# An Evaluation of the Effect of Child Programming Courses on the Child's Mathematical Ability

PSYCH 335 Developmental Neuropsychology Written Assignment

Shuangyi Tong (s9tong@uwaterloo.ca)

## 1 Product Overview

This article is focusing on an educational product offered by Hatch Coding[1]. The product provides educational services for developing the child's programming skills. Hatch Coding is a company that aims to offer various programming courses and tools for children in different age groups ranging from 7 to 16 years old. For the courses, the instruction mode is on-campus. From their claims on the website, they have 3 teaching centers in the Great Toronto Area. Besides directly offering courses to children, the company sells the software which they use in classes separately to individuals or organizations for teaching purposes.

Hatch Coding offers three different types of courses targeting children in different age groups. All courses are expected to be completed in 8 to 12 weeks. These three types of courses are: 1. Hatch Junior, 2. Hatch Prime, and 3. Hatch Alpha. Hatch Junior accepts students as young as 7 years old. The course mainly teaches the child how to type fast and accurate on a standard QWERTY keyboard. Therefore, it is not exactly a course in programming. We will discuss later in this article on the possible reasons of making such arrangements. Hatch Prime is advertised for children between 7 to 12 years old. The main computer programming language used in this course is JavaScript which is currently one of the most popular web programming languages (Cass, 2018). This course is instructed with a designated software called "Hatch Studio JS". The goal for this course is to help children start programming, which in their words, becoming programming literate. Hatch Alpha is the most difficult course in the line, and it targets the oldest age group: 9 to 16 years old. This course focuses on mathematical and computational reasoning and uses a popular programming language in machine learning called Python.  Similar to the software in Hatch Prime, Hatch Alpha also comes with a teaching software called "Studio P", presumably "P" stands for "Python".

The software the company developed for their courses, Hatch Studio JS and Studio P, are based on existing widely-used programming languages. Technically, software of this kind is called an integrated development environment (IDE) which essentially provides useful tools and unified user interface that make it easier to write and debug computer programs (Konsynski, Kottemann, Nunamaker Jr., & Stott, 1984). Hatch Studio JS and Studio P are also different from common IDEs because they are

[1] Hatch Studio Coding Home Page: www.hatchcoding.com

educational software and have users in a much younger age group compared to general developers. From the description on the website, Hatch Studio JS provides interactive programming experience, and it designs programming tasks with different levels of difficulty to fit the needs of children with different programming ability. The effectiveness of such software will be evaluated in later sections.

## 2 Product Benefits

While the product website lists many advantages of taking a programming course for a young child, it can be categorized into three major aspects.

First, the product includes a typing practicing component. As the company claimed on its website, the first step of learning programming is typing. For most adults who have been using modern electronic devices for a long time, typing is not a difficult task. However, it may not be true for a young child, which makes it reasonable to have a course in typing.  Moreover, typing is a fine motor control task. We discussed in the lecture that the ability of a child writing out the name correctly has a greater correlation with the future academic performance than an IQ test score. Other research showed motor development information can be used to predict outcomes in later cognitive development (Piek, Dawson, Smith, & Gasson, 2008).

Furthermore, the company emphasizes this product develops real-world skills. They claim the programming languages used in teaching are popular in the industry. With this fact, they further argue early experience with these languages can help a child succeed. This argument is not closely related to a discussion in neuropsychology. We will not go into the details, but the fact that popular computer languages are changing rapidly can be a strong point challenging their argument (Atwell, 2019).

Last, and most importantly to this article, the website advertises these programming courses can develop children's computational logic, computational thinking, and mathematical reasoning skills. These terms can be grossly categorized into mathematical skills. Most people would believe programming is closely related to logic and computation, so these courses should help a child's mathematical skills in some way. There are numerous literatures in developmental psychology studying the development of a child's mathematical skills. Determining how these programming courses can influence the development of mathematical skills is the central claim we want to evaluate in this article.

## 3 A Coarse Review on Mathematical Skills and the Development

Mathematical skills is a broad concept, and it includes various abilities that can be measured with distinct tests. It is hard and not necessary to give reviews on all aspects of mathematical skills in this section, so we will focus on specific sets of functions advertised for this product, namely, numbers, calculations, and mathematical reasoning.

## 3.1 Numbers and Symbols

Numbers, as a set of fundamental symbols in mathematics, is an important indicator of a child's future performance on general mathematics (Jordan & Kaplan, 2009). Jordan & Kaplan (2009) states: "early number competence is important for setting children's learning trajectories in mathematics" (pp. 866). Therefore, it is not

An fMRI meta-analysis article reviewed and identified several possible areas for number forms (Yeo, Wilkey, & Price, 2017). Experiments conducted to identify number forms regions are usually performed by running an MRI scan and asking the participant to solve number comparing questions. Numbers can range from the simplest natural numbers to fractions in different research articles. Yeo et al. (2017) concluded there are three major areas in the brain identified which are related to number forms. Among many fMRI studies, given the assumption that the number forms area does exist, right inferior temporal gyrus (ITG) appears to be responsible for the most reproducible localization. Besides number forms localization in ITG, there are evidence suggesting a network in the frontoparietal region is used in additional processing of semantics, syntax, and lexicon (Starrfelt & Behrmann, 2011). Combining with ITG, Yeo et al. (2017) gave a precise name for this network: symbolic number processing network. One interesting finding to note, numbers appeared as digits (or Arabic numerals) are easier to identify than numbers written in letters or other logographic written systems like Chinese characters (Price & Anasari, 2011). This clearly links to the thinking procedure in programming, because a programming language is not a tool like a calculator where numerical digits are the major inputs. Programming languages heavily use symbols as a variable and a specific number is hidden behind the letters behind the variable. Hence, by the above findings, we may hypothesize using variable is a challenging task for children, which requires understanding not only numbers but also the semantics of a variable.

## 3.2 Arithmetic Computation

The last brain region Yeo et al. (2017) proposed for being responsible for number forms processing is frontal regions. We did not include this finding in the previous section is because the role of this frontal region described by Yeo et al. (2017) is more related to arithmetic computation rather than just a numerical concept. Yeo et al. (2017) state selective visual-spatial attention can be the important function of the frontal region in number form processing. This result coincides with another research in numbers and calculations in children (Arsalidou, Pawliw-Levac, Sadeghi, & Pascual-Leone, 2018). Arsalidou et al. (2018) emphasize the importance of fontal areas in calculation involving attention and working memory. These functions, as we discussed in class, are all in the category called executive functions. It is natural to think when performing computations, working memory is critical in manipulating numbers and other mathematical concepts. This point is firmly supported by research. An experiment done in UK schools show working memory is heavily used by both children and adults for even simple arithmetic (Cragg, Richardson, Hubber, Keeble, & Gilmore, 2017). Moreover, Cragg et al. (2017) suggested other executive functions can compensate weak working memory function in mathematical assessments. Thus, we have reasons to believe performing computation is strongly supported by executive functions in the frontal region of the brain.

## 3.3 Mathematical Reasoning

The literature on mathematical reasoning (or logic) is not as abundant as on numbers and arithmetic. Especially for the child's logic function development, it is difficult to find some solid research articles. Presumably, this is because mathematical reasoning is a higher level of function than other cognitive functions like arithmetic, and children develop complex reasoning skills in relatively later stages. Raven's Progressive Matrices test, as we have seen in the intelligence chapter in the class, can be a measure of abstract reasoning. Interestingly, studies on an experimental teaching program conducted in sixty UK schools show that with an emphasize in teaching children how to use language to reason give rise to children's individual scores on Raven's test (Mercer, Wegerif, & Dawes, 1999). This may provide some validity of learning programming languages can also improve a child's mathematical reasoning.

Neuroimaging experiment results on adults also support our hypothesis that abstract reasoning is a high-level function in terms of evolutionary status (Friedrich & Friederici, 2009). Friedrich & Friederici (2009) proposed the more complicated and abstract a reasoning sequence is, the more anterior the activation is in the inferior frontal gyrus (IFG). Besides IFG being a critical part of abstract reasoning, the posterior of Broca's area is also activated. We learned in class that Broca's area is responsible for language expression, and it is interesting to see all these verbal elements involved in reasoning.

Another worth noting finding on reasoning also comes from adult research. A solid fMRI evidence shows human brain shifts processing information with perceptual bias to logical reasoning is achieved by inhibition of posterior part of the brain to activate a left-prefrontal network where middle-frontal gyrus, Broca's area, the interior insula, and the pre-SMA are all involved (Houde, et al., 2000). As we have studied in this course, inhibition function is one of the major components of executive functions. A famous example demonstrating the inhibition function is A-not-B error (Smith, Thelen, Titzer, & McLin, 1999). Therefore, we have seen executive functions play an essential role in mathematical reasoning ability.

## 3.4 Section Summary

In this section, we selectively reviewed some mathematical skills related to this product. In summary, basic concepts like number forms started from the posterior part of the brain, specifically, inferior temporal gyrus accounts for the localization of number forms. The parietal region is responsible for additional semantic, syntactic, and lexical processing. Lastly, executive functions supported by frontal regions are critical for higher-level mathematical skills, ranging from simple arithmetic to complex mathematical reasoning.

## 4 Evaluate the Product's Effects on Children's Mathematical Skills

With the above discussion on mathematical skills, it is a good point to begin evaluating what are the effects brought by this product that teaches children programming. Because the whole product is offering children a course in addition to the standard class curriculum, it is hard to conclude there are any significant damages this product has to do with children's neuropsychological functions' development. Therefore, we will expand this section by evaluating the possible neuropsychological benefits according to the previous review sections.

### Numbers, Variables, and Syntax

Variables are fundamental in programming and is a higher-level concept than simply a numerical number. As we have noted in number forms section, learning how to fluently manipulate variables (or symbols) can be a challenging task. To successfully use variables, a child must learn to bind the symbol to a quantity and disassociate the symbol with the quantity in a different setting. In mathematical logic studies, a programming language is a form of formal language so that every statement is unambiguous (Hoare, 1969). Then correct syntax is necessary for a program to have minimal functionality. The localization of these functions is in ITG, parietal regions, and Broca's area. With these facts, we may conclude learning to program can be beneficial for a child's development in representations of symbols and correct manipulation of symbols according to a given syntax.

### Arithmetic

As programming rarely involves manual computation, the ability in manipulating numbers may not be directly used. Therefore, it is hard to determine if the child's programming course can have effective improvements in arithmetic performance. Based on the above reasoning, we believe the improvement in arithmetic scores brought by child programming can be negligible.

### Mathematical Reasoning

We have shown mathematical reasoning is a high-level cognitive function mainly supported by frontal and parietal regions of the brain. As we also mentioned previously, programming languages is a form of formal language which requires clear and precise logical thinking. With formal language, one can build an extremely long and correct logical thinking sequence that heavily uses working memory and abstract reasoning. To complete a programming task, one has to attend to the problem and inhibit perceptual bias to activate logical thinking network. Hence, it is reasonable to hypothesis programming can improve a person's logical thinking ability. Although programming languages are very different from natural languages, programming languages share similarities to natural languages. Research show people do respond to visual programming languages cognitively different than verbal programming languages (Blackwell, 1996). So, we have reasons to believe programming in verbal programming languages has a cognitive component overlapped with verbal expression. If we treat programming as a certain form of verbal expression, then research done by Mercer, Wegerif, & Dawes (1999) is the evidence proving teaching children programming does improve children's abstract reasoning.

From the above discussion, we may conclude current evidence does support teaching child programming can be a viable method to improve mathematical performance in aspects like manipulating variables and making abstract reasoning.

## 5 Introducing Cooperative Tasks to Improve the Child's Collaboration Skills

Hatch Coding is not the first product aiming for teaching children programming. Scratch, a visual programming software developed by Lifelong Kindergarten Group at the MIT Media Lab, is a more famous child programming software (Bernnan, 2013). Scratch not only focused on improving the child's computational thinking, but also includes a community collaborative component. With Scratch, the researchers published numerous papers on these two topics. But the way they examine mathematical skills development is more from an education studies perspective that does not involve analysis on the brain structure rather than from the neuropsychological perspective (Brennan & Resnick, 2012). So, this article presents the effect of child programming at a different angle.

Modern software engineering usually involves collaboration with others. Studies show a software developing team has the optimal individual performance with around nine team members (Rodriguez, Sicilia, Garcia, & Harrison, 2012). Scratch also emphasizes their collaborative community helps children collaborate with others. Thus, programming naturally can help children develop communication skills if introducing team works in the class or setting up an online community for children.

However, it seems Hatch Coding does not realize this. On the website, it is advertised as a course that only requires independent work. Therefore, we suggest Hatch Coding should introduce some collaborative components. Because Hatch Coding currently offers on-campus courses only, they can implement more teamwork assignments or programming tasks. Also, Hatch Coding sells their teaching product to other individuals and instructors, then building a shared and collaborative community based on their product can be helpful for the users.

It is important to note, a team assignment or an online community can fail in terms of its effectiveness without proper guidance, especially children are the major population of these tasks. Instructors of the course or company personnel running the online community should be responsible for guiding children to be engaged in collaborative work. The importance of instructors in teaching children is uniform across different subjects (Gillies, 2016).

## 6 Conclusion

This article discussed possible benefits in mathematical performance brought by child programming courses. Also, we proposed a potential improvement to this product by introducing cooperative tasks to develop a child's collaboration skills. From this article, a child programming course is a worth-taking elective course. However, we need to note programming is cognitively demanding, and it can be hard for an average child to master the introductory level. This is similar to saying teaching calculus in grade one can improve a child's mathematical skills, which is not true for every child. Therefore, to be conservative, we concluded this product can be helpful for some children in certain mathematical skills.

## 7 Reference

Arsalidou, M., Pawliw-Levac, M., Sadeghi, M., & Pascual-Leone, J. (2018). Brain areas associated with numbers and calculations in children: Meta-analyses. *Developmental Cognitive Neuroscience, 30*, 239-250.

Atwell, C. (2019, October 3). *10 Years of Programming Language Evolution*. Retrieved from EE Times: https://www.eetimes.com/document.asp?doc_id=1335168

Bernnan, K. A. (2013, Feburary). BEST OF BOTH WORLDS: ISSUES OF STRUCTURE AND AGENCY IN COMPUTATIONAL CREATION, IN AND OUT OF SCHOOL. *(Unpublished doctoral dissertation)*. Cambridge, Massachusetts, United States: Massachusetts Institute of Technology.

Blackwell, A. F. (1996). Metacognitive theories of visual programming: what do we think we are doing? *Proceedings 1996 IEEE Symposium on Visual Languages* (pp. 240-246). Boulder, CO, USA: IEEE.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Annual American Educational Research Association Meeting* (pp. 1-25). Vancouver, BC, Canada: AERA.

Cass, S. (2018, July 31). The 2018 Top Programming Languages. *IEEE Spectrum*, 1-2.

Cragg, L., Richardson, S., Hubber, P. J., Keeble, S., & Gilmore, C. (2017). When is working memory important for arithmetic? The impact of strategy and age. *PLOS One*, 1-18.

Friedrich, R., & Friederici, A. D. (2009, May 28). Mathematical Logic in the Human Brain: Syntax. *PLOS One*, 1-7.

Gillies, R. M. (2016). Cooperative Learning: Review of Research and Practice. *Australian Journal of Teacher Education, 41*(3), 39-54.

Hoare, C. A. (1969, October). An axiomatic basis for computer programming. *Communications of the ACM, 12*(10), 576-580.

Houde, O., Zago, L., Mellet, E., Moutier, S., Pineau, A., Mazoyer, B., & Tzourio-Mazoyer, N. (2000). Shifting from the Perceptual Brain to the Logical Brain: The Neural Impact of Cognitive Inhibition Training. *Journal of Cognitive Neuroscience, 12*(5), 721-728.

Jordan, N. C., & Kaplan, D. (2009). Early Math Matters: Kindergarten Number Competence and Later Mathematics Outcomes. *Developmental Psychology, 45*(3), 850-867.

Konsynski, B. R., Kottemann, J. E., Nunamaker Jr., J. F., & Stott, J. W. (1984). plexsys-84: An Integrated Development Environment for Information Systems. *Journal of Management Information Systems, 1*(3), 64-104.

Mercer, N., Wegerif, R., & Dawes, L. (1999). Children's Talk and the Development of Reasoning in the Classroom. *British Educational Research Journal, 25*(1), 95-111.

Piek, J. P., Dawson, L., Smith, L. M., & Gasson, N. (2008). The role of early fine and gross motor. *Human Movement Science 27*, 668–681.

Price, G. R., & Anasari, D. (2011, August). Symbol processing in the left angular gyrus: Evidence from passive perception of digits. *NeuroImage, 57*(3), 1205-1211.

Rodriguez, D., Sicilia, M. A., Garcia, E., & Harrison, R. (2012, March). Empirical findings on team size and productivity in software development. *The Journal of Systems & Software, 85*(3), 562-570.

Smith, L. B., Thelen, E., Titzer, R., & McLin, D. (1999). Knowing in the context of acting: The task dynamics of the A-not-B error. *Psychological Review, 106*(2), 235-260.

Starrfelt, R., & Behrmann, M. (2011, July). Number reading in pure alexia—A review. *Neuropsychologia, 49*(9), 2283-2298.

Yeo, D. J., Wilkey, E. D., & Price, G. R. (2017). The search for the number form area: A functional neuroimaging meta-analysis. *Neuroscience and Biobehavioral Reviews, 78*, 145-160.