PRE-PRINT: Noss, R., Hoyles, C., Saunders, P., Clark-Wilson, A., Benton, L., & Kalas, I. (Accepted). Constructionism can work: the story of ScratchMaths In N. Holbert, M. Berland, & Y. Kafai (Eds.), *Constructionism in Context*. Cambridge, Massachusetts: MIT Press.

Constructionism can work: the story of ScratchMaths

Richard Noss, r.noss@ucl.ac.uk; Celia Hoyles c.hoyles@ucl.ac.uk; Piers Saunders, piers.saunders@ucl.ac.uk; Alison Clark-Wilson, a.clark-wilson@ucl.ac.uk; Laura Benton, l.benton@ucl.ac.uk; and Ivan Kalas, kalas@fmph.uniba.sk
University College London, UK

Abstract

In this chapter, we describe the ScratchMaths (SM) project, which designed and implemented a longitudinal two-year intervention across English schools to promote students' (aged 9-11 years) computational thinking in alignment with mathematical thinking and reasoning , through carefully designed and sequenced classroom activities involving programming in Scratch. We describe SM's research design and pedagogic approach. We report the positive impact of SM on student computational thinking(CT), as measured by a CT test at the end of the first year of the intervention, and that this was particularly evident among educationally disadvantaged students. We note that there was no evidence of any interaction between the impact of SM on CT test scores and gender, so girls and boys both appeared to engage in a with SM to a similar extent. We report that there was no impact of SM on mathematics attainment as measured by the national mathematics test results at the end of the second year of the intervention. We identify the notion of fidelity of implementation and how it appeared to have been influenced by high stakes testing in mathematics as a possible explanatory factor. Finally, we pose some research challenges for the future.

1. Introduction

Seymour Papert proposed that a powerful way for students to build knowledge structures in their minds is to build with external representations, to construct physical or virtual entities that can be reflected on, edited and shared (Papert, 1980). This was the heart of the case for programming embedded in a constructionist environment where students can not only construct and explore powerful ideas guided by feedback but also in so doing retain some ownership of the construction process (Noss & Hoyles, 2017).

Research undertaken during the 1980s and 90s explored the potential beneficial impact and the challenges of learning to program and noted the need to master the programming syntax as well as the semantics of the code, (see Lewis, 2010). In this respect, blocks-based programming languages, such as Scratch, with visual cues including colour, shape and constrained nesting to indicate usage, flow and scope do seem to render some complex concepts more accessible (Resnick, 2012).

In England, a statutory National curriculum for computing was introduced in 2013 with the intention that: "... pupils are taught the principles of information and computation, how digital systems work, and how to put this knowledge to use through programming…" (Department for Education, 2013). The case for computer science in schools is made in Peyton Jones's polemical piece "Code to Joy" (Peyton Jones, 2015), who welcomed the computing curriculum but at the same time, called for digital technology to support teaching and learning in *every* subject. It is only in this way, he argues, that the national computing initiative could be transformational. The *ScratchMaths* (SM) project, through the mobilisation of computational thinking for mathematics learning, represents one example of how this might be achieved by building on prior research into the impact of programming on students' mathematical thinking (Hoyles & Noss, 1992).

In most countries in the developed world, schools have relatively easy access to hardware and software. However, a comprehensive set of materials designed to support a computing curriculum that 'connects' with students and addresses the core concepts of computing is necessary to support implementation (at least in its early stages), along with documentation of the teacher's role. Specification of how 'computing across the curriculum' might be realised is critical but largely under-researched. Moreover, curriculum specification is only the first step towards exploiting computing across the curriculum. What can fluency in programming bring to learning? What might teacherd do to make this happen in their classrooms? In this chapter, we set out to address these questionsin the context of SM. .

2. Aims and Structure of the Research

SM aimed to develop the CT and mathematical knowledge of students aged 9-11 years through programming. Through a process of design research, SM developed a two-year curriculum for this age group (Years 5 and 6 in England), which was aligned to the national computing and mathematics primary curricula, and required approximately 40 hours of teaching time. The SM curriculum promoted the teaching of carefully selected core ideas of computer programming alongside specific fundamental mathematical concepts. SM devised materials for teachers as well as students and for professional development to be delivered face-to-face over 2 days per year. All the activities and approaches were iteratively designed and trialled in four 'design schools'.

The SM content was divided into six modules, three modules per year. In the first year for 9-10 year-old students, computational concepts were foregrounded with mathematical ideas more implicit in modules titled *Tiling Patterns*, *Beetle Geometry* and *Collaborating Sprites*. In the second year, the same students, now 10-11 years old, were introduced to mathematical concepts and mathematical reasoning explicitly through a programming approach along with a set of new computational concepts[1] in modules titled *Building with Numbers*, *Exploring Mathematical Relationships*, and *Coordinates and Geometry*.

Given the challenge of implementing a brand new curriculum, the SM teachers were provided with detailed guidance for navigation through the materials, which were themselves carefully structured and progressive[2]. However, the SM team recognised the tension arising from their quest to provide comprehensive support and the need for teacher appropriation and autonomy, whereby teachers had space to customise the materials to suit their own goals and their students' needs. This is often referred to as the tension, or gap, between *fidelity*[3] and *adaptation*. At the very least, the challenge is to reduce this gap and crucially, to avoid "lethal mutations" (Brown & Campione, 1996), where the aims of the intervention are lost in its implementation.

The SM approach was first to specify each activity in terms of both the computing curriculum *and* the mathematics curriculum. For example, Fig 1 below shows the overview of Module 1.

Figure1:Module1Overview

Module	Investigation	Computing concepts (+ Scratch terms)	Mathematics concepts
Module 1: Tiling Patterns 	*1.1 Moving, Turning and Stamping*	Sprite and its attributes Command, command with input (stamp, move, turn) Program, sequence of commands	• (Y2) Patterns • (Y2) Rotation • (Y3) Angles • (Y4) Coordinates • (Y4) Symmetry • (Y4) Multiplication • (Y5) Translation • (Y5) Transformation • (Y5) Sequences • (Y5) Positive and negative numbers
	1.2 Repeating and Alternating Patterns	Control structures, repetition (repeat) Designing, building and debugging programs (costume)	
	1.3 Circular Rose Patterns	Algorithms Logical reasoning	
	1.4 Defining your own Pattern Blocks	Defining new commands (make a new block)	

Second, the SM team made explicit within the materials, and in the professional development, the pedagogical framework through which the SM curriculum was designed to be operationalised in the classroom, the '*5Es framework*' (Benton, Hoyles, Kalas, & Noss, 2017). This provided guidance for teachers as to how they might support their students to appropriate the core ideas of computational thinking, to reason on the basis of their programs and later, to express key mathematical ideas in Scratch[4].

The five (unordered) constructs underpinning the framework are:

Explore: Students should have opportunities to explore different ways of engaging with and developing computational and mathematical concepts and be encouraged to take control of their own learning as they express these concepts in their programs.

Explain: Students should have opportunities to explain their own ideas, articulate their own learning and the reasoning behind choices of approach, as well as answer and discuss reflective questions from the teacher and peers. Students should be encouraged to use the programming language as a 'tool to think with' and support their explanations.

Envisage: Students should predict outcomes of their own and others' programs *prior* to testing out on the computer.

Exchange: Students should have opportunities to share and build on others' ideas and be encouraged to justify their own solutions and understand or debug others' perspectives.

bridgE: Students should be supported to make links between the Scratch environment and the mathematics domain through explicit re-contextualization and reconstruction.

3. Evaluation of the SM project

In the first phase of SM, the team engaged in design research with four schools where learning goals, curriculum materials and pedagogic strategies were refined in the process of trialling. At the same time, SM was subject to an independent evaluation, conducted by a team of researchers from another university (Boylan, Demack, Wolstenholme, Reidy, & Reaney-Wood, 2018), who adopted a randomised control trial methodology (RCT) involving 6,2325 students in 110 schools. 2986 student scores in 55 treatment schools and 3,246 student scores in 55 control schools were compared on a specially designed test of CT administered after one year of the project (students age 9-10 years), and on the national tests in mathematics (Key Stage 2 Standard Assessment Tests) after two years at the age of 11 years. The schools were matched at the unit of the school according to two standard measures: socio-economic status (using a proxy measure of eligibility for free schools meals, FSM) and prior attainment as measured by the national standardised mathematics assessment at age 8 years.

Professional development was undertaken in 7 ‘hubs’ across England and led by the SM team with support from local coordinators. Inferences were made about SM implementation from survey data. All schools participating in the trial were asked to complete online surveys at the end of the first year following the teaching of the 9-10 year olds (S1) and again after the second year following the teaching of the same students now 10-11 years old (S2). 38 schools responded to S1, 31 responded to S2 and 28 schools responded to both.

These survey results, alongside data triangulation with follow-up communications with schools and selected school visits, were used by the SM team to classify the schools according to their fidelity (Dane & Schneider, 1998), that is, how far the innovation was implemented according to its aims and objectives. The team developed five criteria to be used as proxy measures of fidelity: engagement in professional development (PD), technology access, curriculum coverage, time and curriculum progression.

3.1 Fidelity of implementation in the first year and outcomes in computational thinking

The fidelity measures were applied to each school in the trial and the results for the first year were as follows.

Professional development: 54 out of our 55 schools attended some SM PD involving a total of 105 teachers. 48 schools were judged as high fidelity on this measure (87% of the sample) 6 medium and one low.

Technology access: Only one school was unable to provide a pupil-to-computer ratio of at least 2:1, which accords with OECD data on the high levels of computer access in UK schools (OECD, 2015).

Coverage: We interpreted fidelity of coverage in Y5 as shown in Table 1 based on S1 survey data, indicating that 26 schools implemented SM with high fidelity and 9 with medium fidelity together representing 97% of the sample.

Table 1: Curriculum coverage in Y5 as reported in school survey with 36 out of 55 respondents.

Fidelity: coverage	**Y5 (n=36)**
High: covered all 3 modules	26
Medium: covered 2 modules	9
Low: remaining schools	1

Curriculum time: We interpreted fidelity of teaching time in Y5 as shown in Table 2, suggesting 19 schools were high fidelity and 16 medium fidelity, together representing 97% of the sample.

Table 2: Curriculum time spent in Y5 as reported in school survey with 36 out of 55 respondents.

Fidelity: curriculum time	**Y5 (n=36)**

High: 20 hours or more	19
Medium: 12 to < 20 hours	16
Low: remaining schools	1

Curriculum progression: For the 36 schools who responded, 35 reported that they followed the progression as set out in the materials.

These data suggest that in the first year of implementation schools were highly engaged with SM and the implementation, at least in terms of these measures, was aligned with the intentions of SM.

The results of the quantitative analysis of the scores on the CT test by the independent evaluator revealed that after one year of the ScratchMaths intervention there was a statistically significant and positive impact on computational thinking for Y5 students aged 9-10 years with an effect size of +0.15 standard deviations and an estimated statistical power of 60%. In addition, when the data were controlled for students who had been eligible for free school meals, the impact of the SM intervention was greater with an effect size of +0.25 standard deviations. This positive result runs contrary to that of other recent coding initiatives in in England[6]. In addition, there was no evidence of any interaction between the impact of Scratchmaths on CT test scores and gender. This again is important and worthy of further investigation given that girls tend not to engage with computing in comparison with boys (see for example Zagami, et al J, 2015).[7]

There are two reasons why the SM team regard this finding as particularly encouraging. First, it must be recalled that *all* schools are *required* to implement the English National Computing curriculum: so control school students would have engaged in computing according to the national curriculum specifications, and not as is sometimes the case in RCTs where control groups undertake an activity not designed for the same learning goals.

Second, the intervention and control school baseline samples though matched on maths and FSM measures did not show a good balance in their classifications of *school effectiveness*[8], as measured by the English Office for Standards in Education (Ofsted, 2018). 18 of the control schools (35%) were classed as '*outstanding*' compared to only 9 of the intervention schools (17%). In addition, two of the control schools (4%) were deemed likely to be classed as '*requires improvement*' compared with 9 (17%) of intervention schools. This suggests that the intervention schools were likely to find it more challenging to introduce an innovation (like SM) than their matched control schools. Yet this was not the case: the results were in fact better in the treatment schools.

3.2 Fidelity of implementation in the second year and outcomes in mathematics national tests

As in the first-year analysis, the outcomes for each of the fidelity measures are reported separately with the interpretation of coverage and curriculum time as in the first year.

Professional development: 42 out of our 55 schools attended any SM PD, a total of 65 teachers. 34 schools were judged as high fidelity on this measure (62% of the sample) a drop from 54 recorded in the first year, 8 were medium and 13 were low, up from only one in first year.

Technology access: Technology access remained high fidelity.

Coverage: Only 11 schools implemented SM with high fidelity and 6 medium, together representing 63% of the sample a sharp drop from the previous year, (although 9 fewer schools responded to the Y6 survey).

Curriculum time: Only 4 schools were high fidelity and 16 medium fidelity, together 74% of the sample, again a considerable drop from the previous year,

Curriculum progression: For the 31 schools who responded, 27 reported that they followed the progression as set out in the materials.

The independent evaluators reported no impact on mathematical attainment as measured by the Key Stage 2 tests for 11-year olds at the end of Y6. This is a disappointing result, although given the circumstances and the reduction in fidelity as indicated above it is not altogether surprising. We also note the acknowledged differences in the control and intervention school contexts as evidenced by their OFSTED judgements, which might go some way to explain this result: Mathematics is such a high-stakes subject it is likely that only more confident schools would follow through a new approach to mathematics in the year leading up the tests. In fact, the SM team learned from survey data that at least 25 schools had stopped teaching SM as early as January of the second year rather than continuing until the KS2 tests took place in May in order to give space for mathematics revision. In addition, because PD was measured at the school level, it is possible that in a high-fidelity school an individual teacher teaching the Y6 SM curriculum may not have participated in any SM training or received any school-based professional support due to changes in staffing[9] as illustrated in the case study below.

4. A high fidelity school: Emerald Primary

The following example highlights that fidelity of implementation is a complex construct potentially oversimplified by the survey data reported above. Consider the case of Emerald Primary,[10] a larger than average two-form entry North London primary school with approximately 8% of students speaking English as their first language. After the school was rated as 'Requires improvement' by its Office for Standards in Education the head teacher enrolled the school in SM as a means of develop computing across the school.

Two Emerald Primary Y5 teachers attended the SM professional development that focused on using the 5E's framework and the core underlying computational ideas from the first three modules. These two teachers taught the majority of the activities from the first three modules but at the end of Y5 only one teacher, Rina, continued to teach SM in Y6.

In the second year, a new teacher, Sally, was appointed to teach the Y6 SM curriculum and to serve as the computing lead for the school. She did not attend any SM professional development before joining the school. An early observation of Sally's teaching revealed that she was *unaware* of the SM teacher materials designed to support teachers in implementing SM. In one observed session, Sally did not understand a core idea of *turn* from the module (the onscreen beetle turns through the exterior angle when drawing shapes) and in another, she had clearly misunderstood how broadcasting was implemented in Scratch[11]. In both cases, she pieced together her lesson from the small knowledge fragments that she did understand. Her pedagogic approach was to explain the code step by step, without using the computer and then to show the correct solution script which the students copied. Sally reflects on her practice saying: *"If someone was dragging you through the grass, and you don't want to be dragged, you try to grab bits of grass to stop you from being pulled! So, I'm trying to remember the content with the pedagogy that I don't have yet. ….. I was trying to grab any blades of grass that would stop me."*.

By contrast, Rina had attended all of the SM professional development and taught the Y5 SM modules. She enacted the planned sequence of activities from the teaching materials consistently using the 5E pedagogical approach. For example, in one teaching sequence the students built a script shown to the class in a PowerPoint slide, then *Explored* the script by running it and then tried to *Explain* what was happening. Finally, the students were brought together for demonstration and *Exchanging* of results. Rina's lessons were video-recorded and then viewed by both teachers as an object for stimulated recall during post-lesson discussion. Sally cites this as an important turning point in the development of her pedagogic approach and her own learning. Sally's later lessons included adaptations and additions to the SM materials using multi-modal representations, for example moving a cut-out beetle on the white board to illustrate beetle turns. In terms of her own learning, there appeared to be two main developments. First, she saw CT as a legitimate and complex curricular challenge, rather than a prescribed set of targets, and second, she was beginning to appreciate its complexity, recognising that *"there aren't enough people who really know enough about CT and the very delicate elements of it…."*. Critically, she had begun to appreciate the importance of seeing how each block worked individually or together in a program:

> *"Scratch in many ways has become an app which kids go on to make games. ……., they play on it, they can move the sprite.* ***But the relationships between the blocks, how they link, how they work, and how they are manipulated to get an outcome, they don't have a true understanding of that."*** (emphasis added*)*.

Sally's case is significant in capturing the change in teacher knowledge as much as change in pedagogy. Sally was coming to see the central elements of computing as a sophisticated network of far-from-arbitrary rules and procedures, while taking on board a pedagogic approach that allowed students time to explore, reflect and explain.

5. Findings and discussion

We begin by returning to the significant positive effect of the SM intervention on CT scores as measured by the test used at the end of the first year of the trial. As far as we know, this is the first reported effect of its kind: for example an effect was not found in similar study in England, Straw, Bamford and Styles (2017). A still more significant outcome was the greater increase in CT score for educationally disadvantaged students, as measured by the standard FSM proxy. In short, disadvantaged students stood to gain most from SM**,** raising attainment in CT beyond that of the control group. The sample of schools were matched only on test scores and FSM, and as reported above, the control schools had much higher OFSTED ratings than the treatment schools, which makes the effect even more striking. What could account for this?

We speculate SM provided a systematic, progressive research-based curriculum to address computing, whereas the control schools were likely to have experienced a less-structured and possibly incomplete approach. The PD curricula and the teacher materials were developed following extensive design research, which revealed fundamental pedagogical challenges, such as student appreciation of algorithms (Benton, Kalas, Saunders, Hoyles, & Noss, 2018). SM was also popular: the independent evaluators remarked that "Teachers who sustained participation were, in general, positive about the quality of the professional development and materials, particularly in Y5" (Boylan et al., 2018). Anectodally, our school visits suggest that students typically labled "less able" thrived in SM. In one such incident, a young girl labelled by the teacher as 'not very good' become one of the most inventive students in her SM class—showing outstanding creativity and desperate to share her work with the two researchers.

We now turn to the outcome of finding no impact of SM on mathematics attainment as measured by the KS2 test results. It does seem clear that SM implementation was impeded, particularly in Y6, by two factors outside of the control of the innovation and more or less independent of it: *high stakes testing in mathematics*, and *teachers teaching SM with little or no professional development*. Related to the first point, the survey data showed that the fidelity of the implementation in the second year dropped dramatically as evidenced by curriculum coverage and time, and the limited engagement in professional development. The notion of fidelity is admittedly a rather crude measure, not least as fidelity was self-reported[12]. Observations from school visits made it clear that SM time was negatively impacted by a focus on high-stakes testing in mathematics at the end of the school year, with more and more time given to revision and practice. We also surmise that teachers felt more able to adopt novel curricula and techniques for a 'new' subject, computing, rather than change their practice in an established and higher-stakes subject, mathematics.

As to the second point, 42 of the 110 SM Y6 classes, that is approximately 1050 students, were in the trial but may not been taught SM, or may have been taught by teachers with neither experience of Year 5 SM, nor exposure to SM professional development. There were two reasons for this. First, there is considerable teacher 'churn' in England (teachers move to teach different year groups, move schools, and even move out of the profession). Second, the design only allowed for the attendance of two teachers per year group available at PD events. So when the schools were larger (up to 4-form entry), it was inevitable that 'untrained' teachers would teach SM. This situation was exacerbated in that even in high fidelity schools, teachers within the same school did not always support each other, as illustrated in the case study.

The SM evaluation was undertaken almost concurrently with the first implementation of SM: so most if not all of the teachers were new to programming. Thus the importance of the PD cannot be over-estimated. Where PD was taken seriously by schools as in the first year of the innovation, implementation tended to be successful. However by contrast, many teachers sent by schools to the second year SM PD sessions were newcomers, and as shown in the fidelity data, many schools sent no Y6 teachers to PD. It is likely that these 'SM novice' teachers would not be familiar with the computional concepts from the first year themselves, nor know how to build on what the students had experienced the year before. Thus without considerable time on PD in or out-of-school, it is hard to see how this group of teachers could implement SM effectively.

We recognise that there is a need for more intensive and systematic classroom research to explore SM classroom implementation in more detail and document how it evolves over time as it becomes embedded in practice. and to track the engagement of different groups of students (for example, girls, 'low attainment'). It is clear that adopting a SM approach to mathematics teaching is challenging. SM provided detailed lesson plans, with a thoroughly designed strategy of gradually building the need for new computational constructs to explore mathematical ideas and a programme of professional training through the 5Es strategy. We wanted teachers to encourage student thinking about their programs and thinking *with* the programs in mathematical and computational activities. Asking teachers to learn programming and new ways to think about and teach (Jones, 2015) is clearly an enormous task. But one we would argue that needs to be addressed seriously if constructionism is 'to work'.

So we end with one of the many positive stories of student engagement with SM that reassures us of the enduring value of the constructionist vision. One girl delighted in the dynamic display following an activity involving angles and polygons exclaimed: "*What we really like …is when you press that start button and you see your script come into life, it's like magic in front of your eyes*".

Acknowledgements

We thank the Education Endowment Foundation for funding this work as well as our SM colleagues Dave Pratt and Johanna Carvajal for their invaluable contributions to the intervention design. We also acknowledge the invaluable insights provided by the ScratchMaths independent evaluation report, undertaken by Sheffield Hallam University. We are also extremely grateful to the teachers and pupils at all of the SM project schools for their continued engagement, hard work and enthusiasm in trialling our intervention, and participating in our research.

References

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging Primary Programming and Mathematics: Some Findings of Design Research in England. *Digital Experiences in Mathematics Education*, 1-24. doi:10.1007/s40751-017-0028-x

Benton, L., Kalas, I., Saunders, P., Hoyles, C., & Noss, R. (2018). Beyond jam sandwiches and cups of tea: An exploration of primary pupils' algorithm ‐ evaluation strategies. *Journal of Computer Assisted Learning*.

Boylan, M., Demack, S., Wolstenholme, C., Reidy, J., & Reaney-Wood, S. (2018). *ScratchMaths Evaluation Report*. Retrieved from

Brown, A. L., & Campione, J. C. (1996). *Psychological theory and the design of innovative learning environments: On procedures, principles, and systems*: Lawrence Erlbaum Associates, Inc.

Dane, A. V., & Schneider, B. H. (1998). Program integrity in primary and early secondary prevention: are implementation effects out of control? *Clinical psychology review, 18*(1), 23-45.

Department for Education (2013). Primary National Curriculum until 2014 - English: Key stage 2.

Hoyles, C., & Noss, R. (1992). *Learning Mathematics and Logo*. Cambridge, MA, USA: MIT Press.

Lewis, C. M. (2010). *How programming environment shapes perception, learning and goals: logo vs. scratch*. Paper presented at the Proceedings of the 41st ACM technical symposium on Computer science education.

Noss, R., & Hoyles, C. (2017). Constructionism and Microworlds *Technology Enhanced Learning* (pp. 29-35): Springer.

OECD. (2015). *Students, Computers and Learning*. OECD Factbook 2015-2016: Economic, Environmental and Social Statistics, OECD Publishing, Paris. http://dx.doi.org/10.1787/factbook-2015-en

Ofsted. (2018). *School inspection handbook: Handbook for inspecting schools in England under section 5 of the Education Act 2005*. Retrieved from https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/730127/School_inspection_handbook_section_5_270718.pdf

Papert, S. (1980). Mindstorms: Children, Computers, and Powerful Ideas.

Peyton Jones, S. (2015). Code to Joy. *Times Educational Supplement*.

Resnick, M. (2012). Reviving Papert's dream. *Educational Technology, 52*(4), 42-46.

Straw, S., Bamford, S., & Styles, B. (2017). *Randomised Controlled Trial and Process Evaluation of Code Clubs*. Retrieved from Slough:

Zagami, J. Boden, M., Keane, T, Moreton, B. & Schulz, K (2015) Girls and Computing: Female participation in computing in Schools *Australian Educational Computing, 2015, 30*(2).

---

[1] For example, events, animation, control structures, variable, operators and expressions.

[2] All SM materials are available at ucl.ac.uk/scratchmaths.

[3] Fidelity of implementation within an education context has been defined as “the determination of how well an intervention is implemented in comparison with the original program design during an efficacy and/or effectiveness study” (O’Donnell, 2008).

[4] The framework was introduced in the professional development days: for example, in the first year around the computational concepts, *direct drive, scripts, making new blocks, randomness* and *broadcasting*.

[5] Following randomisation, data on the mathematics outcome was obtained from 5,818 students

[6] For example, no effect on computational thinking was found in the evaluation of the Code Clubs in England (Straw, Bamford and Styles, 2017).

[7] It is worth noting that the ScratchMaths project was given ‘5 padlocks’, the funder’s (Education Endowment Fund, EEF) hallmark for the most robustly conducted research.

[8] OFSTED judges schools on a 4-point scale: outstanding, good, requires improvement and inadequate. For details of OFSTED inspections and their data methods, see https://www.gov.uk/government/organisations/ofsted). All schools are required to make their latest OFSTED report available via their website.

[9] In addition to teacher movement, some treatment schools had more than a 2-form entry so inevitably given only 2 teachers could be trained per school some teachers would not have engaged in the PD.

[10] All school and teacher names have been changed.

[11] Broadcasting is a Scratch metaphor for understanding how sprites can communicate.

[12] The SM team estimate that there was less coverage in Y6 than reported, as teachers were clearly concerned to put a positive gloss on their school’s engagement.