Computational Thinking – Coding Education: Supplement to the Primary Curriculum

Prepared by

the Curriculum Development Council

Recommended for use in schools by

The Education Bureau

HKSARG

2020

Contents

Chapter 1 Introduction

The "Computational Thinking — Coding Education: Supplement to the Primary Curriculum (Draft)" is published in November 2017 by the Education Bureau (EDB). This supplementary document aims to provide a systematic and detailed description of computational thinking and coding education, as well as a list of relevant learning elements, to facilitate the teaching in P4 to P6. The Curriculum Development Council (CDC) recommended schools to implement coding education on a school-based basis and develop the necessary knowledge, skills, and attitudes among students to face the growing digital economy era in the future.

Since the announcement of this supplementary document, the EDB has collected the schools' views on the implementation of the supplementary document through school visits, questionnaires for teachers of professional development programmes and school survey. The Committee on Technology Education (TE) under the CDC set up the Ad Hoc Committee in February 2019 to review and revise the content of the supplementary document based on the feedback gathered. The learning elements of coding education at upper primary level are explained in a clearer and more focused manner in the revised supplementary document.

The focus of coding education is to enable students to master the coding skills and apply the skills to different contexts to complete assigned tasks / jobs. The learning contents of the supplementary documents have generally been covered in the Computer Awareness Programme (CAP) developed by the EDB. The EDB will keep on enriching the CAP to assist teachers facilitating students to grasp the basic programming skills and cultivate the ability of computation thinking. Computational thinking and coding is an approach to solving problems which can be used in different key stages and in a variety of disciplines. Most regulations in the games, sports and music activities have already involved the concepts of sequence and condition in computational thinking. Schools can provide daily-life experiences at different key stages to expose students on learning the related concepts, so as to inspire students' learning of computational thinking and coding education at a later stage.

We recommend schools to adopt "The Computational Thinking — Coding Education: Supplement to the Primary Curriculum*"* in the curriculum planning and implement coding education systemically to cultivate students' computational thinking.

Views and suggestions on this supplementary document are welcome. These may be sent to:

Chief Curriculum Development Officer (Technology Education)
Curriculum Development Institute
Education Bureau
Room W101, 1/F, West Block
Education Bureau Kowloon Tong Education Services Centre
19 Suffolk Road
Kowloon Tong, Hong Kong
Fax: 2768 8664
E-mail: teched@edb.gov.hk

Chapter 2 Objectives

Students will be able to:

- Understand the basic computational thinking concepts and practices, including abstraction, algorithm, and automation.
- Possess the ability to develop programs and process data to solve problem.
- Understand the process of problem solving and limitations of coding.
- Connect coding with real-life problems and other subjects.
- Solve problems through communication and collaboration with teamwork effectively in the process.

The goal of implementing computational thinking and coding education at upper primary level is not to train and nurture computer programmers. Instead, it is to give students hands-on experience and enable them to solve problems with confidence, as well as to solve problems through collaboration and repeated trials.

Chapter 3 Learning Elements (Applicable to Key Stage 2)

It is recommended that students should learn the following content which aims to develop computational thinking and coding skills among upper primary students. Schools can make adaptation to cater for their needs.

Learning Elements	Learning Contents
● Abstraction	
◆ Express the algorithm / pattern with diagram / table	• Recognise the pattern • Use diagrams or tables to model an algorithm and a pattern
◆ Modularization	• Recognise the concept of modularization • Know how to decompose the problem into sub-problems
● Algorithm	
◆ Problem Solving Procedures	• Identify the phases of problem solving including problem identification, problem analysis, algorithm design and programming , etc.
◆ Concepts and Practices of Basic Programming Constructs	
■ Sequence Concept	• Understand that the instructions are carried out one after another as a sequence in an algorithm
■ Branching/Selection Concept	• Understand that branching / selection is a single or multiple “selection” or “decision” • Use the branching / selection structure(s) to compose single and multiple branched commands in an algorithm • Use comparison operators (>, < and =) and logical operators (AND, OR, NOT)

Learning Elements	Learning Contents
■ Iteration Concept	• Understand that “iteration” is the repetition of process in programming • Understand repeated commands can be simplified by “iteration” and ending condition needs to be set in an iteration
◆ Data Processing	• Understand how data is collected and analysed
■ Variable Concept	• Be aware of using “variable” and their usages in a program • Create program instructions for storing and modifying the value of “variable”
◆ Program Development / Coding Concept and Practices	
■ Understand the commands of programming tools	• Understand the usage of commands
■ Design, reuse, and remix programs/codes	• Design and compose the procedures of solving problem • Reuse program / codes or modify the commands or parameters of the existing program / codes to solve problem
■ Testing and Debugging	• Develop appropriate steps and data to test programs • Fix the errors of program
● Automation	
◆ Stored Program Concept	• Describe the application of automation in daily life • Understand the importance of stored program in automation
● Interacting with Physical Objects	• Control physical objects with the Integrated Development Environment

Learning Elements	Learning Contents
◆ Forming a system connected with physical objects	• Using sensors and actuators to interact with the environment • Be aware of the development of sensors and embedded system • Be aware of the control and operation of remote monitoring system

Chapter 4 Implementation

At present, many schools have allocated one lesson per week at each level offering school-based Information Technology programme to teach CAP or related contents. Schools should plan the curriculum in a systematic manner according to their circumstance to implement coding education in the lessons for all upper primary students.

Arrangement of lesson periods

Schools should allocate lesson time flexibly for conducting related learning and teaching to cater for the needs of students. For instance, they may make use of the "flexible time" to allocate one lesson (35/40 minutes per lesson) per week at each upper primary level or other feasible arrangement to cover learning contents of "Computational Thinking — Coding Education" supplementary document and CAP. Schools are recommended to allocate about 10 to 14 hours at each level to teach the contents of this document. Two modes of implementation are provided for schools' reference as below.

- **To implement through school-based programme:** Schools can flexibly design their implementation schedule to cover related contents, such as "block scheduling" offering the programme in one term with a double lesson per week, instead of one lesson per week for the whole academic year. With the longer lesson time arrangement, it will reduce time being spent on transition between lessons, allowing students to spend more time on completing the learning task.

- **To implement through theme-based teaching:** Schools can adopt theme-based teaching in cross curricular manner, to focus learning on a particular theme covering the learning content and skills of different subjects. This teaching approach integrates different contents, creating a learning target. Based on students' interests and life experiences, their knowledge, skills, and attitudes can be developed in a more meaningful way.

Teachers are encouraged to use the time flexibly to help students achieve the learning objectives in computational thinking through providing various learning experiences inside and outside the classroom.

Chapter 5 Learning and Teaching

Learning computational thinking and coding is a complex, multi-faceted, active and interactive process. Apart from traditional lecturing approach, teachers should infuse active learning elements into the classroom activities. Teachers should adopt diversified teaching strategies to facilitate students to acquire the knowledge, concepts and skills that this supplement encompasses, such as the ability of problem-solving, creativity, communication, as well as to enhance the capacity of learning to learn. Therefore, not only should teachers possess certain understanding of computational thinking and coding, but they should also master the related pedagogies, knowledge and skills.

Guiding Principles

The following outlines the basic rationale and guiding principles for learning and teaching:

- **Knowledge:** Knowledge exists in different forms and contexts. Some knowledge is established while some is dynamically changing and contextualised. In order to be useful, all knowledge has to be constructed by learners in an active manner.
- **Learning:** Learning can take place in different ways. Knowledge can be acquired from teacher guidance and reading literatures. It can also be learnt through different activities such as self-reflection, as well as collaboration and interaction with others.
- **Clear learning targets:** Each learning activity should be designed with learning targets which are clear to both teachers and students.
- **Teaching for understanding:** The pedagogies chosen should aim at enabling students to understand what they are learning rather than just to memorise the content.
- **Building on prior knowledge and experience:** The design of learning activities should be built on the prior knowledge and experience of students.
- **Using appropriate pedagogies:** A diversified learning and teaching approaches and activities should be designed for different learning objectives and to cater for students' various learning styles, so that effective learning can be achieved.
- **Promoting interaction:** Student can explore what they know and don't know through interactive activities. Teachers should use open-ended questions that stimulate students' thinking and encourage them to share their views, so that students can learn from each other.
- **Promoting independent learning:** The generic skills and reflective thinking skill of students can be nurtured through learning activities. Teachers should encourage students to take

responsibility for their own learning.

- **Using formative assessment:** To improve learning and teaching effectively by making good use of formative assessments. Assessment activities should be designed to collect information on student performance to facilitate feedback on learning and teaching.
- **Effective use of resources:** Various types of teaching resources should be used effectively as learning tools. Teachers can make reference to the learning and teaching resources[1] provided by the EDB to facilitate student learning in coding education.
- **Enhancing motivation:** Effectiveness of learning could be enhanced when students are motivated to learn. Teachers should adopt appropriate motivation strategies to arouse students' interest in learning.
- **Maximising engagement:** The active engagement of students is very important when conducting learning activities.
- **Embracing learner diversity:** Learners have different characteristics and abilities. Teachers should adopt various strategies to cater for learner diversity. For example, establish a learning community in which learners of varied ability can support each other's learning.

[1] https://www.edb.gov.hk/en/curriculum-development/4-key-tasks/it-for-interactive-learning/modular-computer-awareness-programme/index.html#9

https://stem.edb.hkedcity.net/en/computational-thinking-coding-education/

Chapter 6 Assessment

Guiding Principles

Assessment aims at collecting students' evidences of learning to enable various stakeholders, including students, teachers, schools and parents, etc., understand the student learning progress.

When developing assessment strategies for the contents related to the TE Key Learning Areas, the following guiding principles may be considered by schools:

- The learning of TE is purposeful and holistic. Assessment should be closely aligned with the related learning elements and should be able to reflect the major components of learning including knowledge, concepts, processes, awareness, generic skills, values and attitudes.
- The purpose of assessment is to ensure students can acquire adequate knowledge and skills and make steady progress. It is important in technology learning that students should observe safety and health measures when they are using tools and equipment.
- Assessment should be infused in the learning process. In general, formative assessment (i.e. assessment for obtaining feedback on learning and teaching) and summative assessment (i.e. assessing student performance) are equally important in both enhancing student learning and charting students' learning progress. Teachers may use the assessment methods, such as classroom participation, homework and project assignment, to understand student performance and learning progress for the teachers' continual reflection and feedback on learning and teaching, so as to enhance student learning.
- It is particularly important that everyone involved in the assessment process, including teachers, students and parents, understand and know how to make use of the assessment results, and are able to formulate the way forward for student learning according to the assessment results.

The assessment objectives should be closely aligned with the learning elements described in the previous chapters. It is expected that students are able to:

- abstract a simple problem in order to design a solution
- understand and use the basic programming constructs / algorithm to solve problem, and be able to see the related constructs in daily life
- anticipate how changing program statements/codes can change the operation / output(s) of the program
- familiarise with the reuse and remix of program/codes, and be able to test and fix bugs in the program
- identify patterns in program and be able to apply them in new situations

Glossary

Term	Description	Page
Computational Thinking (CT)	"Computational Thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006). Students become tool builders instead of tool users through a set of CT concepts and practices such as abstraction, algorithm and automation. Computational thinking is a problem solving methodology that can be transferred and applied in different contexts. Computational thinkers can accurately describe the problems and construct an algorithm that solves it.	3 - 10
Computer Awareness Programme (CAP)	As early as 2000, the Education Bureau has developed the Computer Awareness Programme (CAP) for primary schools to help students master computer knowledge and basic programming skills. The content of the programme was updated and enriched in 2015 and 2019, by including the latest tools such as Scratch, App Inventor, Arduino and micro:bit to enhance the learning and teaching of programming.	3, 9
Coding	Throughout this supplement document, the terms 'coding' and 'programming' are considered to be identical in meaning and are used in a broad sense to refer to a process that leads from a formulation of a problem (computing problem) to an executable program (computer program).	3 - 11

Reference

Wing, J. M. (2006). Computational thinking. ***Communications of the ACM***, 49(3), 33-35.

Membership of the Ad Hoc Committee for Reviewing the "Computational Thinking – Coding Education: Supplement to the Primary Curriculum"

Chairperson	Mr YEUNG Wai-yin PLK Dr Jimmy WONG Chi-Ho (Tin Sum Valley) Primary School
Members	Mr AU Kin-keung Hong Kong Taoist Association Wun Tsuen School
	Dr. CHEUNG Chak-chung, Ray Department of Electrical Engineering The City University of Hong Kong
	Prof. KONG Siu-cheung Department of Mathematics and Information Technology The Education University of Hong Kong
	Mr KWOK Hung-I, Henry SKH Lui Ming Choi Memorial Primary School
	Mr KWOK Man-chiu Christian & Missionary Alliance Suen Ki Primary School
	Mr LEE Kim-wah Po Chiu Catholic Secondary School
	Mr LI Andy PLK Dr Jimmy WONG Chi-Ho (Tin Sum Valley) Primary School
	Ms LO Kit-yan The Education University of Hong Kong Jockey Club Primary School
Ex-officio Members	Mr CHENG Wing-cheung Technology Education Section Curriculum Development Institute

Dr. LAM Wai-lim, William
Kindergarten and Primary Section
Curriculum Development Institute

Ms. LEE Yuk-ching, Yuki
Technology Education Section
Curriculum Development Institute

Mr. WU Man-wai, David
Technology Education Section
Curriculum Development Institute

Secretary Mr LUI Kam-ming, Michael
Technology Education Section
Curriculum Development Institute