Realizing Computational Thinking in the Mathematics Classroom: Bridging the Theory-Practice Gap

Weng Kin Ho[*†] wengkin.ho@nie.edu.sg
Chee Kit Looi[*] cheekit.looi@nie.edu.sg
Wendy Huang[*] wendy.huang@nie.edu.sg
Peter Seow[*] peter.seow@nie.edu.sg
Longkai Wu[*] longkai.wu@nie.edu.sg
National Institute of Education
Nanyang Technological University
Singapore

15 August 2019

Abstract

With the impact of computer technology on human civilization in the 21st century, Computational Thinking has become an important topic of research and discussion in the area of education over various non-computer science disciplines; for example, mathematics, sciences, languages, social sciences, etc. By now, Computational Thinking has often been referred to as a paradigm of knowledge and problem solving by which the problems and their solutions can be implemented using an effective means, e.g., by a computer, though it never really possess a universal definition. In this paper, we invite the reader to revisit the original meaning of the word Computational Thinking as intended in 1980 by its inventor, Seymour Papert. By so doing, we bring into focus the concrete issues that a mathematics teacher need to consider so as to realize Computational Thinking in his or her lessons. We analyze the design principles put forth earlier by the authors, and understand these principles in the context of mathematics teachers professional development and classroom implementation.

1 OECD bids us to relook at mathematics education

As to what mathematical competence is viewed, different countries in the world hold different stands and have various visions. These differences give rise to different ways in which education

[*]This work is supported by funding OER 10/18 LCK for the project "How to bring Computational Thinking (CT) into Mathemtiacs classrooms: Designing for disciplinary-specific CT".

[†]Corresponding author

systems are organized around the objective of teaching mathematics at schools so that the future generations acquire some set level of mathematical competence. Mainstream education systems in the past had always recognized mathematical competence to include basic mastery of the four arithmetic operations, manipulations of fractions, decimals and percentages, and mensuration of simple geometric shapes. However, in the present dominance of technology in our civilization, the use of data in all aspects of life has become indispensable: education, career-planning, health and financial management, medicine, data analytic, climatology, international economics, etc. Because of the ubiquity of data in life, we can only expect that the next generation not only be digitally literate but also digitally creative and competent to solve the future problems! Because of the worldwide need to bring up the digitally competent 'next generation', many countries have worked collaboratively towards defining the future of education.

The Organization for Economic Cooperation and Development is a special forum where the governments of 34 different economies work with one another, together with another 70 non-member economies to promote economic growth, prosperity and sustainable development. One particular area of importance and interest to the OECD is that of education. The Programme for International Student Assessment (PISA) is a worldwide study by OECD in member and non-member economies to evaluate educational systems by measuring 15-year-old students' scholastic performance on mathematics, science and reading. The first PISA was performed in 2000, and then repeats every three years. PISA provides comparable data so as to enable participating countries to improve their education policies and outcomes.

With regards to mathematics education, two questions are of particular interest to policy makers and teachers alike. The first question is:

Question 1 *What mathematics do students need learn mathematics, and which students need to learn mathematics? [10, p. 4]*

To this question, the most frequently given answer is to have all students learn mathematics by appealing to the versatility mathematics in practical situations. Interestingly, OECD in drafting the new PISA 2021 mathematics framework, remarks that

> *"this argument alone gets weaker with time – a lot of simple activities have been automated"* [10, p. 4],

and bids us to relook at the aforementioned question more seriously. The paradox lies in the statement that advancement in computer technology has unexpectedly become an impediment to the learning of mathematics – the very discipline that fuelled it! Perceiving mathematics to be merely a useful toolbox is far too restrictive, often confining teachers and learners to a rigid list of mathematical topics and procedures, and is deemed to be too narrow for the needs of today.

The second question is:

Question 2 *Are some mathematics teaching methods more effective than others? ([9, p. 18])*

This question concerns the teaching practice, i.e., exactly what do mathematics teachers employ to engage students in deep learning.

It is of central interest to the ATCM community to explore novel, meaningful and effective ways by which mathematics can be taught to students with the affordance of computer technology. The authors thus expects that this conference community would be interested in

hearing out our answers to the above questions. To answer these questions, we tap further into the wisdom of certain OECD's publications on PISA, e.g., [9, 10].

With regards to the first question, we start with OECD's official definition of mathematical literacy in PISA 2021 – this new definition has expanded the scope of the older versions:

> *Mathematical literacy is an individual's capacity to reason mathematically and to formulate, employ, and interpret mathematics to solve problems in a variety of real world contexts. It includes concepts, procedures, facts and tools to describe, explain and predict phenomena. It assists individuals to know the role that mathematics plays in the world and to make well-founded judgements and decisions needed by constructive, engaged and reflective 21st century citizens.* [10, p. 7]

There are two emphases: (1) the individual's capacity to reason mathematically, and to engage in the processes of problem solving, and (2) the 21st century citizens' responsibility to apply the mathematics that they learnt. Here, reasoning refers to the use of mathematics content knowledge to recognise the mathematical nature of a situation or problem, especially in real world contexts, and to formulate it using mathematical terms. Additionally, reasoning is involved in the selection of mathematical tools and how they are applied to solve the problem.

As for the second question, an answer comes from the comparison of the PISA 2021 framework with those of PISA 2003 and PISA 2012. We note that PISA 2021 recognises the movement of the current world trends towards a world that is constantly driven by new technologies in which citizens are creative and engaged. Because of the growing role of technology in students' lives, PISA 2021 acknowledges that the "long-term trajectory of mathematical literacy should also encompass the synergistic and reciprocal relationship between *mathematical thinking* and *computational thinking*" ([10, p. 7]). Thus, PISA 2021 has included a new expectation that students ought to "possess and be able to demonstrate computational thinking skills that include pattern recognition, decomposition, determining which (if any) computing tools could be employed in the analysing or solving the problem, and defining algorithms as part of a detailed solution". [10, pp. 8–9]

Additionally, [9] raises teachers' attention to the importance of teaching strategies that "give students a chance to think deeply about problems, discuss methods and mistakes with others, and reflect on their own learning". This class of teaching strategies is termed as *cognitive activation*, which is typified by the processes of summarising, questioning and predicting as students are engaged in problem solving. Importantly, cognitive activation strategies stresses on the practice of engaging students in genuine problem solving over an extended period of time – this is when students are encouraged to think and reason, the new emphasis highlighted in the PISA 2021 Mathematics Framework.

The first thing we gather from the PISA documents is that skills related to *computational thinking* will be of paramount importance in the mathematical training of the future generation, especially in view of the technology-driven world we live in – this is the theoretical aspect. The second thing we learn from the PISA documents is that the means by which mathematics can be taught must be effective – this is the practical aspect. This then leads us to the natural question of how we, as 21st century mathematics educators, can bridge this theory-practice gap by realizing computational thinking in the mathematics classroom as an effective means of teaching mathematics.

In this paper, we set out to bridge this theory-practice gap. In Section 2, we make clear what we mean (or not mean) by computational thinking (at least, as far as this paper is concerned).

Next, in Section 3, we lay down four design principles that can guide teachers in designing mathematics lessons that incorporate computational thinking – these lessons, we call them "Math + C"[1] lessons. We then argue that cognitive activation is encouraged in these Math + C lessons that are constructed from our proposed design principles. To further convince the reader of the practicality of these principles, we manufacture some sample Math + C lessons in Section 4. Lastly, in Section 5, we end with some remarks for recommendations for future study and considerations.

2 Rethink computational thinking

While the new PISA 2021 draft posits computational thinking skills as an essential skill set to be mastered by the 21st century students, it has never put in its own official definition of computational thinking as it does for the definition of mathematical literacy. Indeed, the draft document simply adopts Wing's definition ([15]) of computational thinking as the somewhat universally accepted version (see [10, p. 7]). Specifically, Wing's version of computational thinking refers to "the way computer scientists think" and is regarded as a thought process entailed in formulating problems and designing their solutions in a form that can be executed by a computer, a human, or a combination of both ([16]).

In this paper, we do not go the easy way of taking the lock, stock and barrel of Wing's interpretation of computational thinking. Instead, we first revisit Papert's original formulation of computational thinking, and then proceed to retain those parts of Wing's interpretation that is applicable to our present context.

Returning to [12], Papert originally intended that "the goal is to use computational thinking to forge ideas". In other words, computational thinking is a paradigm which is intended to "change patterns of access to knowledge" ([11]). This justifies that computational thinking, like any other existing ways of thinking, should be taught to everyone – the sooner the better.

Computational thinking, the way Wing perceives, possesses distinctive features and dispositions and it encompasses four fundamental components: *Decomposition*, *Pattern recognition*, *Abstraction*, *Algorithm design*. Since these are computer science jargon, one easily falls into the trap of straight-jacketing computational thinking into 'computer science'. At this juncture, we point the reader to Papert's theory of constructionism. Constructionism is an offspring of Piaget's theory of constructivism, which asserts that the learner can improve his or her learning if he or she is fully engaged in "constructing a meaningful product". Because meaning is specific to the domain of knowledge in which it lives, we assert that computational thinking is domain-specific – in our present paper, it is computational thinking in mathematics.

In this paper, computational thinking in mathematics is thus understood as follows.

Decomposition is the process by which the mathematics problem is broken down into smaller sub-problems or sub-tasks. Problem solving heuristics such as simplifying the problem, making suppositions, trying out on smaller cases/numbers can be regarded as specific actions of problem decomposition. Through decomposition, the original problem, which at first seems complicated, now becomes more tractable since each of the smaller sub-problems/tasks can be managed easily.

Pattern recognition involves seeking out common patterns, structures, trends, characteristics

[1] "Math + C" stands for "Mathematics and Computational thinking"

or regularities in the sea of mathematical data. The disciplinarity of mathematics trains the students to be sensitive to the regularity of structures and patterns. Pattern recognition is involved in every mathematical practice: teasing out the patterns from what seems messy data is the very thing a mathematician does. Data, of course, manifest in various forms, such as numbers, vectors, shapes, algebraic expressions, mathematical structures, etc.

Abstraction is the process of generalizing recognized patterns in the form of theorems or formulae. In particular, abstraction takes place when a problem in real world context is reformulated in mathematical terms. The act of formulating an "everyday content" using the language of mathematics is called *mathematizing*.

Algorithm design involves the planning and development of a set of precise and step-by-step instructions for solving the problem. We call such a set of instructions a program which can be executed by a computer (machine or human being) in an insightless manner. In the language of recursion theory, we say that the problem we are solving is said to be effectively calculable or computable.

Having stated what computational thinking in our paper, we must caution the reader to the situation that there is no universally accepted definition for the term computational thinking. This problematic situation arises partly because of the varied interpretations of the term over different domains and disciplines (e.g., science, mathematics, etc.), and of the connections of these disciplines with computer science, as well as those issues centred on the degree of involvement of the computer. A point to note: recently, a team of researchers led by Weintrop has defined computational thinking in terms of a taxonomy of practices focusing on the applications of computational thinking in mathematics and science ([13]). We shall return to this taxonomy later in our description of the lesson design principles in Section 3.

In summary, we highlight the main features of what we believe characterizes Computational Thinking in Mathematics:

- Computational thinking is a paradigm of knowledge that is supported by constructivism, i.e., computational thinking in action results in the creation of a final product – physical or not.
- Computational thinking cannot take place in vacuum, and must be discussed with respect to a specific domain of knowledge. In our present paper, we speak of computational thinking in mathematics.
- Computational thinking in mathematics is visibly manifest in problem decomposition, pattern recognition, abstraction and algorithm design, which results in the creation of mathematical knowledge (e.g., understand mathematical concepts, learning a mathematical procedure), and mathematics problem solving.

3 Designing "Math + C" lessons

3.1 Four design principles

One of the classroom teacher's main concerns is instructional design. This section presents four design principles to guide practitioners who plan lessons for their students to think computationally in mathematics. These design principles hinge on the four key components of compu-

tational thinking: decomposition, pattern recognition, abstraction and algorithm design. We phrase these design principles in the form of questions to which the teacher answers.

Complexity Principle. Does the mathematical concept give rise to sufficiently complex problem? The problem should involve the use of the identified concept, and be complex enough so that decomposition of this main problem into sub-problems is a needful step. If the problem or task is routine or too simple, e.g., there exists a ready-made solution or method, then decomposition is uncalled for.

Data Principle. Can the mathematical concept occur in various forms so that it is possible to collect data for its occurrence? The topic should involves observable and quantifiable data that can be collected, created, analyzed, and shared.

Mathematics Principle. Can the problem associated to the mathematical concept be mathematized? Mathematization is the formulation of the problem using mathematical terms. It turns a problem in real world context in an abstract and precise manner to a mathematics problem. We do not restrict mathematics to mean only numbers, algebra, geometry, and so on. Rather, mathematics can have a more inclusive meaning of encompassing abstract concepts and structures which are definable, representable, and can be reasoned about within some logical framework.

Computability Principle. Does there exists an effectively calculable solution to the mathematized problem? By 'effective calculable' we use it in the sense of Recursion (or Computability) Theory, that is, there exists a a computer program that can calculate a solution to the problem through a finite procedure via a physical agent (e.g., machine, human being).

3.2 Construction of meaningful product

Based on the Papert's maxim that learning can be enhanced when the learner is engaged in constructing meaningful product, each Math + C lesson designed based on the preceding set of principles must engage students in constructing some product. This product (physical or not) should be observable, i.e., there must be visible or testable quantities to allow teacher's assessment of students' understanding of the mathematical concept. Visible products need not be restricted to final products, as they can refer to a variety of entities that are expressed as evidence of computational thinking. Here we rely heavily on Weintrop's taxonomy of practices associated to computational thinking in mathematics classroom.

Data practices. According to [6, p. 27], "all sciences share certain common features at the core of their problem solving and inquiry approaches. Chief among these is the attitude that data and evidence hold a primary position in deciding any issue". Data play an important role in the conduct of mathematical inquiry in that it allows one to seek for patterns and regularities present in the observed data – a crucial first step in making guesses, conjectures and claims. The subprocesses in data practices are listed in Figure 1.

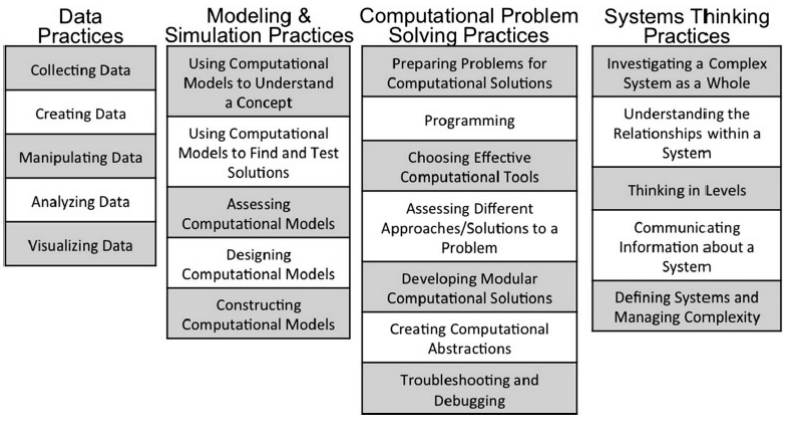

Figure 1: Weintrop's taxonomy of practices related to computational thinking in mathematics and sciences, *Source*: [13, p. 135]

Modelling and simulation practices. Mathematicians frequently manufacture, refine and use mathematical models of phenomena. Models manifest in many forms, e.g., flowcharts, diagrams, equations, computer simulations, as well as physical models ([7]). Computational models have been employed by mathematicians to understand concepts, find and test solutions. Problems that cannot be solved analytically often have effective numerical solutions that are generated by computer simulations, e.g., by Monte Carlo methods. The subprocesses in modelling and simulation practices are listed in Figure 1.

Computational problem solving practices. The power of computers can now be exploited to make quick and tedious calculations, which would have otherwise not been possible. Computational strategies become part of the problem solving toolkit of mathematicians. Research has also flagged up evidence that students who employed computational problem solving approaches, e.g., used programming, developed algorithms, etc., developed deep understandings of mathematical phenomena ([14]). The subprocesses in computational problem solving practices are listed in Figure 1.

Systems thinking practices. Real world problems of the present age are often highly complex that involve a large number of variables, depend on many direct and indirect effects and factors, and comprise several parts. Systems thinking approach, unlike traditional approaches, allow the problem solver to focus on an "inclusive examination of how the system and its constituent parts interact and relate to one another as a whole" ([3]). The subprocesses in systems thinking practices are listed in Figure 1.

3.3 Math + C and cognitive activations

Recall from [10, p. 19] that cognitive activation concerns teaching pupils generic problem solving strategies such as making a summary, asking question to clarify, and making predictions. These skills are essential to their later success in mathematics problem solving. Figure 2 shows

the OECD average for the percentage of students who reported their teachers use cognitive-activation strategies in every lesson or most lesson.

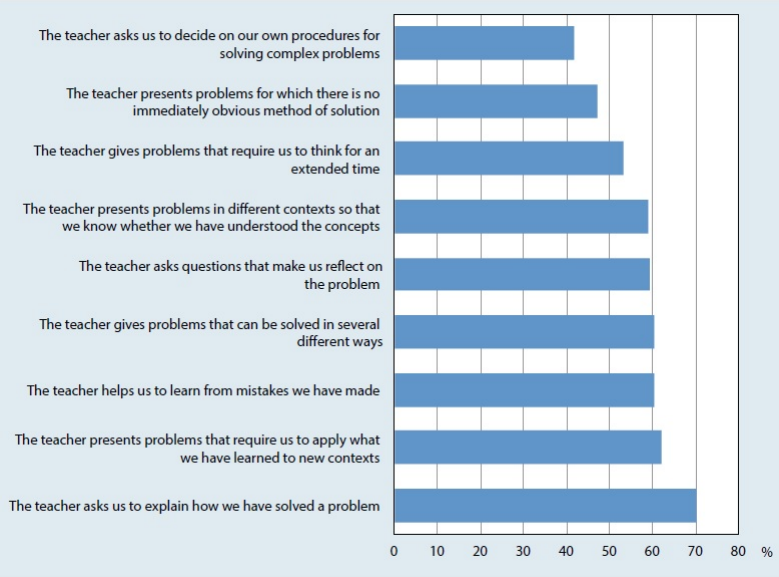

Figure 2: Cognitive-activation instruction

Even if a lesson is very well-designed with every intention for the best learning outcome, it will remain useless until it is being implemented in the classroom. Math + C lessons are crafted by giving due consideration on the complexity of the problem at hand. This would entail that a typical Math + C lesson would engage students over an extended period of time in problem solving. Because the problem selected cannot be solved using ready-made methods or procedures, students can focus on linking the new information (e.g., assumptions and requirements of the problem) to information that they have already learned, and applying their skills to a new context. Making connections between mathematical facts, procedures and ideas will result in enhanced learning and a deeper understanding of the concepts ([4]). The authors hold to the belief that the richness of the Math + C lessons allow teachers ample opportunity to incorporate cognitive-activations which are listed in Figure 2.

4 Sample Math + C lesson

We now employ the design principles described in Section 3, keeping in mind Weintrop's taxonomy of practices for computational thinking in mathematics as well as the use of cognitive-activations in lessons.

Due to page constraints, we show only one sample lesson. For the sample lesson, we shall employ the questions listed in the design principles as a checklist, and highlight the related practices. Suggestions will also be provided for the classroom teacher concerning the choice of cognitive-activations in the sample lesson.

4.1 Students' profile

In Singapore, students aged 16 sit for a national examination based on the Cambridge 'O' level Mathematics syllabus. In schools, computer laboratories are available and Microsoft EXCEL is 'ubiquitous' in desktop computers. To keep the computer-science/technology overheads low, we use *only* EXCEL spreadsheets for simple coding. Just-in-time computer skills will be taught to the students on a need basis; we do not assume any pre-requisite knowledge from Computer Science for this section.

The 'O' level Mathematics syllabus ([8]) includes a subtopic called "primes and prime factorisation" under the strand "Number and Algebra". We shall select this subtopic for our lesson design, and apply the lesson design principles proposed earlier to craft a Math + C lesson.

Students are expected to know the definition of prime numbers, and be able to recognize, by sight, all the prime numbers less than 100. The syllabus suggests explicitly that students should have opportunities to classify whole numbers based on their number of factors, and be able to explain why 0 and 1 are not primes. Students are also expected to obtain highest common factor (HCF) and lowest common multiple (LCM), squares, cubes, square roots and cube roots by prime factorisation. Figure 3 shows a typical assessment item used in schools to this level of students.

Sample question.
Find the smallest whole number k such that $\frac{1274}{k}$ is a perfect square.

Figure 3: Sample question for prime factorisation

The above sample question assesses the student's ability to perform prime factorisation of 1274, i.e., $1274 = 2 \times 7^2 \times 13$. Since a perfect square (greater than 1) has an even multiplicity of prime factors, it follows that the least positive integer value of k is $2 \times 13 = 26$. Because this question is considered a routine task for even a 13-year-old, it would not be seen as an unfamiliar problem to a 16-year-old.

4.2 Main problem

We propose the computational thinking may help forge students' deeper understanding of prime factorisation. The complexity of the above sample question is not sufficiently high for us to design a Math + C lesson that incorporates cognitive-activation.

In order to create a problem for which the student has no ready solution, we consider replacing the dividend 1274 by an expression which is not so familiar and yet not incomprehensible for a 16-year old. We select the sum of consecutive cubes:

$$S_n = 1^3 + 2^3 + 3^3 + \cdots + n^3,$$

which is an algebraic expression in n. In the 'O' level syllabus, students are accustomed to number patterns and sequences, and so in this respect, students of this level understand the meaning of the expression. However, they are not expected to know the closed formula for S_n.

The above sum of consecutive cubes is suitable because the computer (either human being or machine) can be tasked to compute its value for a given value of n. To test the student's understanding of prime as 'irreducibles', it would be appropriate to ask for what value(s) of n is

the above sum divisible by a fixed prime p. Although a complete answer to this question requires the knowledge of the closed formula for the sum, the student can exploit computational thinking to perform computational problem solving. A simple enumeration of n and the evaluation of the corresponding sum $S_n = 1^3 + 2^3 + 3^3 + \cdots + n^3$, followed by a test of divisibility by the fixed prime p would yield the answer by brute force.

We now craft this item formally below.

Problem. (Prime factorisation)
A student is playing with the number pattern generated by summing the first n perfect cubes:

$$S_n = 1^3 + 2^3 + \ldots + n^3.$$

(a) In the midst of her investigations, she wishes to find out the smallest value of n such that $S(n)$ is a multiple of the prime number p, for each of the following cases:
(i) $p = 5$; (ii) $p = 11$; (iii) $p = 13$.

(b) Relying on your preceding experience, can you formulate a simple rule that one can apply to find the smallest value of n for which $S(n)$ is a multiple of a given prime p?

(c) Later the student discovered from a textbook that the general formula for S_n is given by

$$S_n = \frac{1}{4}n^2(n+1)^2.$$

Can you use this new information to explain why your rule always work?

4.3 Applying design principles

The lesson is intended to be pair work, where two students work on the problem with suitable scaffolding provided by the teacher. We assume that the students have the background knowledge of prime numbers and prime factorisation. We now invoke the four design principles in turn to validate the lesson task, and generate the desired lesson plan.

Complexity Principle. The topic of summation and series is not within the scope of the 'O' level syllabus, and students are expected not to have an immediate solution to this problem. This problem involves prime numbers, which is the identified subtopic within the given strand. This problem can be broken down naturally into smaller sub-problems:

(1) Given the value of n, can we calculate the value of S_n?

(2) Can we test whether the given prime p divides the value of S_n?

(3) Can we spot a pattern as one runs 'down' the possible values, by brute force, i.e., $k = 1, 2, \ldots, n$, and test the divisibility of each S_k by p?

Data Principle. The data involved are all positive integers. In this problem, it is crucial to generate the consecutive cubes and sum them up. So, the sum S_n is a derived datum from n. Additionally, the boolean value of the test of divisibility of S_n by p is a crucial derived datum.

The list of consecutive positive integers forms a new data. The required data can be created using spreadsheet tools. The tests of divisibility will yield a list of boolean values which can be analyzed to tease out the salient pattern.

Mathematics Principle. The given problem deals with the number-theoretic properties and not a problem in real world context, and hence the mathematization is straightforward.

Computability Principle. Summation and test of divisibility are computable functions, which can be done by hand or using a computer. Just-in-time skills need to be taught to the students:

(1) Recursive definition in EXCEL can be used to generate the list of consecutive integers.

(2) Functions can be defined by using equation applied to arguments of designated cells.

(3) Test of divisibility can be carried out using a program.

In addition, the part (b) of this problem requires the student to write down a simple "rule of the thumb" to determine the least value of n for a given p so that the divisibility criterion is met. In actuality, the question item demands the student to adopt an algorithmic mind-set by manufacturing a rule or a recipe.

4.4 Meaningful Math + C product

One important feature of a Math + C lesson is the students' product that mechanises the part (a) of the problem. This would allow the students to play with different prime values, and solve the problem computationally. In this case, the teacher displays the finished product – an EXCEL spreadsheet which is constructed for the purpose of determining the least positive value of n for which the given prime divides the value of S_n. The lesson package comprises of three sessions (about 50 minutes each). This would allow an extended time period for the students to explore the problem and produce a replica of the product shown by the teacher at the beginning of the first lesson. Figure 4 shows the desired finished product.

	A	B	C	D	E	F
1	n	S(n)	Test for p	5	11	13
2	1	1		1	1	1
3	2	9		4	9	9
4	3	36		1	3	10
5	4	100		0	1	9
6	5	225		0	5	4
7	6	441		1	1	12
8	7	784		4	3	4
9	8	1296		1	9	9
10	9	2025		0	1	10
11	10	3025		0	0	9
12	11	4356		1	0	1
13	12	6084		4	1	0
14	13	8281		1	9	0

Figure 4: Desired finished product in EXCEL form

The three sessions can be organized by the teacher to guide the students in producing the students' version of the above product. Just-in-time EXCEL coding skills need to be taught to the students in order to achieve each of the smaller goals below. For each of these goals, we indicate clearly the practices under the Wientrop's taxonomy.

- **Data practices (Subprocess: Creating data).** The column `A` contains the consecutive whole numbers, starting from 1, situated at Cell `A2`. The crucial command is given in Cell `A3` by the equation `= A2 + 1`. By "dragging" the content of Cell `A3`, EXCEL automatically creates a recursive call by repeating the pattern of the equation. In other words, the sequence generated in column `A` is the recurrence relation:

$$T_{k+1} = T_k + 1,\ T_1 = 1$$

  and, so $T_k = k$ for all $k = 1, 2, \ldots$.

- **Data practices (Subprocess: Creating data).** The column `B` stores the values of $S(n)$. This achieved through the EXCEL realization of the recurrence relation:

$$S_{k+1} = S_k + {T_{k+1}}^3.$$

  Since $T_{k+1} = k + 1$, it follows that

$$S_{k+1} = S_k + (k+1)^3$$

  so that $S_k = \sum_{r=1}^{k} r^3$, which is the value of $S(k)$. Hence Cell `B2` is defined by the equation `= A2^3`, and Cell `B3` is defined by the equation `= B2 + A3^3`. The column `B` is then produced by recursion.

- **Data practices (Subprocess: Manipulating data), and Computational Problem Solving (Subprocess: Preparing problems for computational solutions).** Each of the columns `D, E, F` stores the remainders when $S(n)$ is divided by the corresponding prime in `D1, E1, F1`. It suffices to see how this can be done for column `D`. In Cell `D2`, the equation `= MOD(B2,$D$1)` is used. When a recursive 'drag' is applied on Cell `D2` to produce more cells in column `D`, the value of `$D$1` is fixed for each subsequent cell.

- **Computational problem solving practices (Subprocess: Programming).** Once the appropriate data have been created as in Figure 4, data analysis is performed on the entries in columns `D, E, F`. The first value of n in column `A` for which the corresponding cell in column `D, E, F` is zero will be the required answer. For example, the least value of n for which $S(n)$ is a multiple of $p = 5$ is $n = 4$ (Cell `D5` contains the first `0` in the column `D`).

- **Data practices (Subprocess: Analyzing data).** The pattern of occurrence of the first `0` in each column is regular. For $p = 5, 11, 13$, the observed least value of n is given by $p - 1$. It is tempting to conjecture that for all prime p, the least positive value of n so that p divides $S(n)$ is $p - 1$. This is where the teacher cautions the students to check their solutions by testing the conjecture for different values of p – smaller values: e.g., $p = 2, 3$, and larger values: e.g., $p = 17, 19$. For this sub-goal, the students are expected to state the complete rule of the thumb as required in (b) of the main problem.

- **Computational problem solving practices (Subprocess: Assessing different solutions to a problem).** Given the closed formula for S_n (in terms of n) as in (c) of the problem, the students are guided to use their understanding of prime factorisation to locate the possible occurrence of p within the expression $\frac{1}{4}n^2(n+1)^2$. The goal of the lesson is for the students to mathematically reason about the regularity in the occurrences of the `0`'s under each column `D, E, F`.

4.5 Checking the cognitive activations

Does the proposed lesson package offer ample opportunity for the teacher to incorporate cognitive activations as an effective teaching method? We validate this lesson using the features highlighted in Figure 2.

- **The teacher asks us to decide on our own procedures for solving the complex problem.** Although the teacher guides the student to achieve the desired EXCEL spreadsheet, the students are free to exercise their own discretion and decide what formulae to use in order to generate the required data.
- **The teacher presents problems for which there is no immediately obvious solution.** The students of this level do not possess immediate solution methods for this problem.
- **The teacher gives problems that require us to think for an extended time.** The given problem is of a sufficient complexity and difficulty that requires the students to work out the solution over three separate sessions, totalling 150 minutes.
- **The teacher asks questions that make us reflect on the problem.** This problem consists of many small sub-problems, and the solutions to these sub-problems require mathematical reasoning. The teacher can take the opportunity to ask questions of a reflective nature, leading students to deeper thinking.
- **The teacher gives problems that can be solve in several different ways.** This problem can be solved in two ways. The first one relies on computational means, and the second by analytical means. Both these methods require mathematical reasoning, e.g., deciding which calculations must be made, how programs should be written to solve the problem, and how prime numbers play a role in explaining what they observe during their computational experiences.
- **The teacher helps us to learn from the mistakes we make.** The teacher plays the role of a facilitator as well as the knowledgeable one. Mistakes in programming are inevitable, and thus students would have ample opportunity to learn from the mistakes they make in their programming experience; not to forget: trouble-shooting and debugging is one of the computational problem solving processes the computationally thinking students must go through.
- **The teacher presents problems that require us to apply what we have learned in new contexts.** Prime numbers and prime factorisation form part of the pre-requisite knowledge to be applied to solve this problem. This problem is unfamiliar to the students and hence presents a new context for them, i.e., concerning the sum of consecutive cubes.

- **The teacher asks us to explain how we have solved the problem.** During each lesson, the teacher consistently visits each pair, and asks students questions, inviting them to explain how they arrived at the program and the solution. In the last session, each pair will be asked to present their finished product, and to demonstrate the versatility of their program to tackle new problems.

5 How do we move on from here?

PISA documents have given us a lot of insights and recommendations – in this case, they call to our attention the use of computational thinking in teaching and learning mathematics, as well as the use of cognitive-activations in effective teaching of mathematics. Herein, we propose four design principles to guide the classroom teacher in crafting mathematics lessons that crucially target mathematics learning via computational thinking. We have been intentional in the incorporation of cognitive-activations in our Math + C lessons.

So far, what we have taken are first steps in bridging the theory-practice gap of realizing computational thinking in the 21st century mathematics classroom. Where can we go from here? Definitely in many different ways!

Firstly, we must put our design principles to the test by inviting mathematics teachers to implement them in authentic classroom situations. In our on-going OER project, the project team has already shared with them the four design principles, and the teacher has also started designing their own Math + C lessons. The implementation of these lessons will happen in the near future, which we shall report in the next ATCM.

Secondly, we must validate whether students actually have deeper learning experience in these Math + C lessons. Again, for this to happen, we have to make field observations for those Math + C lessons to be actually implemented. We plan to interview both the students and the teachers of the Math + C lessons to find out how they feel about teaching and learning mathematics using computational thinking.

Thirdly, we must figure out how to assess the students' learning and, hence how to measure the effectiveness of this teaching innovation that exploits computational thinking. This requires us to draft a set of assessment rubrics for crediting students' effort in applying computational thinking in their Math + C lessons. At the moment, the project team is studying the APOS theory (Action-Process-Object-Schema) pioneered by Dubinsky and his collaborators ([5, 1, 2]), and looking into the possibility of applying this framework as a basis of observing the hierarchies of cognition associated to computational thinking in mathematics.

Fourthly, we see that the teacher plays a crucial role in the Math + C lessons. It cannot be assumed that the teacher is equipped with necessary pedagogical content knowledge or the mathematical content knowledge (especially, the computational thinking aspect of it). Further support must be given by the project team in building the teaching competencies of the teachers who wish to implement Math + C lessons.

References

[1] Arnon, I., Cottrill, J., Dubinsky, E., Oktaç, A., Roa Fuentes, S., Trigueros, M., and Weller, K. (2014). *APOS Theory*. Availabe at `https://doi.org/10.1007/978-1-4614-7966-6`.

[2] Asiala, M., Brown, A., DeVries, D. J., Dubinsky, E., and Matthews, D. (1997). *A Framework for Research and Curriculum Development in Undergraduate Mathematics Education* (Vol. 40).

[3] Assaraf, OB-Z., and Orion, N. (2005). Development of system thinking skills in the context of earth system education. *J. Res. Sci. Teach.*, **42**(5), pp. 518 – 560.

[4] Burge, B., Lenkeit, J., and Sizmur, J. (2015). PISA in practice – Cognitive activation in maths: How to use it in the classroom, National Foundation for Educational Research in England and Wales (NFER), Slough.

[5] Dubinsky, E., and McDonald, M. A. (2001). APOS: A constructivist theory of learning in undergraduate mathematics education research. *The Teaching and Leaning of Mathematics at University Level*, 275–282.

[6] Duschl., R. A. , Schweingruber, H. A., Shouse, A. W. (2007). *Taking science to school: learning and teaching science in grades K-8*. National Academies Press, Washington, DC.

[7] Harrison, A. G., Treagust, D. F. (2000). A typology of school science models. *Int. J. Sci. Educ.*, **22**(9), pp. 1011 – 1026.

[8] Ministry of Education. (2012). Mathematics Syllabus: Secondary One to Four – Express Course, Normal Academic Course. Singapore.

[9] OECD. Ten Questions for Mathematics Teachers ... and how PISA can help answer them. PISA, OECD Publishing, Paris.

[10] OECD. PISA 2021 Mathematics Framework (Draft). November 2018.

[11] Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*, Basic Books, Inc. New York.

[12] Papert, S. (1996). An Exploration in the Space of Mathematics Educations, *International Journal of Computers for Mathematical Learning*, **1**(1), 95 – 123.

[13] Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., and Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Sciences Classrooms. *Journal of Science Education and Technology*, **25**(1), 127–147.

[14] Wilensky, U. (1995) Paradox, programming, and learning probability: a case study in a connected mathematics framework. *J. Math. Behav.*, **14**(2), pp. 253 – 280.

[15] Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, **49**(3), 35–35.

[16] Wing, J. (2011). Computational Thinking – What and Why?, *The Magazine of Carnegie Mellon Universitys School of Computer Science*, March 2011. The LINK, Research Notebook. `https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why`.