PROPOSAL

Computational Thinking: A New Curriculum and Qualification to Support Post-16 Technical Education

Conrad Wolfram, Alec Titterton, Mary Curnock Cook

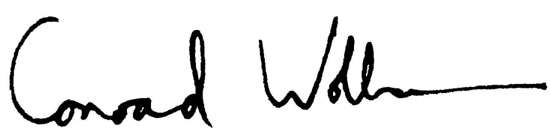
September 2019

Foreword by Conrad Wolfram

Never before has new technology threatened to take over and extend such quintessentially human capabilities as "knowledge" and "intelligence". Yet the AI age we're entering promises to do just that—posing urgent questions for education, its outcomes and content. What are today's survival skills? What are today's top value-adds? What subject matter will lead to success: best combining human intellect with AI, rather than losing at a head-to-head competition with what computers do best?

Urgently answering these questions, in practice on the ground, will be key to the UK's future prosperity. With good computing resources, impetus in data science and AI, and a societal ability to think outside the box, the UK is in a prime position to be a leader on the starting grid of AI-age education. But it cannot realise this potential without the right core subject, pragmatically introduced to fit with today's already pressured educational system—with support by key parties.

The prize of incisive intervention is a real opportunity for the UK to leapfrog other countries, much as it did with universal education in the nineteenth century.



Executive Summary

This paper proposes a new curriculum and qualification suite which uses real-life problems to engage students with a range of technical, mathematical, data, modelling, statistical and thinking skills to support life and work in the 21st century technology age. Termed 'Computational Thinking', this new multi-disciplinary domain will support the Government's Industrial Strategy as well as providing the underpinning skills to support technical education and STEM subjects. By using a step-graded qualification system (like music exams), Computational Thinking can be taught *ab initio* or at higher levels depending on prior attainment. Colleges and universities would run a series of different-level classes simultaneously so that students of differing capability could progress through the grades, regardless of age or stage of learning.

As well as developing critical skills for the 21st century, Computational Thinking will address significant numeracy and mathematics deficits whilst also supporting successful further learning across the full range of STEM and non-STEM disciplines.

Sir Adrian Smith's Review of Post-16 Mathematics[1] focussed repeatedly on the need for mathematical skills to embrace more fully statistics, data analytics, modelling and quantitative skills. This emphasis was largely absent from his earlier 'Making Mathematics Count' report in 2004, underlining how recently the requirements of mathematics have changed. Adrian Smith summed up the problem thus:

> *"England remains unusual among advanced countries in that the study of mathematics is not universal for all students beyond age 16. Almost three quarters of students with an A*-C in GCSE mathematics at age 16 choose not to study mathematics beyond this level. England was the only country in a 2013 sample of developed economies where young adults performed no better than older adults in numeracy proficiency.*
>
> *With the exception of mathematics degrees, more than 40 per cent of English 19 year olds studying STEM subjects in UK universities do not have a mathematics qualification beyond GCSE. This increases to over 80 per cent for students on non-STEM degree courses, many of which have a significant quantitative element. A lack of confidence and anxiety about mathematics/statistics are problems for many university students; and many have done little or no mathematics pre-university for at least two years."*

---

[1] https://www.gov.uk/government/publications/smith-review-of-post-16-maths-report-and-government-response

Furthermore, the UK lags behind international competitors:

Proportion of students in post-16 (or 'upper secondary') education or training studying any mathematics

All (95-100 per cent)	Czech Republic, Estonia, Finland, Japan, Korea, Russia, Sweden, Taiwan
Most (81-94 per cent)	Canada (BC), France, Germany, Hungary, Ireland, USA (Mass.)
Many (51-80 per cent)	Australia (NSW), Netherlands, New Zealand, Singapore
Some (21-50 per cent)	Hong Kong, Scotland, Spain
Few (6-20 per cent)	England, Wales, Northern Ireland

Table 1. Proportion of students in post-16 (or 'upper secondary') education or training studying any mathematics, based on the 2010 Nuffield Foundation report.[67]

The proposed Computational Thinking curriculum specifically meets the first 'Grand Challenge' of the Industrial Strategy:

> *AI and Data Economy—We will put the UK at the forefront of the artificial intelligence and data revolution*

And the key Policy Aim to:

> *Establish a technical education system that rivals the best in the world to stand alongside our world-class higher education system*

By taking the focus off mathematics as a subject and emphasising instead (and amongst other things) the mathematical tools and skills needed to solve a range of problems, a Computational Thinking curriculum could make meaningful inroads to the UK's persistent and intractable deficit in mathematical and quantitative skills. Basing the curriculum around problems that students want to solve gives meaning and purpose to learning relevant mathematical tools and could help prepare new generations with skills that are in high demand for both technical and non-technical occupations and further learning. The importance of Computational Thinking was referenced in the US as early as 2005 by the PITAC (President's Information Technology Advisory Committee) which advised that

> *'…computational science is one of the most important technical fields of the 21st century because it is essential to advances throughout society'*[2]

Initially introducing this new qualification system post-16 in colleges and post-18 in universities, Computational Thinking can be tested and refined prior to considering its introduction at earlier stages of education. This approach would also remove the need to disrupt the already-crowded primary and secondary curriculum or introduce more change to an already reform-fatigued system. It could, however, provide a welcome alternative to resitting for those who have failed to achieve a Grade 4 in GCSE mathematics.

Following successful pilots, the Computational Thinking approach could be implemented as a requirement attached to the drawdown of public funding for specified programmes of study (including apprenticeships) at Levels 4, 5, 6 and higher.

[2] http://vis.cs.brown.edu/docs/pdf/Pitac-2005-CSE.pdf

What Is Computational Thinking (CT)?

Computational Thinking is not a 'subject'. Rather it is an inter-disciplinary domain, strongly underpinned by mathematics and highly aligned with, and derived from, problem solving in a technology and data-rich environment. Computational Thinking will provide a broad underpinning for all further study including Higher Technical Qualifications and STEM degrees. It will specifically support the Government's ambitions for leadership in the AI and data economy, as well as providing the problem-solving, quantitative and data skills which are needed across the creative arts, humanities and social sciences, and in most modern workplaces.

As Alan Bundy from the School of Informatics at the University of Edinburgh noted in 2007:

> *'…the computational thinking revolution [] is changing the way we think. Computational concepts provide a new language for describing hypotheses and theories. Computers provide an extension to our cognitive faculties. If you want to understand the 21st Century then you must first understand computation.'*[3]

The CT concept was primarily designed as a new approach to mathematics education, to modernise and contextualise mathematics in a way that developed the required skills for 21st century life and work as well as engaging students better in a subject traditionally rejected by a large proportion of the population. The following table shows a variety of mathematical concepts more generally in demand today, none of which are covered in any depth in the current secondary curriculum and not in the context of real-world problems.

New Mathematics or Maths-Related Topics	
• Machine learning (including ethics)	• Encryption and privacy
• Hypothesis creation, testing and interpretation	• Fitting models to data
• Model creation and validation techniques	• Optimisation of parameters
• Pattern matching / image recognition	• Data visualisation
• Graphs and networks	• Digital representation of 3D objects
• Significance and risk	• Resource scheduling
• Monte Carlo simulation	• Causation vs correlation

[3] https://core.ac.uk/download/pdf/28961399.pdf

The examples below show how CT differs from the traditional mathematics curriculum:

Example Traditional Mathematics Topics

- Invert a matrix
- Use the chain rule
- Complete the square
- Calculate angles in a triangle
- Solve simultaneous equations
- Simplify a surd or recurring decimal

Example Computational Thinking Problems

- Who gets picked?
- Can I spot a cheat?
- Should I insure my laptop?
- Where's the missing plane?
- How many words do I know?
- How do I train my computer?
- How do I design controls for my game?
- How can I create natural looking shapes?

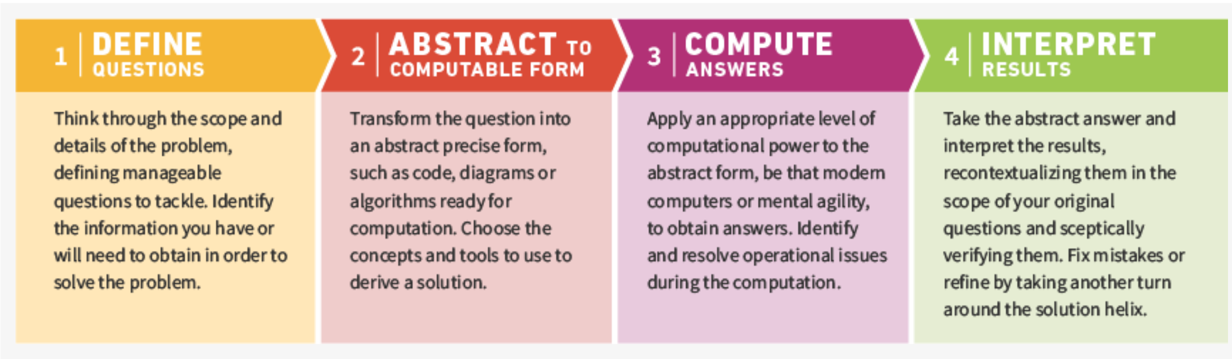
However, computational thinking is equally applicable to a very wide range of subjects. Because it is a way of thinking, whether in design (How can I design a streamlined cycle helmet?) or history (What was the key message each President's inaugural address delivered?) or music (How did Bach's use of motifs change over his career?), all subjects will benefit from a computational thinking approach. **Annex A** provides a wide range of example topics and subject relationships.

The range of skills utilised in the CT curriculum is extensive, for example:

Abstract a diagram	Experiment	Quiz
Assess validity	Find the information	Role-play
Brainstorm	Find the mistake	Run a simulation
Compute answer	Guided discussion	Sorting/Classifying/Ordering
Compose/collect raw data	Interpret a chart	Summary/Reinforce learning
Comprehension	Manipulate to discover	Synthesise code
Cross-examine	Modify code	Video essay
Data structuring	Obtain pre-existing data	Visualise data
Distill opinions	Pose a question	Watch video
Essay or report	Presentation	Write instructions
Estimate a value	Primer	

Computational thinking requires a rigorous and repeatable four-step problem-solving process to be applied to ideas, challenges and opportunities.

You start by **defining the question** that you really want to address—a step shared with most definitions of ‘critical thinking’.



But computational thinking follows this with a crucial transitional step 2 in which you take these questions and **abstract** them into a computational form—be that code, diagrams or algorithms. This has several purposes. It means that hundreds of years’ worth of concepts and tools can be brought to bear on the question (usually by computer), because you've turned the question into a form ready for this high-fidelity machinery to do its work. Another purpose of step 2 is to force a more precise definition of the question. In many cases this abstraction step is the most demanding of conceptual understanding, creativity, experience and insight.

After abstraction comes the **computation** itself—step 3—where the question is transformed into an abstract answer—usually by a computer.

In step 4 we take this abstract answer and **interpret the results**, re-contextualising them in the scope of our original questions and sceptically verifying them.

The process rarely stops at that point because it can be applied over and over again, with output informing the next input until you deem the answers sufficiently good. This might take just a minute for a simple estimation or a whole lifetime for a scientific discovery. It is helpful to represent this iteration as ascending a helix made up of a roadway of the four steps, repeating in sequence until you can declare success. Whilst emphasising the process end of computational thinking, its power of application comes from the very *human* qualities of creativity and conceptual understanding. The magic is in optimising how process, computer and human can be put together to solve increasingly tough problems.

Before modern computers, step 3—computation—was very expensive because it had to be done by hand. Therefore, in real life, people would try very hard to minimise the amount of computation at the expense of much more upfront deliberation in steps 1 (defining the question) and 2 (abstracting). It was a very deliberate process. Now, more often than not, you might have a much more scientific or experimental approach with a looser initial question for step 1 (like 'Can I find something interesting in this data?'), an abstraction in step 2 to a multiplicity of computations (like 'Let me try plotting correlation of all the pairs of data.') because computation in step 3 is so cheap and effective you can try it repeatedly and not worry if there's wastage at that step. Modern technology has dramatically shifted the effective process because you don't get stuck at step 3.

Computational thinking will require knowledge of the possible concepts and tools that can be used in the computation step (examples also in **Annex A**), and experience of using code to apply many of these will be integral to the learning of problem solving in this manner.

A Graded Curriculum and Assessment

By organising Computational Thinking into eight Grades (in a similar fashion to music exams), all providers can deliver the curriculum without any specific requirement for prior attainment. Students would start at the appropriate grade for their ability, assessed by an initial diagnostic test.

Broadly the Grades might be pegged to the Ofqual Regulated Qualifications Framework as follows:

Grade 1	(Ab initio) Level 2/sub-level 2
Grade 2	Level 2
Grade 3	Level 2
Grade 4	Level 3
Grade 5	Level 3
Grade 6	Level 4
Grade 7	Level 5
Grade 8	Level 6

Assessment needs to utilise more open-ended questions that target the new outcomes of the computational thinking curriculum and assume a computer is available as a default.

Examination remains a key instrument in summative assessment, and this would be computer based—carried out in the same environment as the learning, with access to the same tools. A range of assessment approaches could be used, such as:

1. *Examination*
2. *Controlled project work (extended hours under restricted conditions)*
3. *Portfolio creation (evidence gathered throughout the duration of a course)*
4. *Peer-reviewed and peer-reviewing activities*
5. *Presentation (multimedia prepared under unrestricted conditions)*
6. *Interview (oral questioning)*

The awarding body for the graded assessments will need to be ascertained.

Teaching and Learning Support

Whilst specific training, curriculum materials and pedagogical support would be needed, it is likely that the skills exist in universities and Further Education Colleges to teach Computational Thinking at all eight grades. Further development would be needed in the following areas:

- *A diagnostic test or assessment to determine the appropriate grade for students to start*
- *Curriculum and assessment support materials to support all eight grades*
- *Online tools and materials for both teachers and students*
- *Creating a mutual support and professional development network of CT teachers*
- *Provider accreditation*

Implementation

Implementation of this proposal would have four distinct phases:

Phase 1	Refine and develop the curriculum and assessment across eight grades; secure policy support for proposal from government and key stakeholders.
Phase 2	[Concurrent/overlapping with Phase 1] Secure funding for pilot and pilot evaluation.
Phase 3	Pilot over three years in at least three universities and three colleges; evaluate; refine curriculum and assessment.
Phase 4	Secure policy support for implementation and roll out to post-16 students in colleges and for undergraduates in higher education—potentially as a mandatory, credit-bearing programme.

Annex A: Example Module titles, showing subject links, and the concepts and tools used

EXAMPLE MODULE TITLES AND BRIEF DESCRIPTION OF THE PURPOSE	SUBJECT LINKS							CONCEPTS USED Areas of mathematics covered by the problem	TOOLS USED Mathematical tools to solve the problem. (Full list here)
	Physics	Chemistry	Biology	Business	Engineering	Technology	Humanities		
Do I know what I don't know? Understanding how making assumptions and clearly stating them is a necessary part of any model.	✓	✓	✓	✓	✓	✓	✓	Probabilistic model, Assumptions, Data, Model, Simulation, Statistics	Histogram, Distributions, Mean, Median, PDF, CDF
What are the economics of behaviour? Analysing effective strategies for marketing, and population modelling.				✓	✓	✓	✓	Data analysis, Modelling, Profiling, Probability distribution, Expectation	Mean, Median, Distributions, Extrapolation, Fitting
How do computers detect and correct errors? Using real verification methods and understanding their limitations.						✓		Hamming codes, Verification, Error-correcting codes, Moving averages, Fourier smoothing	Check sum
Can you find the best deal? Modelling a varied set of financial plans over time and forecasting the most effective.		✓	✓	✓	✓	✓	✓	Optimization, Modelling, Multivariate problems, Extrapolation, Confidence	Region, Plot, Maximize, Minimize, Nearest, Furthest
Can I crack your password? Comparing brute force techniques to more intelligent methods.			✓	✓	✓	✓		Exponential, Modelling, Behavioural analysis, Combinations	Function, Loop, Count
Cause or just correlation*?* Realising that a correlation can be random, causation is not implied by correlation.	✓	✓	✓	✓	✓	✓	✓	Multivariate data, Correlation, Dependence, Independence, Causation, Fitting	Correlation coefficient, Plot
What resolution do you need? Is the retinal screen really needed? How many pixels do you need at home/cinema/phone?	✓		✓		✓	✓		Trigonometry, Density, Similarity, Image processing	Trigonometrical functions, Graphics
How do populations vary over time? Using models of predator-prey relationships to gauge the impact of contributory factors.		✓	✓	✓	✓	✓	✓	Modelling, Feedback loops, Control systems, Differential equations, Cellular automata	Differential solve
Where should I build the distribution centre? Analysing networks for efficient layouts for distribution of services or communications.				✓	✓	✓	✓	Graph theory, Networks, Optimization, Modelling	Graph, Find shortest path, Find maximum flow, Closeness, Centrality
When will the next peak happen? Analysing trends from data, fitting a model to the data.	✓	✓	✓	✓	✓	✓	✓	Data analysis, Rates of change, Fitting a function to data, Minima, Maxima, Summation	Derivative, Fit, Plot, Integrate, Solve

EXAMPLE MODULE TITLES AND BRIEF DESCRIPTION OF THE PURPOSE	**SUBJECT LINKS**							**CONCEPTS USED** Areas of mathematics covered by the problem	**TOOLS USED** Mathematical tools to solve the problem. (Full list here)
	Physics	Chemistry	Biology	Business	Engineering	Technology	Humanities		
How many words do I know? Extrapolating from a sample to estimate a population parameter.			✓	✓			✓	Data analysis, Sampling, Statistics, Parameter estimation, Variation, Distributions, Confidence, Error, Sampling bias, Probabilistic model	Mean, Median, Quantile, Histogram, Min, Max
Can my eyes be fooled? Using geometry to reproduce 3D effects from 2D screens.	✓				✓	✓		Trigonometry, Dimensions, Perspective, Projection, Transformation, Cartesian coordinates	Graphics, ArcSin, ArcTan, ArcCos, Line, Infinite Line, Polygon, Polyhedra, Image perspective transformation,
How do you map the world? Producing 2D images of 3D surfaces and vice versa.	✓				✓	✓	✓	Projections—linear, Projections—functional, Dimensions, Transformation, Distortion, Great Circles, Nets, Polar coordinates, Limits	Geographics, Polygon, Image transformation, Projection
How big could the biggest specimen be? Using a model distribution to estimate the maximum value.		✓	✓	✓			✓	Probability distributions, Distribution fitting, Sampling bias	Distribution fit, Random choices, Mean, Median
Can I spot a cheat? Testing patterns against known distributions to determine confidence in randomness.		✓	✓	✓	✓	✓	✓	Hypothesis testing, Confidence interval, Significance, Probability distribution, Expectation	Tally, Count, Tuple, Riffle, Quantile, Histogram, Hypothesis tests

About the Authors

Conrad Wolfram, physicist, mathematician and technologist, is strategic director and European cofounder/CEO of Wolfram Research—the 'computation company' behind Mathematica, the Wolfram Language and Wolfram|Alpha (which powers knowledge answers for Apple's Siri). Over the last 30 years he has been a key part of the technology transformation that has brought maths, computation and data science to the forefront of today's world and moved us towards the fourth industrial revolution and AI. This has given him a unique insight into the broadening chasm between school maths and real-world maths, placing him in a pivotal position to fix it—founding computerbasedmath.org in 2010 and computationalthinking.org in 2017. Conrad holds degrees in Natural Sciences and Maths from the University of Cambridge.

Alec Titterton is the content development manager for computationalthinking.org, responsible for taking the vision for a problem-solving, computer-based curriculum and turning it into ready-to-use classroom resources and activities. Alec was previously the national coordinator for Mathematics and Computing specialist schools, building on the experience gained from teaching in secondary schools in the UK for 16 years. Alec holds a degree in Electronic and Computer Engineering from Birmingham and a PGCE from Cambridge University.

Mary Curnock Cook is a seasoned educationalist with expertise in curriculum, qualifications and progression pathways. Mary was CEO of UCAS from 2010-2017. She now holds a variety of NED positions including Chair of the Governing Body of the Dyson Institute, and the Open University Council. She is also a NED for the Student Loans Company and new higher education startup, the London Interdisciplinary School. Mary is a Sloan Fellow with an MSc from London Business School.

Thanks also to Timo Hannay, founder of SchoolDash, and Philipp Legner of mathigon.org for their invaluable input.