You will complete 1 programming test on the course contents (counting for 30% of the final grade). The maximum score is 30. This component is compulsory in order to pass the course.