International Journal of Computer Science Education in Schools, April 2020, Vol. 3, No. 4
ISSN 2513-8359

The Relationship between Executive Functions and Computational Thinking

Judy Robertson[1]
Stuart Gray[2]
Toye Martin[1]
Josephine Booth[1]

1 University of Edinburgh
2 University of Bristol

DOI: 10.21585/ijcses.v3i4.76

Abstract

We argue that understanding the cognitive foundations of computational thinking will assist educators to improve children's learning in computing. We explain the conceptual relationship between executive functions and aspects of computational thinking. We present initial empirical data from 23 eleven year old learners which investigates the correlation between assessments of programming and debugging in the visual language Scratch and scores from the BRIEF2 assessment of executive functions. The initial data shows moderate to large correlations between assessments of debugging and programming with the BRIEF2 teachers' rating of executive function as manifested in classroom behaviour. Case studies from the empirical data are used to qualitatively illustrate how executive functions relate to a game making task. We discuss the implications of these findings for educators, and present suggestions for future work.

Keywords: computational thinking, executive functions, primary school

1. Introduction

The recent Royal Society report on computing education in UK schools reviewed the landscape after major curricular reform in which computing lessons became a requirement for learners aged 5 and older in England and Wales (Royal Society, 2017). While it welcomes the changes (which it was partly responsible for instigating) and notes that there are pockets of excellence, it identifies that computing education is still "patchy and fragile" (Royal Society, 2017, p. 6). The report demonstrates the curriculum changes are not enough; they must be supported by high quality teacher education and computing education research. It proposes that a research agenda in the UK should focus on the questions: "What is the most effective, best-evidenced curriculum framework for computing? …Which specific instructional techniques and teaching strategies are most effective for raising attainment in computing?"(Royal Society, 2017, p. 95). These are also open questions within the international research community. In order to answer these questions, however, we need to further develop our understanding of the cognitive and psychological skills which underpin different aspects of computational thinking, and how these develop throughout childhood. This paper focuses on the potential link between computational thinking and underlying executive functions.

Much work has been done on defining computational thinking (also referred to as CT) and its component skills. In this paper, we use the Royal Society's clear and succinct definition of computational thinking: " the process of recognising aspects of computation in the world that surrounds us, and applying tools and techniques from Computer Science to understand and reason about both natural and artificial systems and processes" (The Royal Society, 2012, p. 12). A review of computational thinking research identified the following core computational

thinking elements: abstractions and pattern generalizations (including models and simulations); systematic processing of information; symbol systems and representations; algorithmic notions of flow of control; structured problem decomposition (modularizing); iterative, recursive, and parallel thinking; conditional logic; efficiency and performance constraints ; debugging and systematic error detection (Grover & Pea, 2013). In addition, "programming is not only a fundamental skill of CS and a key tool for supporting the cognitive tasks involved in CT but a demonstration of computational competencies as well." (Grover & Pea, 2013, p. 40)

There has been debate about the extent to which computational thinking can be distinguished from the other sorts of thinking which children learn at school. Although historically some doubt has been cast on the nature of the relationship between programming and problem solving (Palumbo, 1990), recent evidence synthesis reveals that learning to program can improve scores on other measures of problem solving. A meta-analysis of studies which explore the transfer of programming skills to general problem solving found an overall transfer effect of g =0.49, with a transfer to mathematical reasoning of g=0.57 (Scherer, Siddiq, & Sanches Viveros, 2018). It is also likely that programming and other computational thinking abilities are enabled by well-researched lower level psychological processes. Grover and Pea make the case that while there might be overlap between computational thinking and other STEM problem solving approaches, it was recognisably and crucially absent from previous curricula (Grover & Pea, 2013). Recent empirical work supports this; while computational thinking is predicted by other cognitive abilities, it appears to some extent to be an independent construct. In a study of 1251 Spanish school students, Román-González and colleagues investigated the relationship between computational thinking (as measured by their CTt instrument) and other cognitive abilities as measured by the Primary Mental Ability and RP30[1] Resolucion-de-Problemas problem solving standardised psychological tests (Román-González, Pérez-González, & Jiménez-Fernández, 2017). They found a high correlation (r=0.67) between general problem solving ability and computational thinking scores. In a regression model, spatial ability and logical reasoning as measured by RP30 problem solving tasks were significant predictors of the CTt scores, explaining 27% of the variance. The authors interpret the high proportion of unexplained variance to suggest a "certain independence of CT as a psychological construct, distinct from the traditional aptitudes" (Román-González et al., 2017, p. 9). They recommend that further research should relate computational thinking with other cognitive abilities including working memory and other executive functions (also referred to as EF).

This paper pursues this line of research by exploring the relationship between two important aspects of computational thinking (programming and debugging) and executive functions. It begins with an explanation of executive functions, the role they play in academic success, and the reasons why they are likely to be related to computational thinking. This is followed by an account of how executive functions (as measured by the Behavior Rating Inventory of Executive Functioning-2 (BRIEF2) instrument) may map to creative programming and debugging. Having made the case that specific EF skills are likely to underpin these aspects of computational thinking, the paper then reports on an empirical study to investigate this issue. Data on 23 eleven year-old learners' creative programming and debugging performance, and EF abilities was gathered. Results of a correlation analysis do indeed support the case that EFs are related to computational thinking. Case studies from the empirical data are used to qualitatively illustrate how executive functions relate to a game making task. The paper concludes with some recommendations for practitioners about how this might affect classroom decision making, and suggestions of future research work.

2. Literature Review

2.1 Executive Functions and the Link to Computational Thinking

Executive functions (EF) is an umbrella term for higher order cognitive functions linked with the frontal lobes of the human brain (Aron, Robbins, & Poldrack, 2004) and include abilities such as inhibiting impulsive responses, the ability to hold and simultaneously manipulate information in mind (known as working memory), attention shifting (or cognitive flexibility), planning and risk taking (Diamond, 2013; Miyake et al., 2000). EF serve as general purpose control mechanisms that help modulate human cognition (Miyake et al., 2000), underpin self-control (Denckla, 1996; Pennington & Ozonoff, 1996) and are commonly implicated in problem-solving or goal-directed-behaviour (Luria, 1966). These functions mature at different rates through childhood and into adolescence (Dolan & Molen, 2006).

Executive functioning and educational attainment in the primary school age-range has also been linked with

[1] http://web.teaediciones.com/RP30--Resolucion-de-Problemas---Problem-Solving.aspx

metacognition (MC) (Bryce, Whitebread, & Szűcs, 2015), with some studies reporting a link between MC and problem solving in computer science learning specifically (Allsop, 2019; Parham, Gugerty, & Stevenson, 2010). Parham and colleagues' analysis of a think-aloud study with eleven college-level programmers indicates that the meta-cognitive strategies of checking/comparing code and stating/revisiting goals were commonly used mental processes. In a study of a class of 11 year learners, Allsop noted the use of metacognitive practices to control and regulate programming activities including planning, monitoring and evaluation (Allsop, 2019). Whilst related constructs in children's thought and action (Lyons & Zelazo, 2011), EF and metacognition have been based in different research and theoretical traditions in the psychological literature (Roebers, 2017a). In this study, for the sake of clarity, we have chosen to focus on EF.

We believe that the link between EF and CT is worth exploring for two reasons: 1) EF is a predictor of academic success in general, including in the development of mathematical skills and science learning (Cragg & Gilmore, 2014) so it is reasonable to assume that they are also required in CT; and 2) conceptual analysis of the processes involved in programming and debugging predict that cognitive regulation aspects of EF are required. However, further empirical evidence of the relationship is required; this paper makes an initial contribution by providing the results of an exploratory study in a primary school classroom.

Programming and debugging are not the only components of computational thinking; it is likely that EFs are also implicated in other aspects. We have chosen to start with these components because they are commonly taught in classrooms in the UK and other countries internationally, and there are extensive online teaching materials to support them. Other aspects of computational thinking should be the subject of future research.

2.2 EF as a Predictor of Academic Success

Executive functions are implicated in a wide range of areas of academic learning and attainment. For example, in reading (Altemeier, Abbott, & Berninger, 2008), maths (Gilmore et al., 2013) and science (St Clair-Thompson & Gathercole, 2006). They are also predictive of school achievement more generally (Bull & Scerif, 2010; McLean & Hitch, 1999; St Clair-Thompson & Gathercole, 2006; Titz & Karbach, 2014) as well as university achievement (Knouse, Feldman, & Blevins, 2014), and job success (Daly, Delaney, Egan, & Baumeister, 2015). Some interpret these findings to suggest a domain-general relationship between EF and school attainment (Best, Miller, & Naglieri, 2011), a growing number of studies highlight a particularly important role for EF in relation to learning in STEM subjects (St Clair-Thompson & Gathercole, 2006; Van der Ven, Kroesbergen, Boom, & Leseman, 2012), although to our knowledge, no previous studies have considered the relationship between EF and CT.

Neuroscience and education research indicates that executive skills play a critical role in developing mathematical proficiency, particularly updating and manipulating working memory, inhibition and shifting (Cragg & Gilmore, 2014). Given Weintrop and colleagues' detailed argument exploring the reciprocal relationship between computational thinking and maths and science learning (Weintrop et al., 2016), and the meta-analysis results which indicate programming improves mathematical test scores (Scherer et al., 2018), there is good reason to investigate the relationship between computational thinking and executive skills. The executive skills which support the development of maths proficiency are also likely to play a role in developing computational thinking. Indeed, empirical evidence from design based research with middle school children confirms that the outcomes of a course in computational thinking were predicted by maths performance (Grover, Pea, & Cooper, 2015).

2.3 Conceptual Analysis Of Efs Involved In Programming and Debugging

2.3.1 Why Efs Are Required For Creative Programming Tasks

The playful constructionist approach advocated by the Scratch creators, based on Papert's intellectual legacy, encourages creativity and self-directed exploration. Following the footsteps of Logo, Scratch was designed as a constructionist environment to support a spiral of creativity, in which learners "imagine what they want to do, create a project based on their ideas, play with their creations, share their ideas and creations with others, and reflect on their experiences—all of which leads them to imagine new ideas and new projects"(Resnick, 2007, p. 18) . Brennan and Resnick describe design in the creative programming context as "an adaptive process, one in which the plan might change in response to approaching the solution in small steps". They describe a process of "iterative cycles of imagining and building, developing a little bit, trying it out then developing further, based on their experiences and new ideas."(Brennan & Resnick, 2012, p. 7).

The merits of pure discovery learning (for example in Papert's constructionist work with Logo) have been questioned (Mayer, 2004). Previous researchers have argued that "learners often struggle with algorithmic concepts, especially if they are left to tinker in programming environments, or if they are not taught these concepts using appropriately supportive pedagogies." (Grover et al., 2015, p. 205). That is, unstructured programming tasks by themselves may not improve computational thinking. Grover et al. recommend minimally guided discovery learning for computational thinking courses for middle school learners (Grover et al., 2015), including approaches such as scaffolding, cognitive apprenticeships, code reading and tracing, and modelling the process of decomposition. They aim to enable children to build on computational concepts which they have been taught, in a way which fosters creativity and ownership.

The complexity of creative programming tasks can greatly challenge novices. The individual computational thinking skills can be difficult to acquire in themselves, but the higher order executive function skills place additional load on the learner, particularly in terms of planning and self-monitoring. The learner must decide what to make, how to make it and be able to monitor her own progress in reaching her goals. Depending on the stage of the learner, it may be the case that the requisite planning and monitoring executive functions are still developing. In addition, the task places a load on working memory because the programmer must hold in mind the end goal, and steps needed to achieve the goal. Inhibition is also required to avoid distractions from the goal.

Papert (Papert, 1991) and later Resnick and Brennan (Brennan & Resnick, 2012) favour a less top down approach to planning in which the programmer is: "guided by the work as it proceeds rather than staying with the pre-established plan" (Papert, 1991, p. 3). A learner working in this way would still task-monitor periodically to evaluate whether the current code produces a desirable output, and if not, decide what changes are needed and map out the sub-steps to get from the current state to the desired state. While open ended creative tasks can be fun when appropriately challenging, it may be overwhelming and frustrating for some learners unless they are adequately supported. For learners with developing EF skills, it may be difficult – and demoralising – to engage with such tasks. With appropriate support with planning and monitoring, however, working on motivating programming tasks may be one approach to EF skills development. It is possible that the requisite EF and programming skills can be further developed alongside each other in creative projects, if the learner has previously had support to develop both programming skills and planning and monitoring skills in other contexts.

2.4 Why EFs Are Required For Debugging Tasks

Resnick and Brennan describe computational practices relating to solving and anticipating problems. An interviewee described her debugging activities as "identify the source of the problem, read through the scripts, experiment with scripts, try writing scripts again, find example scripts that work, tell or ask someone else, take a break…" (Brennan & Resnick, 2012, p. 7). Rich and colleagues present a helpful literature syntheis and learning trajectory of debugging for children (Rich, Andrew Binkowski, Strickland, & Franklin, 2019). They identify strategies documented in the literature for finding and fixing errors including: hypothesising and testing theories about the cause of a problem, and deciding how to change a program when it does not produce the intended results. These behaviours are likely to rely on underpinning executive function capacities such as working memory and cognitive flexibility ("the ability to shift between response sets, learn from mistakes, devise alternative strategies, divide attention, and process multiple sources of information concurrently" (Anderson, 2002, p. 74)). Rich et al. observe that emotional regulation and the ability to preserve in the face of failure is a requirement for successful debugging; emotional regulation is an aspect of executive function which is assessed by the emotional control subscale within the BRIEF2 instrument which we used in this study. In addition, Rich and colleagues note that at the end of the learning trajectory, learners become aware that debugging techniques can be chosen strategically. The ability to evaluate and slect the best strategy for a task requires well developed executive function capacities (Roebers, 2017b).

In order to systematically detect errors in code, the learner must have the ability to understand the decription of the incorrect program behaviour and why it is different from the required behaviour, and be able to develop and follow a plan of detecting, fixing and testing which places high demands on working memory, as well as the ability to switch between tasks and switch back and forth between different representations. This last point is particularly salient in a visual language like Scratch where the user must look at the visual behaviour of a sprite on screen, compare it to a mental representation of what the ideal behavior would look like, and then switch to a different visual representation of code blocks in order to fix the problem.

The programmer must think of possible reasons why the program is not working, prioritise which is most likely to begin with, pinpoint where the error would occur in the code, identify whether it is actually present in the code

and if so, fix it. If that particular error was not present, or if it was present but fixing it did not result in the target code behaviour, the programmer must move on to consider another possible reason for the flaw. Beginner programmers are disadvantaged because they do not have the experience to quickly identify or prioritise possible errors. From this point of view, the debugging exercises used in this study were designed to start with examples where the learner need only solve one bug at a time and progress to finding and correcting multiple bugs.

3. A Classroom Study

We conducted an initial study with a class of 11 year old children to begin the process of testing the theorised link between EF and CT with empirical data. The aims of the study were to quantiatively explore the relationship between EF and CT, specifically: the overall relationship between creative programming and debugging with aspects of executive function, as measured by BRIEF2. A further goal for this study was to use case studies from the empirical data to qualitatively illustrate the relationship between EF and CT.

3.1 Participants

This study involved twenty-five children (16 boys, 9 girls) aged 11-12 years from the same Primary 7 class at a Scottish Primary School. All children had experience of programming using Scratch as part of their computing curriculum work. The group were recruited by their class teacher and received written information sheets and consent forms (in age appropriate language) two weeks before the study to be read, signed, and returned before they could take part in the sessions. The ethical procedures were approved by the [*blank for review*] ethics committee. Of the twenty-five children who gave their consent to be part of the study, we were able to collect data on all EF/CT measures for twenty-three.

3.2 Data Collection

The following data was collected in this study.

Executive functions

The Behavioural Rating Inventory of Executive Function (BRIEF2) is a rating scale used to assess everyday behaviours associated with executive functions at home and school. It is considered to be an ecologically valid method of assessing the extent to which individuals are able to successfully pursue their own goals in complex every-day problem solving tasks (Toplak, West, & Stanovich, 2013) . It is used for clinical assessment of children for whom there may be concerns about self- regulation (e.g those with autistic spectrum disorders, attention disorders, depression and other conditions). The BRIEF2 is a questionnaire completed by teachers about individuals' behaviour and emotional regulation, aspects of EF which are also important for classroom learning. A recent review of BRIEF2 considered it to be a theoretically and psychometrically sound measure of executive functioning for children and adolescents (Dodzik, 2017), with internal consistency for the teachers form in the range of alpha coefficient =0.88 to 0.98, and a test/re-test reliability of 0.82.

In this study, the BRIEF2 teacher rating scale is used to assess behaviours which might impact on typically developing children's ability to complete complex creative programming and debugging tasks. BRIEF2 has three indices: behavioural regulation (consisting of inhibit and self-monitor scales), emotion regulation (consisting of shift and emotional control scales) and cognitive regulation (consisting of initiate, working memory, plan/organise, task-monitor, and organisation of materials scales).

The class teacher filled in a 63 item scale for each pupil, indicating whether the statement is true of the child *never* (scored as 1), *sometimes* (scored as 2), or *often* (scored as 3). The raw score was then converted to a T-score which is normalised for age and gender according to. T-scores range between 36 and 90 for 11-13 year old girls and between 37 and 88 for 11-13 year old boys (Gioia, Isquith, Guy, & Kenworthy, 2015). Higher scores indicate higher level of difficulty in a specific domain of executive function. The Global Executive Composite score is reported here as it is considered as a useful summary measure.

Creative programming task: We used an automatic assessment of aspects of computational thinking as manifested in the source code of a Scratch program collected from the participants (flow control, data representation, abstraction, user interaction, synchronisation, parallelism and logic). The code was analysed using Dr Scratch, software which performs static analysis of Scratch source code (Moreno-León & Robles, 2015). Assessment from Dr Scratch has shown to be consistent with other software metrics of code complexity (Moreno-Leon, Robles, & Roman-Gonzalez, 2016) and correlate strongly with the assessments of expert human

evaluators (r=0.82) (Moreno-León, Harteveld, Román-González, & Robles, 2017). Overall scores range between 0 and 21, with higher scores indicating higher proficiency.

Debugging task: A set of 7 custom Scratch debugging tasks were developed by the authors, based on the Debug It! Exercises available on Scratch Studio [1]. The seven exercises required debugging examples involving conditionals, fixed loop, variables, conditional loop, parallelism (simple problems), and two integrated examples which brought together several of these concepts (complex problems). Participants were given a specification of what an example Scratch program should do, a description of the buggy behaviour of the code when it runs, and the code itself. They were then asked to locate and fix the error. Each simple problem was scored with 2 points for a complete solution, and one point for a partial solution (where a clue had been given), while the two complex problems which had a maximum score of 6 points because there were multiple bugs. The maximum overall score was 22, with higher scores indicating higher proficiency. The tasks were scored by the first and second authors; discrepancies in the scores were resolved through discussion.

3.3 Procedure

Data collection sessions were undertaken by the second author and a research assistant at the Primary School. These staff members both had up-to-date certification to work with children.

3.3.1 Scratch Creative Programming Session

The creative programming sessions were hosted in the children's classroom using school Windows laptops and facilitated by the second author. He first explained to the children that they were being asked to use Scratch online to individually create a program of their choosing within a 60-minute time limit and gave the children an opportunity to ask questions about the exercise. The children had a written document of a description of the task and instructions to refer to and could ask for clarification and assistance if required.

3.3.2 Scratch Program Debugging Session

The debugging session was again facilitated by the second author but the research assistant was also present to help support the children. The session began with the second author explaining to the children that they were being asked to fix a series of 7 broken Scratch programs within a 60-minute time limit. The children were asked to work alone without discussion.

The children were given a worksheet with a description of each problem, stating what the program should ideally look like (the specification) and the problems with the current version of the program (see supplementary materials). The children also had access to videos of the ideal of each program and the problematic version.

4. Results

4.1 Descriptive Statistics

The descriptive statistics indicate that this class of children had room for improvement in debugging (with an average just slightly over half marks) and creative programming (with a mean under half marks). Note that the number of participants N varies due to student absences on different data collection days.

Table 1. Descriptive Statistics for Overall Measures

	N	Mean	Standard deviation
Debugging score	22	7.5	4.13
Creative programming score	23	9.35	4.53
BRIEF2 T-score	25	51.12	15.55

[1] https://scratch.mit.edu/projects/10437439/

The pearson correlation between the Creative Programming task score and the Debugging score is r=0.6 (95% CI 0.26, 0.82]). This can be interpreted as a large correlation between these two measurements of computational thinking (Cohen, 1992).

4.2 Findings

As shown in Table 3, programs which have higher Dr Scratch scores are produced by children who have lower BRIEF2 scores (i.e. those who have more mature executive functioning): BRIEF2 scores explain 60% of the variance in Dr Scratch scores. Similarly, learners with a greater level of debugging skills have better developed EF skills, explaining 40% of the variance in debugging scores. Focusing on how the Scratch score relates to the BRIEF2 sub-scales, the Behaviour Regulation Index (BRI) correlates with r=-0.69, the Emotional Regulation Index (ERI) correlates with r=-0.57, and the Cognitive Regulation Index (CRI) correlates with r = -0.55. Given that the BRI seems to have the strongest relationship with the Scratch score, it was worth examining the relationship of the further subscales within it: the Inhibit subscale is r=-0.7, and the Self-monitor subscale is r=-0.67. The definition of Inhibit from the BRIEF2 manual is "the inhibit scale assesses inhibitory control (i.e. the ability to inhibit, resist, or not act on impulse), including the ability to stop one's behaviour at the appropriate time." (Dodzik, 2017, p. 33). The definition of Self-monitor is "the self-monitor scale assesses awareness of the impact of one's own behaviour on other people and outcomes". It includes "awareness of one's own effectiveness in problem solving and the ability to monitor important outcomes" (Dodzik, 2017, p. 34).

Table 2. Relationships between Measures of CT and "Reflective" Aspects of EF

Measures	**Pearson correlation r**	**95% CI**
Creative Programming total and BRIEF2 global composite	-0.6	[-0.8, -0.25]
Debugging total and BRIEF2 global composite	-0.4	[-0.70, -0.02]
Creative Programming and BRIEF2 behavioural regulation index (BRI)	-0.69	[-0.86, -0.40]
Creative Programming and BRIEF2 emotional regulation index (ERI)	-0.56	[-0.79, -0.19]
Creative Programming and BRIEF2 cogntivie regulation index (CRI)	-0.55	[-0.79,-0.18]
Creative Programming and BRIEF2 BRI Inhibit	0.7	[-0.86, -0.40]
Creative Programming and BRIEF2 BRI self-monitor	-0.67	[-0.85, -0.36]

4.3 Qualitative Illustrations of How EF Relates to Programming Tasks

It is instructive to examine the games produced by learners with different EF profiles in order to identify sorts of help which teachers could provide when supporting similar tasks in the future (see Table 3 for a summary of their numerical scores).

Table 3. Summary of Case Study Learners' Scores

Participant number	**Sex**	**Age**	**BRIEF2 Total score**	**Creative programming score**	**Debugging score**
P18	M	11	164	1	7
P9	M	11	160	N/A[1]	N/A
P5	M	11	83	0	3
P1	F	12	69	14	12
P11	F	11	63	8	5
P17	F	11	62	12	10

[1] Scratch analyser crashes when this program is run; no score is output.

P18 is an eleven year old boy with one of the highest Behavioural Regulation Index scores in the class, scoring highly on both the inhibit and self-monitor scales indicating serious issues with these areas of functioning. Indeed, P18 has extremely elevated EF profiles across all BRIEF2 subscales. This learner's Scratch file has no code. It is hand-drawn stick man sprite and some vertical black lines on a backdrop. It is possible that the black lines were intended to form a maze like some of the other children's. The game does not convey personal interests or show exploratory behaviour within the tool like some of the other games with very little functionality. This learner has an elevated score in Initiate, suggesting that he finds it difficult to get started on a task. P18 scored 7 out of 21 on the debugging test, which illustrates that he has some basic understanding of Scratch constructs but he was unable to put them into practice. He was unable to marshall his efforts to produce a plan for a meaningful program in an open ended creative task. This learner would benefit from being given a specific task which would consolidate his knowledge of the Scratch concepts which he has been taught.

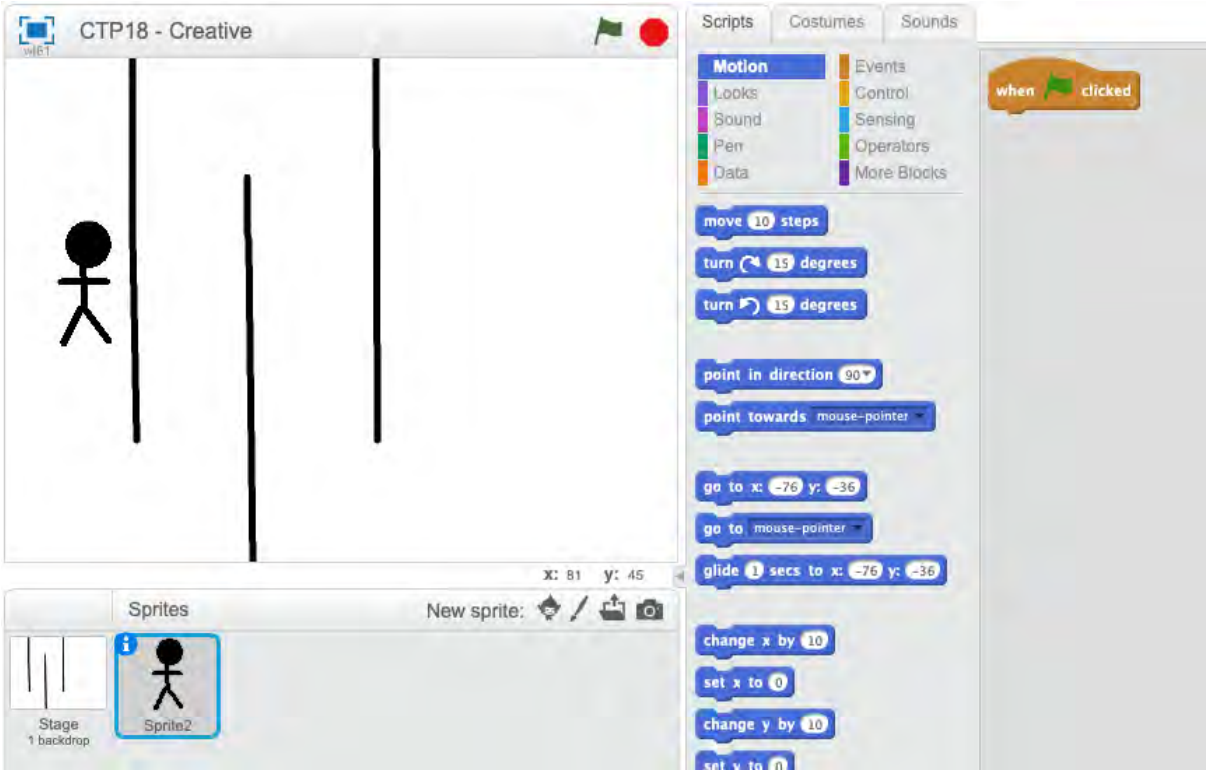

Figure 1. A Screen Shot of P18's Game

Figure 2 shows a screen shot of P9's game. P9 is also an eleven year old boy with an elevated EF score across all the subscales and was joint with P18 in having a particularly high score for the self-monitoring scale again, indicating difficulties across numerous areas of EF. In this game, there a multiple copies of each sprite with the same code copied between sprites. When the code runs, the sprites rapidly rotate and make a noise. The overall effect is strikingly colourful and overwhelming. There is quite a lot of code, but it is repeated sequences of animation instructions and loops which do not serve a clear purpose. There are multiple unnecessary nested loop constructs which suggest that the learner does not understand that only one forever block would have caused the cat sprite to repeatedly miaow. The learner does not appear to have been following a plan to create a particular interactive program, but rather gives the impression of exploring the Scratch interface to produce an entertaining visual result. He was absent on the day of the debugging session so there is no additional information about his Scratch knowledge. While children often find it fun to experiment with visual effects in Scratch when they initially encounter it, this learner would benefit from support in developing a specific goal and identifying how this could be accomplished in Scratch.

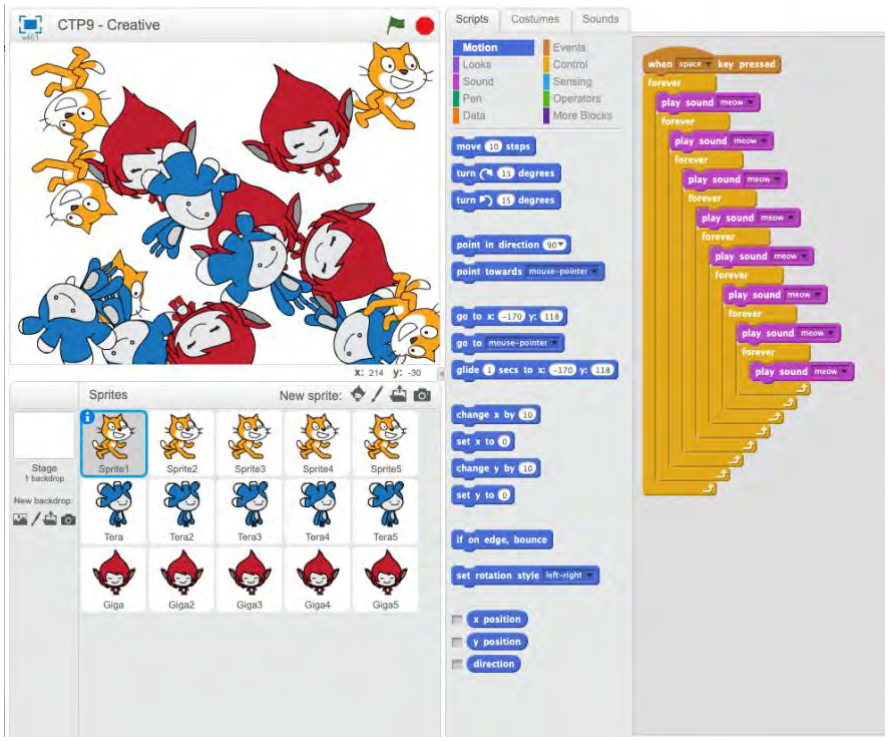

Figure 2. A Screenshot of P9's Game

As an example of a child who did have a plan but who did not write any code, P5 made a backdrop which reads "in this game you have to get your green ball to the yellow ball and you will progress to the next round". This is a plan for a potentially entertaining game, but it has not been implemented at all, with no sprites representing either colour of ball. His debugging score of 8 suggests that he understands how Scratch concepts work although he did not use them in his own game. P5 is in the 59% percentile for his overall EF score, but he has an extremely elevated score for Inhibit which indicates that in general he is distractable and may be diverted from executing plans.

In the case of P1, a twelve year old girl, there is an indication that she thought of a plan but did not prioritise finishing the subtasks which were essential for the user to play the game. According to the instructions for the user, the purpose is to jump up and collect apples from a tree. The event handling code to implement this is not present, although the learner spent time creating scene changes to transfer from a bedroom to an orchard and back. P1 has a mid-range EF score in general, but it is elevated for the Shift subscale. Shift includes the ability to switch or alternate attention and change focus from one topic to another. Here, the learner may have failed to switch attention to the apple gathering mechanic in time to complete the game. The game would have been more successful if the learner had some support in identifying which aspects were essential to get a working prototype of the game completed.

P11 (an eleven year old girl) has developed a plan for her game although her knowledge of Scratch appears to be insufficient to put it fully into practice. P11 has low scores for EF, benchmarked in the 26th percentile for a girl of her age according to the BRIEF2 manual indicating no substantive difficulties with EF in comparison to peers. In the game, there are clear instructions which tell the user to hunt for the apple in each scene. There are multiple scenes, and multiple sprites for decoration. Code is copied across sprites but is not relevant to them. The code does not work because when the user clicks on the apple it switches to only one other room. There is also redundant code with "if" statements which will never execute. Her debugging score of 5 also suggests that her knowledge of Scratch concepts could be improved. In this case, the learner could progress with some support in learning conditional language constructs to help her achieve her initial plan.

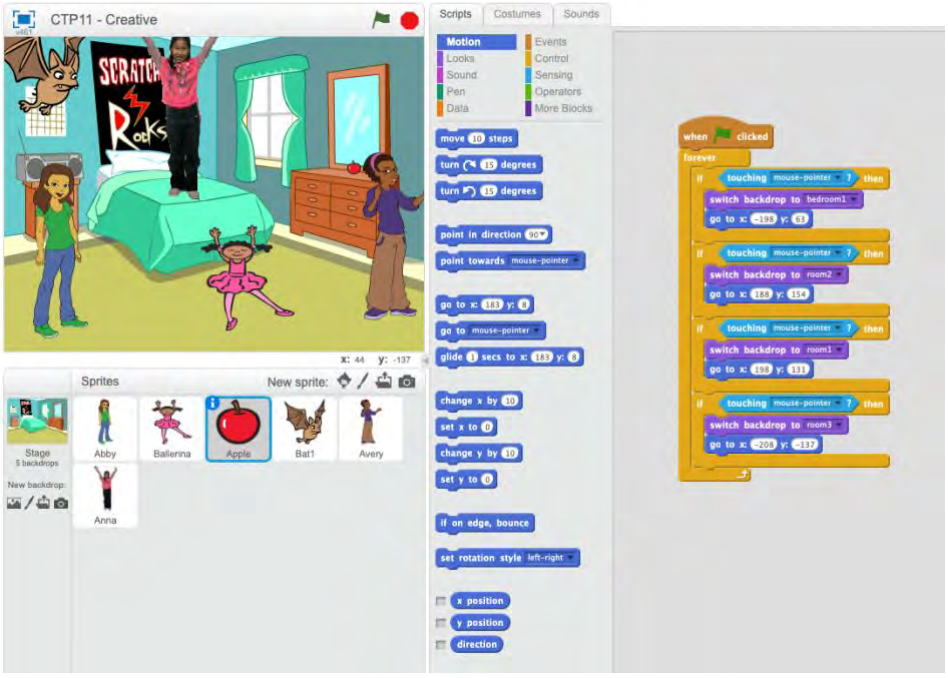

Figure 3. A Screen Shot of P11's Game

Like many of the children, P17 (an eleven year old girl) made a maze game (see Figure 4) but it is unusually well executed. P17 has low EF scores indicating no difficulties in EF (in the 23rd percentile) and her debugging score of 8 indicates that she has some working knowledge of Scratch constructs. The maze game has several levels of progressing difficulty implemented, indicating some advance planning of the maze features. The code is concise and elegant in comparison to that of her peers, using broadcast to generically level up. A step forwards for this learner might be to try implementing more complex game mechanics such as a scoring mechanism which would require additional computational thinking skills.

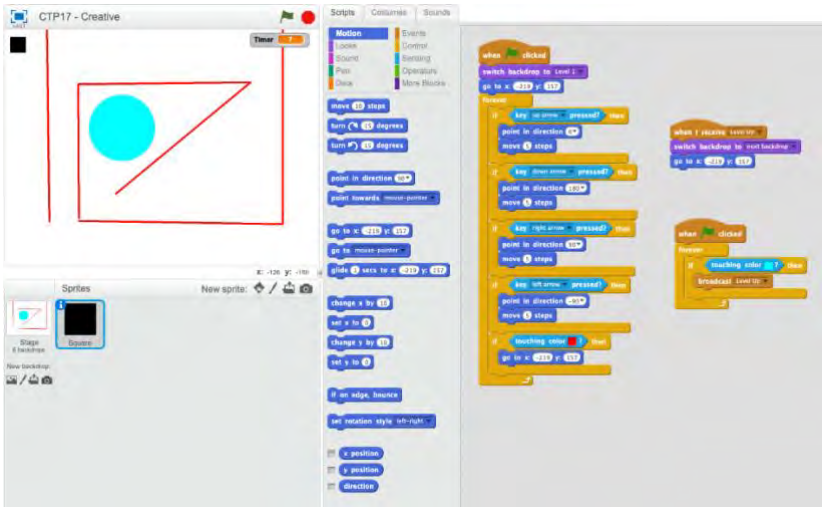

Figure 4. A Screen Shot of P17's Game

5. Discussion and Future Research

Initial empirical results suggest that there is a relationship between EF and both creative programming and debugging. This is consistent with the recent finding that computational thinking assessments correlate with general cognitive abilities (r= 0.67) (Román-González et al., 2017). We do not interpret this to mean that CT is

just "ordinary" problem solving and that the current worldwide emphasis on developing CT is mistaken. Rather, we see it is one step further in establishing the *nomological net* (Román-González et al., 2017) which links CT to other cognitive variables.

The case studies give an indication of how programming skills and executive functions are both required for learners to be successful in open ended creative programing tasks. Our study suggests EF is linked with the CT abilities of children but metacognition (in terms of self-regulation and self-monitoring) may also contribute to the ability of children to engage in CT. Our findings from a standardized test of EF and analysis of the code produced by the children along with their debugging performance are complementary to previous evidence about children's metacognition when programming. Allsop's study in a primary school classroom (in a similar educational context and age group) triangulated evidence from semi-structured interviews, learner journals, observations as well as completed games to examine the planning, monitoring and evaluation skills used by the children (Allsop, 2019). As with our case study learners, Allsop found that debugging required a degree of skill in monitoring and evaluation. Interestingly, the learners in her study reported that their methods of planning their games (such as sketching) transferred into their ability to plan for other learning domains. Allsop's methodology gives some insight into the children's thought processes which is beneficial, although in our case the use of standardized tests of EF alongside the assessment of programming enabled us to investigate the relationship between EF and CT. Future research could further examine the impact of MC in the emerging EF-CT relationship.

An important finding from Allsop's work is that the children in a natural classroom setting used language as an instrument for making decisions, evaluating and regulating activities in conversation with their peers. This suggests that attempts to objectively assess individual performance on creative programming and debugging in a "test" situation where peer collaboration does not occur may underestimate the children's capabilities. That is, Scratch code and debugging scores from "test" conditions reveal what the learner is capable of without the assistance of a more able peer, whereas more naturalist methodologies show what the learner is able to do within their zone of proximal adjustment (i.e. the human and artefact resources that are the most appropriate form of assistance for a given learner at a particular moment in time (Luckin, 2008)).

As this was exploratory work, further studies are required to confirm these findings, and establish the direction of the effect. This study should be replicated with a larger sample size to establish whether the relationships between these variables hold. While the automated assessment of Scratch programs using Dr Scratch is convenient, hand analysis of the games suggests that there are discrepancies between the inclusion of a code construct and its correct usage. Future versions of Dr Scratch may address such problems such as the inclusion of "dead" code which is never used.

There have been recent calls for improvements in standardised methods for assessing computational thinking (Román-González et al., 2017). This study focussed on programming and debugging, by analysing the product of these processes (program source code). Future work could use assessments of wider computational thinking skills, perhaps using tools such as the CTt (Román-González et al., 2017). We would anticipate that individual test items would be shorter and they would require less sustained concentration on a single problem than an open-ended programming tasks.

6. Implications for Educators

The results of this pilot work may be of benefit to classroom teachers who are planning how to support their learners during Scratch projects. It would appear that learners who have difficulties with executive functions, particularly behavioural regulation (including not being able to successfully self-monitor or inhibit their behaviour) could find an open creative task challenging. For such learners, it could be beneficial to introduce a variety of support mechanisms for a zone of proximal adjustment (Luckin, 2008). The teacher could introduce regular check points where each class member reflects (perhaps with a classmate) on the extent to which they have achieved their initial plan. Some learners may need help to come up with a coherent plan in the first place (beyond exploring the interface to find serendipitous effects), and could potentially benefit from lessons in structuring and prioritising the required steps to achieve an initial game idea. This can be scaffolded through activities in which the learners devise a plan to implement a given specification, or are asked to prioritise a list of tasks which need to be achieved to make a specified game.

Other learners in this study appeared to have the executive skills to conceive, execute and monitor a plan but lacked the knowledge of programming language constructs to carry it out. A possible remedy for this would be to recommend project specifications to learners to develop their current level of Scratch knowledge e.g. a learner

who has successfully implemented a maze game could be encouraged to try developing a scoring system for it once they have studied a lesson on variables.

While teachers are skilled in scaffolding tasks according to learners' stages for other areas of learning, they may find it harder to achieve if they are themselves new to programming and have not yet developed an intuition for how difficult tasks are to achieve in the target programming language. This emphasises the need for learning materials which are clearly graded in terms of computational thinking difficulty.

If future work was to confirm a strong relationship between EF and aspects of CT, what would be the implications? Firstly, it would be useful for those designing curricula and teaching materials for CT, because previous empirical results about the developmental trajectory of EF would give some guidance of the stage at which it would be appropriate to introduce particular problem solving tasks in CT. Well established empirical results about the development of the ability to look ahead when planning, the development working memory and shifting could be applied. These could be used to design external representations, software scaffolding or pedagogical approaches which would assist learners whose EFs are still developing.

Secondly, knowledge of how certain types of CT task rely on secure EFs could help teachers to plan tasks which are appropriate to their learners. Research into teachers' knowledge of executive functions in mathematics learning illustrates that experienced teachers are aware of the importance of working memory, inhibition and shifting from observation during their practice (Gilmore & Cragg, 2014). However, Gilmore and Cragg found that it may take some years for this understanding to develop and student teachers may not encounter these concepts during their studies. It is therefore important that developers of classroom learning materials think carefully about the executive function demands of their programming and debugging activities and indicate clearly the stage of learning for which each task is suitable. Plain language indications of the underlying skills would also be helpful (such as the statements used in the BRIEF2 tool), as Gilmore and Cragg found that even the experienced teachers who understood the concepts were not familiar with the technical terms from the psychology literature. An introduction to EF and how it relates to CT could be a useful part of initial teacher education programmes.

Lastly, it is possible that CT activities could be a motivating and engaging way to help learners improve their EFs. Because EF is a predictor of life success, academic success and health in later life, interventions which successfully improve EF in young learners are very valuable (Diamond, 2012). Attempts to train EFs (such as working memory) in isolation have shown limited effectiveness when transferred to improving maths proficiency (Titz & Karbach, 2014). More holistic curriculum based interventions have met with more success, for example Diamond's model of the routes to developing EF emphasises the importance of joy; social belonging and support; and the building of confidence, pride and self-efficacy (Diamond, 2012). The design goals of the Scratch community are strikingly similar (Resnick et al., 2009), with the emphasis on fun, low floor, high ceiling and wide walls (as a route to building confidence and pride in achievements), and a large online community for sharing and support[1]. For these reasons, practice during motivating, authentic and appropriately challenging computational activities could be a rich environment in which to develop the executive functions which will be crucially important to children's lives.

References

Allsop, Y. (2019). Assessing computational thinking process using a multiple evaluation approach. *International Journal of Child-Computer Interaction*, *19*, 30–55. https://doi.org/10.1016/j.ijcci.2018.10.004

Altemeier, L. E., Abbott, R. D., & Berninger, V. W. (2008). Executive functions for reading and writing in typical literacy development and dyslexia. *Journal of Clinical and Experimental Neuropsychology*, *30*(5), 588–606. https://doi.org/10.1080/13803390701562818

Anderson, P. (2002). Assessment and Development of Executive Function (EF) During Childhood. *Child Neuropsychology*, *8*(2), 71–82. https://doi.org/10.1076/chin.8.2.71.8724

[1] Scratch is the most obvious example, but it is not unique. Unplugged approaches and other programming environments often share the same features.

Aron, A. R., Robbins, T. W., & Poldrack, R. A. (2004). Inhibition and the right inferior frontal cortex. *Trends in Cognitive Sciences*, *8*(4), 170–177. https://doi.org/10.1016/j.tics.2004.02.010

Best, J. R., Miller, P. H., & Naglieri, J. A. (2011). Relations between executive function and academic achievement from ages 5 to 17 in a large, representative national sample. *Learning and Individual Differences*, *21*(4), 327–336. https://doi.org/10.1016/j.lindif.2011.01.007

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Annual American Educational Research Association Meeting, Vancouver, BC, Canada*, 1–25. https://doi.org/10.1.1.296.6602

Bryce, D., Whitebread, D., & Szűcs, D. (2015). The relationships among executive functions, metacognitive skills and educational achievement in 5 and 7 year-old children. *Metacognition and Learning*, *10*(2), 181–198. https://doi.org/10.1007/s11409-014-9120-4

Bull, R., & Scerif, G. (2010). Developmental Neuropsychology Callosal Contribution to Procedural Learning in Children. *Developmental Neuropsychology*, *19*(3), 37–41. https://doi.org/10.1207/S15326942DN1903

Cohen, J. (1992). A power primer. *Psychological Bulletin*, *112*(1), 155–159. Retrieved from http://psycnet.apa.org/journals/bul/112/1/155/

Cragg, L., & Gilmore, C. (2014). Skills underlying mathematics: The role of executive function in the development of mathematics proficiency. *Trends in Neuroscience and Education*, *3*(2), 63–68. https://doi.org/10.1016/j.tine.2013.12.001

Daly, M., Delaney, L., Egan, M., & Baumeister, R. F. (2015). Childhood Self-Control and Unemployment Throughout the Life Span: Evidence From Two British Cohort Studies. *Psychological Science*, *26*(6), 709–723. https://doi.org/10.1177/0956797615569001

Denckla, M. B. (1996). Research on executive function in a neurodevelopmental context: Application of clinical measures. *Developmental Neuropsychology*, *12*(1), 5–15. https://doi.org/10.1080/87565649609540637

Diamond, A. (2012). Activities and Programs That Improve Children's Executive Functions. *Current Directions in Psychological Science*, *21*(5), 335–341. https://doi.org/10.1177/0963721412453722

Diamond, A. (2013). Executive functions. *Annual Review of Psychology*, *64*, 135–168. https://doi.org/10.1146/annurev-psych-113011-143750

Dodzik, P. (2017). Behavior Rating Inventory of Executive Function, Second Edition Gerard A. Gioia, Peter K. Isquith, Steven C. Guy, and Lauren Kenworthy. *Journal of Pediatric Neuropsychology*. https://doi.org/10.1007/s40817-017-0044-1

Dolan, C. V, & Molen, M. W. Van Der. (2006). Age-related change in executive function : Developmental trends and a latent variable analysis. *Neuropsychologia*, *44*, 2017–2036. https://doi.org/10.1016/j.neuropsychologia.2006.01.010

Gilmore, C., Attridge, N., Clayton, S., Cragg, L., Johnson, S., Marlow, N., … Inglis, M. (2013). Individual Differences in Inhibitory Control, Not Non-Verbal Number Acuity, Correlate with Mathematics Achievement. *PLoS ONE*, *8*(6), 1–9. https://doi.org/10.1371/journal.pone.0067374

Gilmore, C., & Cragg, L. (2014). Teachers' Understanding of the Role of Executive Functions in Mathematics Learning. *Mind, Brain and Education : The Official Journal of the International Mind, Brain, and Education Society*, *8*(3), 132–136. https://doi.org/10.1111/mbe.12050

Gioia, G., Isquith, P., Guy, S., & Kenworthy, L. (2015). *BRIEF 2 Behaviour Rating Inventory of Executive Function: Professional Manual*. Lutz, FL.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189x12463051

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, *25*(2), 199–237. https://doi.org/10.1080/08993408.2015.1033142

Knouse, L. E., Feldman, G., & Blevins, E. J. (2014). Executive functioning difficulties as predictors of academic performance: Examining the role of grade goals. *Learning and Individual Differences*, *36*, 19–26. https://doi.org/10.1016/j.lindif.2014.07.001

Luckin, R. (2008). The learner centric ecology of resources: A framework for using technology to scaffold learning. *Computers & Education*, *50*(2), 449–462. https://doi.org/10.1016/j.compedu.2007.09.018

Luria, A. (1966). *Higher cortical functions in man*. London: Tavistock.

Lyons, K. E., & Zelazo, P. D. (2011). Monitoring, metacognition, and executive function. Elucidating the role of self-reflection in the development of self-regulation. In *Advances in Child Development and Behavior* (1st ed., Vol. 40). https://doi.org/10.1016/B978-0-12-386491-8.00010-4

Mayer, R. E. (2004). Should There Be a Three-Strikes Rule Against Pure Discovery Learning? *American Psychologist*, *59*(1), 14–19. https://doi.org/10.1037/0003-066X.59.1.14

McLean, J. F., & Hitch, G. J. (1999). Working memory impairments in children with specific arithmetic learning difficulties. *Journal of Experimental Child Psychology*, *74*(3), 240–260. https://doi.org/10.1006/jecp.1999.2516

Miyake, A., Friedman, N. P., Emerson, M. J., Witzki, A. H., Howerter, A., & Wager, T. D. (2000). The Unity and Diversity of Executive Functions and Their Contributions to Complex "Frontal Lobe" Tasks: A Latent Variable Analysis. *Cognitive Psychology*, *41*(1), 49–100. https://doi.org/10.1006/cogp.1999.0734

Moreno-León, J., Harteveld, C., Román-González, M., & Robles, G. (2017). On the Automatic Assessment of Computational Thinking Skills: A Comparison with Human Experts. *Conference on Human Factors in Computing Systems (CHI)*, 2788–2795. https://doi.org/10.1145/3027063.3053216

Moreno-León, J., & Robles, G. (2015). Dr. Scratch. *Proceedings of the Workshop in Primary and Secondary Computing Education on ZZZ - WiPSCE '15*, (November), 132–133. https://doi.org/10.1145/2818314.2818338

Moreno-Leon, J., Robles, G., & Roman-Gonzalez, M. (2016). Comparing computational thinking development assessment scores with software complexity metrics. *IEEE Global Engineering Education Conference, EDUCON*, *10-13-Apri*(April), 1040–1045. https://doi.org/10.1109/EDUCON.2016.7474681

Palumbo, D. B. (1990). Programming Language/Problem-Solving Research: A Review of Relevant Issues. *Review of Educational Research*, *60*(1), 65–89. https://doi.org/10.3102/00346543060001065

Papert, S. (1991). Situating Constructionism. *Constructionism*, 1–11. https://doi.org/10.1111/1467-9752.00269

Parham, J., Gugerty, L., & Stevenson, D. E. (2010). Empirical evidence for the existence and uses of metacognition in computer science problem solving. *SIGCSE'10 - Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, 416–420. https://doi.org/10.1145/1734263.1734406

Pennington, B. F., & Ozonoff, S. (1996). Executive functions and developmental psychopathology. *Journal of Child Psychology and Psychiatry*, *37*(I), 51–87. https://doi.org/10.1111/j.1469-7610.1996.tb01380.x

Resnick, M. (2007). Sowing the Seeds for a More Creative Society. *Learning and Leading With Technology*, *35*(4), 18–22.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., … Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, *52*(11), 60–67. https://doi.org/10.1145/1592761.1592779

Rich, K. M., Andrew Binkowski, T., Strickland, C., & Franklin, D. (2019). A k-8 debugging learning trajectory

derived from research literature. *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 745–751. https://doi.org/10.1145/3287324.3287396

Roebers, C. M. (2017a). Executive function and metacognition: Towards a unifying framework of cognitive self-regulation. *Developmental Review*, *45*, 31–51. https://doi.org/10.1016/j.dr.2017.04.001

Roebers, C. M. (2017b). Executive function and metacognition: Towards a unifying framework of cognitive self-regulation. *Developmental Review*, *45*, 31–51. https://doi.org/10.1016/j.dr.2017.04.001

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, *72*, 678–691. https://doi.org/10.1016/j.chb.2016.08.047

Royal Society. (2017). *After the reboot : computing education in UK schools*.

Scherer, R., Siddiq, F., & Sanches Viveros, B. (2018). Learning to Code—Does It Help Students to Improve Their Thinking Skills? In S. KONG (Ed.), *International Conference on Computational Thinking Education* (pp. 37–40). Retrieved from http://www.eduhk.hk/cte2018/doc/CTE2018 Proceeding_Full_20180604.pdf

St Clair-Thompson, H. L., & Gathercole, S. E. (2006). Executive functions and achievements in school: Shifting, updating, inhibition, and working memory. *Quarterly Journal of Experimental Psychology*, *59*(4), 745–759. https://doi.org/10.1080/17470210500162854

The Royal Society. (2012). Shut down or restart? The way forward for computing in UK schools. In *Technology*. https://doi.org/10.1088/2058-7058/25/07/21

Titz, C., & Karbach, J. (2014). Working memory and executive functions: effects of training on academic achievement. *Psychological Research*. https://doi.org/10.1007/s00426-013-0537-1

Toplak, M. E., West, R. F., & Stanovich, K. E. (2013). Practitioner Review: Do performance-based measures and ratings of executive function assess the same construct? *Journal of Child Psychology and Psychiatry and Allied Disciplines*, *54*(2), 131–143. https://doi.org/10.1111/jcpp.12001

Van der Ven, S. H. G., Kroesbergen, E. H., Boom, J., & Leseman, P. P. M. (2012). The development of executive functions and early mathematics: A dynamic relationship. *British Journal of Educational Psychology*, *82*(1), 100–119. https://doi.org/10.1111/j.2044-8279.2011.02035.x

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, *25*(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5