How programming can serve young children as a language for expressing and exploring mathematics in their classes

E. Paul Goldenberg, *pgoldenberg@edc.org*
Education Development Center (EDC), Waltham, MA, USA

Cynthia J. Carter, *ccarter@rashi.org*
The Rashi School, Dedham, MA, USA

Abstract

Ideas can come from any kind of experience, but language is a key part of learning. We commonly rely on two languages for mathematics: the one we speak (and its written form) serves our needs for expressing the context of our problems and the nature of our thinking; and conventional mathematical notation (CMN) lets us record mathematical expressions, equations and formulas. Each is essential. But each also has limitations, especially for young learners.

With support from the National Science Foundation, EDC is researching the use of programming as a third language for children to use in expressing, exploring, and making mathematics. We study children's use/misuse of the first two languages (spoken natural language and CMN) to inform the design of a set of mathematical "maker spaces," microworlds in which students in grades 2–6 use programming in their regular lessons *as a language* for learning mathematics.

Our microworlds' goals are driven by math, not coding, but preserve conceptual coherence and developmental appropriateness in the programming and computer-science/computational-thinking (CS/CT). Topic-centered approaches in mathematics classrooms hide interconnections within mathematics, removing opportunities for surprise and delight and rendering the subject as non-creative. The microworlds we've built and those we're proposing aim to satisfy schools' topical orientation but without siloes, creatively using and foreshadowing other mathematical learning. They are designed and tested to fit easily within regular mathematics classroom study.

In this paper we show how these microworlds are inspired and informed by the way children, especially young children, use linguistic pattern in *spoken language* to build mathematical ideas. We also show how our design aims to circumvent the challenges beginners face with CMN, mathematics' *written language*, which provides the precision and concision natural language lacks and that mathematical exploration requires but which sometimes diminishes clarity for young children rather than enhancing it. We give brief descriptions of two (of four) microworlds that we designed and already used with over 200 second graders, and brief descriptions of how we are extending these to support later-grade learning. These illustrate how programming can be a valuable third language for mathematics, supporting creative exploration, key content, and essential mathematical habits of mind. CS/CT develops in tandem, integrated into the mathematical thinking. In another paper (Goldenberg & Carter, submitted), we describe wholly new microworlds under co-development with 6th graders.

The work described above uses careful observation to understand children's use of language in mathematics and to guide the iterative redesign of our microworlds. That part of our work is an exploratory study, using no formal methodology. We are *also* conducting a more formal study of the effects of the microworlds on children's learning—using cognitive interviews, video, and control classrooms—but that study is still in process and not reported here.

Keywords

mathematics; elementary school; microworlds; programming; mathematical maker space

How programming can serve young children as a language for expressing and exploring mathematics in their classes

At EDC, we are designing *Math + C* (thinkmath.edc.org), a set of mathematical “maker spaces” in which students use programming in their regular lessons in order to express, explore, and make the mathematics they are learning. These microworlds are driven by the math, not by coding. During AY 2018-2019, with support from the National Science Foundation, we developed and piloted four microworlds (and accompanying materials) with >200 7-year-olds in their in-school mathematics classes (Goldenberg, 2019; Goldenberg et al., submitted). We are continuing that research with >200 more 7-year-olds and are designing/testing similar materials for grades 3-5, again four microworlds each. Microworlds are tested both below and above the target grade, because early studies showed that this alternative experience changed the learning trajectories.[1] Learning by making is different from learning on paper.

New experimental microworlds for 6th graders are growing out of a collaboration of students, teachers, and a curriculum designer at EDC.

Mathematical maker spaces for schools? Maker spaces[2] are environments in which people are free to create with the available materials and tools. These are not just rooms with stuff; stuff alone doesn’t foster creativity. Maker spaces have domains: a “typical” maker space (is there such a thing?) might help one explore ideas for robots or for wood or plastic puzzles but not so much for poetry or mathematics. How might one *design* an elementary school mathematics maker space?

The ability to *use* a maker space—to *make* in it—taps one’s prior experiences with tools and materials. We don’t even *think* of using a scissor—let alone a less ubiquitous tool—if we don’t know such things exist. Beyond that, we need *some* experience or help. Maker environments also have a culture. And rules (at the minimum, for safety), as all cultures do. They can be non-coercive, non-didactic precisely *because* of a culture in which *any* person present is seen as a potential source of help, information or ideas.

Our mathematical maker spaces are intended to live in school classrooms, where the cultures (due to constraints both “good” and “bad”) are quite different. Designing *mathematical* maker spaces so that teachers can integrate making into classwork at grade level imposes even more constraints.

Our microworlds start with few tools and are limited to *particular* mathematical constructions, satisfying schools’ typically topical orientation but without siloes, connecting mathematical learning, and preserving opportunities for surprise and creativity. We’ve honed the process for getting 2nd graders familiar with the tools to take only 10 minutes gathered on the rug, mostly doing the demonstrating themselves. Then they can do puzzles we pose or explore freely and invent their own ideas and puzzles. Because the tools are programming blocks, students can also create new tools.

Our programming microworlds are designed to support the mathematical content that teachers and schools deem most critical. We want children to experience material that is new to *them* just as mathematicians approach what’s new to *them*: experiencing surprise, becoming curious, playing, tinkering and researching, puzzling through, and extending rather than just memorizing.

[1] Even *physical* (motoric) milestones vary widely (including 4-month-olds standing erect independently without holding on!) depending on experience. “In contrast to the consistent and orderly progression implied by the [familiar] milestone chart,” the authors report, “the skills infants acquire, the ages they first appear, and their subsequent developmental trajectories are highly responsive to cultural and historical differences in childrearing practices and infants’ everyday experiences” (Rachwani, et al. in press). The order of acquisition of intellectual skills is also not immutable; experience matters.

[2] See, e.g., https://www.makerspaces.com/what-is-a-makerspace/

Our microworlds use programming as a language for mathematics, inspired by the way children, especially young ones, use linguistic pattern in *spoken language* to build mathematical ideas. They also aim to circumvent the challenges beginners face with conventional mathematical notation (CMN), mathematics' *written language*, which provides the precision and concision natural language lacks and that mathematical exploration requires. Microworlds we designed and used with children illustrate how programming as a third language for mathematics supports both creative exploration and essential content.

Spoken language and mathematics

It's fascinating how children use their adept language-brains to build or make sense of early mathematical ideas. Natural language conveys the semantic/contextual settings for mathematical problems. It's also the common way children share and refine their thinking. But, beyond classroom discourse, there's yet another way—one that gets unfortunately little attention in curricula or pedagogy—that children, especially young children, use their language-brains to process mathematical ideas: linguistic pattern.

We asked kindergarteners what they thought "five eighths plus five eighths" might be. One confidently chirped "Ten ayfs!! [pause] What's an ayf?" (Goldenberg, et al. 2017). That's correct, but linguistic, not "mathematical." (More precisely, the boundary between mathematical and linguistic thinking may be blurry at that age.)

We also asked kindergarteners to name the biggest numbers they could think of. Children offered "a hundred" or "infinity" or "a million," but not "two-hundred" or "three-million" (although one said "infinity-googolplex"). When asked what two-hundred plus three-hundred might be, most confidently answered five-hundred. We interpret that as a *linguistic* rather than mathematical strategy because the same children responded to "a hundred plus a hundred" by saying "a hundred," "a million" or "I have no idea." "Hundred" simply meant "big number." To some, a big number plus a big number is a big number. To others, big plus big is bigger. And to the sweet child who turned up her hands, smiled, and said "I have no idea," a big number plus a big number could be anything. These children weren't using the numberness of *hundred*; it was as if we asked what's three goats plus two goats. But "what's a goat plus a goat?" sounds unclear. Is it a trick? What's being asked?

Here's another way children use linguistic strategies. In a game with second graders, they call out half of whatever number we state. We start with familiar facts—the idea is to teach, not stump. If they can halve six or eight they have no trouble with sixty, eighty or eight-thousand; and they feel proud to use such big numbers. For just a few minutes, so the activity doesn't grow old, we mix these no-brainer numbers with others the children know, like halving ten, twenty, forty, four-hundred, twelve-million, repeating some for fun. Then we throw in eight-hundred-and-six (still deliberately chosen to be clear) and many can halve that. We sometimes say, as if revealing a great secret, that "some people don't realize that half of forty-eight is really half of forty and half of eight, but it is." When we then ask a class what half of forty-eight is, roughly half answer twenty-four; most others answer twenty-eight. That non-random, common error identifies the problem: executive functioning, not math. The children are juggling five numbers—forty-eight, forty, eight, twenty, and four—and that's hard for 7-year-olds doing this for a first time. Impressively, if we say only "Great! I heard twenty-four and twenty-eight. One of those is correct!" without saying *which* is correct, children improve at halving 64, 86, 42, and six-million-and-six. Why? And why does half of twenty-two remain a mystery for these same children?

Second-graders love repeating this exercise for a minute or so a couple of times a day for several days. To "compute" half of forty-eight, all they need do is *recite* the numbers (half-of-forty, half-of-eight) that they hear in their head. But half of twenty-two is not ten-one. Nothing about "eleven" *sounds* like half of twenty-two; less than ten minutes into their first try at halving wild numbers, their "mathematical" prowess remains mostly linguistic.

Halving twenty-two requires a calculation—a simple one, but not just a linguistic act. After a few days with numbers that keep the steps feeling intuitive (but with no "instruction"), children are ready for a new idea, *not* purely linguistic. They now understand twenty-two and can now halve fifty-four as twenty-five *plus* two. Halving *seventy*-four requires another step, halving seventy, so it waits for another day. Goldenberg, et al. (2015) report on high-schoolers for whom these child-strategies had been drummed out. One girl for whom halving forty was insultingly easy responded “how should I know?!” when asked to halve forty-eight; prior math experience taught her that what isn’t memorized must be computed on paper. But the logic was easy to regain; she rose quickly to the top of where we want high-schoolers to be, and where too many are not.

Why didn’t the second graders need to be told *which* answer was correct, as long as they knew *one* was? Young children appear to have a built-in cognitive (intuitive) analogue to the distributive property. Children aren’t, of course, factoring out goats (or hundreds or “ayfs”) to know that two goats plus three goats are five goats. Likewise, when asked to double 🗑●●●●●, they draw 🗑●●●●●🗑●●●●● but aren’t thinking “distributive property.” Children don’t start out knowing that half of forty-eight is half-forty and half-eight, but that logic feels so familiar that even when they lose track and fumble the numbers, confirmation that *one* of 24 and 28 is correct lets them intuit *which* one. Seven-year-olds need to build skill at holding five or so numbers mentally and tracking manipulations of them (executive function) but they soon develop that ability and get quite good at halving and doubling, starting with easy numbers.

Mathematics builds new knowledge from old. In elementary school arithmetic, we solve 26×4 not by memorizing *that* fact, but by having *some* starting places (perhaps the standard set of facts, or perhaps 25×4) and applying mathematical logic (e.g., a standard school algorithm, or an ad hoc procedure) to derive the new result. The more mental facts we easily recall, the less work is needed for deriving new results, but acquiring and maintaining mental facts takes work. So, we strike a balance, memorizing *some* easy starting places.

Memorized facts is one way. Linguistic strategies provide another. Laura, a 2nd grader learned to think of “twenty-eight” as a first and last name, like her own, and easily understood what remains if the last name (or first name) is removed. That’s not a mathematical process, but it’s also not a linguistic trick. Numbers aren’t born with names; we name them to make such computations easy. Understanding $28 - 8$ that way, Laura readily computed $28 - 9$, following the linguistic step with a mathematical one. Similarly, after out-loud counting 10 from twenty-three to thirty-three and iterating to get forty-three, fifty-three, sixty-three, a child *hears* the linguistic pattern and anticipates seventy-three without counting. Adding *nine* can then use the linguistic pattern and adjust with an arithmetic step.

These oral/aural linguistic patterns that children are so competent (and inclined) to find powerfully support some concepts and ideas of mathematics that written techniques and manipulative methods don’t make apparent or even obscure.

Written language of mathematics

Reading and writing CMN is not like reading and writing English.

First, reading *any* text is different from nearly every other kind of visual skill. The ability to recognize familiar objects in unfamiliar positions isn’t born in, but infant brains quickly develop such essential geometric transformations, letting babies recognize a bottle even when the nipple doesn’t face them directly. This visual processing takes learning but becomes automatic. For reading and writing, children must suppress that hard-won, now-automatic skill; otherwise p, d, b, and q are all the same. Writing ε and 3 interchangeably is normal, totally expectable at first. This is a reading-writing-only exception, following different rules from any other seeing. Attention to order also changes. We rarely care about order when listing three objects we see but understanding print requires it: *was* and *saw* are different. *Mathematical* print changes the rules again. While 315 and 513 are unequal, but 3×5 and 5×3 are equal. And worse, unlike prose text, mathematical expressions are not read strictly left to right.[3] Convention requires reading/processing both $5 + 4 \times 2$ and $5 \times (4 + 2)$ right to left, a strain to teach and to learn.

Worse yet, unlike prose text, mathematical notations are two-dimensional, requiring attention to both vertical and horizontal position. Students often encounter that first with graphs and charts. Later they see it even in CMN. The typically uneven size and level of children's scrawls are just esthetic concerns; meaning isn't affected. But in mathematics, 315, 31^5 and 31_5 mean different things. Young children don't use those notations but do see $\frac{1}{2}$ early. By middle school all these distinctions matter.

A final example. Spoken (or printed, but spelled out as we do here), "five-eighths plus five-eighths" generally evokes the correct "ten-eighths" response. But written $\frac{5}{8} + \frac{5}{8}$ often evokes the canonical add-everything-in-sight wrong answer, even from bright eighth-graders. Similarly, some mathematical properties (like distributivity) seem essentially built into early cognition. Yet when that idea and its name are "introduced" in third grade (as commonly mandated in the U.S.), it is typically *taught* through a written string like $8 \times 7 = 8 \times (5 + 2) = (8 \times 5) + (8 \times 2) = 40 + 16 = 56$. Far from adding precision or a new idea, this obfuscates what children already know. Processing such a string takes focus and effort, so it isn't the optimal way to *introduce* ideas to an eight-year-old. In fact, despite your own mathematical fluency and adult cognition, probably even you zipped through it without reading closely enough to see if we typed it correctly. For a beginner, the cognitive load of decoding the notation obscures the idea.

CMN, which we now take for granted, was a relatively recent invention; mathematical progress absolutely depended on it. The features that make it so valuable to mathematics are its concision and precision. But, for learners, this comes at a price. Concision removes redundancy. In speech and prose text, some redundancy *helps* understanding. Context helps a beginning reader recognize a word or infer its meaning. And in speech and prose text, minor imprecisions often don't affect comprehension at all, which is on reason it is so difficult to proof-read a document and catch all the errors.[4] In CMN, one missing or misplaced character changes the meaning.

Primary teachers understand that invented spelling, technically "incorrect," is evidence of the emerging (correct!) sense that letters encode specific speech *sounds*, progress from the pre-schooler's letter-salads that show that the child recognizes only that letters tell stories. Conventional spelling comes later. Children learn to read and write text by applying a reservoir of world experience to the thought, content, and meaning of spoken language that the text records. We similarly accept "firemans" without instant correction; the meaning is clear, cosmetic details will come later.

In mathematics, too, thought, content and meaning must come first, conventions later. The difficulty in *acquiring* CMN is that, for young students, world experience and natural language don't sufficiently supplement CMN's concision to support that learning; the difficulty of *using* CMN makes it not ideal for building a basis of thought, content, and meaning for mathematics. When CMN does get taught, teachers must allow the same level of "algebraic babbling" (Cusi, Malara, Navarra, 2011) that is natural and necessary in developing spoken and written natural language.

Here's the problem. On the one hand, in order to express and explore mathematical ideas, even very young children need greater precision than natural language provides. Imaginably, that could be the role of CMN. But learning *any* subject—especially mathematics, which requires focused control of attention—via a language (e.g., CMN) in which one is not fluent raises the cognitive load. Children need another language.

[3] Well, to be precise, neither may be our visual processing of prose text!

[4] Did you catch it?

Programming as a language for mathematics

This idea isn't new but the educational contexts in some of the seminal works (e.g., Papert, 1972, 1980) were informal ones, quite different from school settings, which are increasingly regulated in content. "Coding" is the new sexy study, with new initiatives, standards and Olympiad-like challenges in many countries[5], but many of these still imply independent efforts in school (i.e., robotics or coding classes) or informal out-of-school settings. Our own thinking was particularly influenced by efforts in Bulgaria (Sendova & Sendov 1994; Sendova 2013) and ScratchMaths (Noss & Hoyles, 2018, Benton, et al., 2016, 2017) both of which aimed at functioning within the constraints of expectable school settings. The ScratchMaths project explicitly refers to programming's role *as a language* for maths. Our understanding of children's language development and use, particularly in mathematics, led us to focus on that role.

Expanding on a linguistic strategy: Building a microworld using the idea of "ayfs." One 2nd grade microworld presents a number line with only 0 labeled, not leftmost, a small number of programming blocks—for moves of −5, −3, +3 and +5 and for specifying a starting place START AT 0 (default 0, changeable by child)—and some suggestions for pure explorations or puzzles (Figure 1). Later puzzles offer new blocks, −500, −300, +300 and, +500 and a "zoomed out" number line to pose "mathematically new" puzzles (Figure 2).

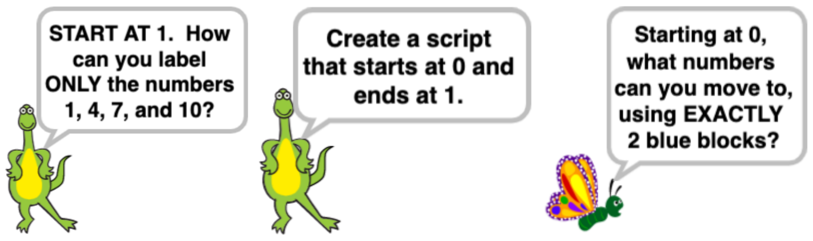

START AT 1. How can you label ONLY the numbers 1, 4, 7, and 10?
Create a script that starts at 0 and ends at 1.
Starting at 0, what numbers can you move to, using EXACTLY 2 blue blocks?

Figure 1: Puzzles with smaller numbers

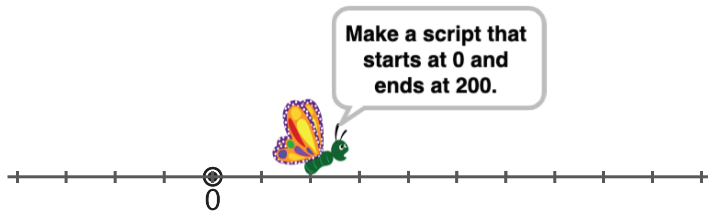


Figure 2: Puzzles that use what the child already knows linguistically but feel mathematically advanced and exciting

Children can create new blocks, if they like—e.g., a +1 block—using their puzzle solutions. Programs' outcomes are not "checked" as a tutorial app might do. Children see the effects they produce and can change them if they like. No puzzle aims to the left of 0 but children often arrive there by accident, and with delight. *Most* children recognized these and felt proud at finding them; *all* of them (*all*!) knew exactly what block to click to get back to "ordinary numbers."[6] Children rarely asked anything more about negative numbers—this *is*, after all, second grade!—

[5] See, for example, https://www.codingforall.org/, https://www.csforall.org/, and http://www.bebraschallenge.org/

[6] On the distinction between "positive numbers" and just "ordinary numbers," see https://blogs.ams.org/matheducation/2018/09/02/ideas-under-construction-children-saying-what-they-know/

but loved them and moved on. They don't need details or follow-up now, but their pleasure with them and experience moving to them and back "out" form a strong basis for future learning. Experience before formality.

Having very few tools and strong contextual support—familiar symbols on blocks and a familiar number line image—meant that introducing this environment to 7-year-olds who had *no* prior programming experience took only 10 minutes, with them doing most of the demonstrating, before they raced to their desks to work the puzzles independently. Learning by doing in context.

With small changes and no number line, 2nd graders express and explore place-value ideas (Figure 3).

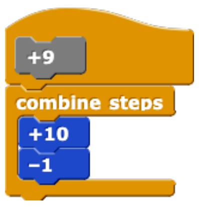
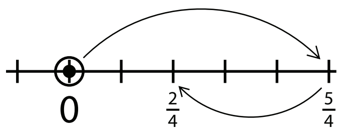


Figure 3: Combining two steps (from ±1, ±10, ±100) to make a +9 block.



Because environments like these "have legs" mathematically, they can grow with children. For example using analogous blocks + 5/4, + 3/4, − 3/4, − 5/4 and puzzles on a zoom-*in* number line serves 3rd grade. That these puzzles feel so familiar as to be "trivial" is the point; fractions are just numbers; $^3/_4 + {}^3/_4$ gives $^6/_4$, not the canonically wrong $^6/_8$.

Algorithmic and functional thought and description. Mathematically comfortable adults might describe $7 \times (5 + 3)$ structurally as function composition, "seven times the quantity five plus three," passing the result of one operation $(+)$ to another $(\times)$. They describe the object not actions. Students likely describe actions "first add 5 and 3, then multiply by 7." The action steps *respect* the composition (add, then multiply), but express it differently. With more elaborate compositions like $\frac{500}{(x+1)^2+14} = 10$, older students' failure to see the composition ($\frac{500}{(something)} = 10$, so the "something" must be 50) leaves them relying mechanically on expansion.

Because CMN admits to both descriptions, both ways of thinking, it doesn't help learners better articulate their current thinking and doesn't help them acquire the thinking they haven't developed.

Functional CMN brings its own mysteries. Students may see $f(x + 1) = x^2 + 2x + 1$ and even $f(x) = x^2$ as if *f* is being multiplied by something. Beginners are also unsure what the variable is when they explore $f(x) = mx + b$ on graphing software (Goldenberg, 1991): after all, what students vary are *m* and *b*, not *x*.

By contrast, the *idea* of function—giving a specific response to a specific input/question—develops *very* early, but that intuitive idea doesn't seem to be readily codified into a way of thinking and describing, perhaps for lack of transparent exemplifying experiences. Certainly, the *notation* doesn't provide that easy experience.

A computer language could. Suitably designed and used, it could help students *recognize* the two ways of thinking. Because it is a *precise* language, it might even help students add precision to their own language as they articulate their thinking.

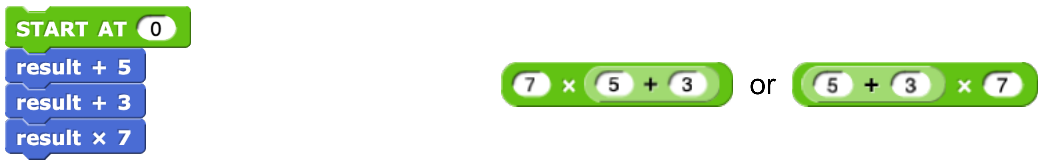


Figure 4. Algorithmic description (left) of the steps involved in computing $7 \times (5 + 3)$ *vs. functional description (right) of the nature of the computation, itself.*

In first and second grade grades, our observations seem to show that the experience of constructing the algorithmic description (figure 4, left) with blocks does help them articulate that process more clearly in words. We don't yet have evidence to show how, or whether, the experience of assembling functional constructs (figure 4, right) affects students' thinking or verbal descriptions; that's part of our current research.

Language in context and live notation. Programming is a "live" language, a notation that can be *run* to give direct, clear feedback on what it says, what effect it has on its "listener" (the computer). This is how children learn natural language and learn through discourse: we say things, others react, and we see what effect we've created. Language in context.

By contrast, a string of symbols that sits on paper (correct or incorrect) gives no feedback without the reader (re)reading and (re)processing it (or relying on outside authority to validate it). Devoid of potentially helpful context, that takes hard work, more than most students are naturally inclined to do.

And part of learning to reason logically involves focusing on the steps. Neither natural language nor CMN express process or algorithm well enough. A good programming language can provide that.

Language for thinking. Programming is also ideally suited to foster not only the mental practices typically associated with CS and sometimes labeled, in aggregate, "computational thinking" (Wing, 2006; DESE, 2016; CSTA, 2019; ISTE, 2019; K12CS, 2019) but also the mathematical habits of mind (Cuoco, et al., 2012; Mark, et al., 2012; Goldenberg, et al., 2012) now codified in the U.S. as the Mathematical Practice Standards (NGA/CCSSO, 2010), including these: (1) Programming provides a language with which students can "construct viable arguments and critique the reasoning of others,"[7] logically, providing a platform for constructing, experimenting with, testing, and thinking about ideas; (2) it eases the process of "beginning with concrete examples and abstracting regularity" or "look[ing] for and express[ing] regularity in repeated reasoning"); (3) used well, it can help students develop the disposition to "make sense of problems and persevere in solving them," and "expecting mathematics to make sense, to feel coherent and not arbitrary, and to have understandable reasons behind facts and methods"; (4) it is a perfect medium for helping students learn to "attend to precision", as computers do what we tell them to do and when that's not what we expect, we can dissect our instructions to find the ambiguity or misstep; and (5), of course, "[using] appropriate tools strategically," the tools being not just the computer and programming but also tools like a number line used *strategically*: *not* just to get answers, but to organize thought. Finally, perhaps best connected with the maker idea, programming can support mathematical creativity—to many people, an oxymoron—posing new puzzles and inventing new ideas.

Conclusion

The challenge is to design microworlds with tasks that are *core* to school at the target age, have mathematical integrity and high cognitive demand, but require little to get started and provide the experience that the harder puzzles depend on. Microworld maker spaces using a programming language that is faithful to mathematics and suitable for learners can do that. Understanding how

[7] All quotations here are from NGA/CCSSO, 2010, pp. 6–8 or Goldenberg, et al., 2015, pp. 6 and 14.

children naturally use linguistic (and other "non-mathematical" cognitive strategies) can guide the work.

Acknowledgment

Funding for doing and reporting the work described in this paper was provided in part by the National Science Foundation, grants 1441075, 1543136 and 1741792. Views expressed here are those of the author and do not necessarily reflect the views of the Foundation.

References

Benton, L., Hoyles, C., Kalas, I., and Noss, R. (2016) Building mathematical knowledge with programming: Insights from the ScratchMaths project. In Sipitakiat, A., and Tutiyaphuengprasert, N. Constructionism in *action 2016. Conference proceedings*. Retrieved from https://drive.google.com/file/d/0BwnqbVelN16LbXY3NHJZaldQY3c/view

Benton, L., Hoyles, C., Kalas, I., and Noss, R. (2017) Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education*, 1–24. https://doi.org/10.1007/s40751-017-0028-x

CSTA. 2019. CS Standards. https://www.csteachers.org/

Cuoco, A., Goldenberg, E.P., and Mark, J. (2012) Organizing a curriculum around mathematical habits of mind. In Hirsch, C., Reys, B. and Lappan, G., eds, *Curriculum Issues in an Era of Common Core State Standards for Mathematics*, Reston: NCTM. (First appeared in *Mathematics Teacher* 103(9) pp. 682-688, May 2010. Reston, VA: NCTM. 2010.)

Cusi, A., Malara, N., and Navarra, G. (2011) Theoretical Issues and Educational Strategies for Encouraging Teachers to Promote a Linguistic and Metacognitive Approach to Early Algebra. In Cai, J. and Knuth, E., eds. *Early algebraization: A global dialogue*. Pgs. 483–510.

DESE. (2016) Massachusetts Department of Elementary and Secondary Education. *2016 Massachusetts Digital Literacy and Computer Science (DLCS) Curriculum Framework*. Accessed at http://www.doe.mass.edu/frameworks/dlcs.pdf, 16 September 2016.

Goldenberg, E.P. (2019) Problem posing and creativity in elementary-school mathematics. *Constructivist Foundations* 14(3), 601–613. http://constructivist.info/14/3/501.goldenberg

Goldenberg, E.P. (1991) The difference between graphing software and *educational* graphing software. In Demana, F., and B. Waits, (eds.), *Proceedings of the Second Annual Conference on Technology in Collegiate Mathematics*. Addison-Wesley, 1991; co-published in Zimmerman, W., and S. Cunningham, (eds.) *Visualization in Mathematics*, Math. Assoc. of America.

Goldenberg, E.P. and Carter, C. (submitted, expected date 2020) Co-designing mathematical microworlds with 6th graders: mathematical maker spaces for grades 2 through 6.

Goldenberg, E.P, Carter, C., Mark, J., Reed, K., and Spencer, D. (submitted, expected 2020) Programming as language and manipulative for second grade mathematics.

Goldenberg, E.P., Mark, J., and Cuoco, A. (2012) An algebraic-habits-of-mind perspective on elementary school. In Hirsch, C., Reys, B. and Lappan, G., eds, *Curriculum Issues in an Era of Common Core State Standards for Mathematics*, Reston: NCTM. (First appeared in *Teaching Children Mathematics* 16(9) pp. 548-556, May 2010. Reston, VA: NCTM. 2010.)

Goldenberg, E.P., Mark, J., Kang, J., Fries, M., Carter, C. and Cordner, T. (2015) *Making Sense of Algebra: Developing Students' Mathematical Habits of Mind.* Heinemann: Portsmouth, NH, USA.

Goldenberg, E.P., Miller, S., Carter, C., and Reed, K. (2017) Mathematical Structure and Error in Kindergarten. *Young Children*, 72(3):38-44. Washington, DC: NAEYC.

ISTE. (2019) Computational Thinking Competencies, https://www.iste.org/standards/computational-thinking

K12CS (K12 Computer Science). (2019) K12 Computer Science Framework. https://k12cs.org/

Mark, J., Cuoco, A., Goldenberg, E.P., and Sword, S. (2012) Developing mathematical habits of mind. In Hirsch, C., Reys, B. and Lappan, G., eds, *Curriculum Issues in an Era of Common Core State Standards for Mathematics*, Reston: NCTM. (First appeared in *Mathematics Teaching in the Middle Schoo*l 15(9) pp. 505-509, May 2010. Reston, VA: NCTM. 2010.)

NGA/CCSSO (2010) Common core state standards for mathematics. National Governors Association Center for Best Practices, Council of Chief State School Officers. Washington, DC.

Noss, R. and Hoyles, C. (2018) The ScratchMaths project is directed by Richard Noss and Celia Hoyles, with Ivan Kalaš, Laura Benton, Alison Clark Wilson, and Piers Saunders. See http://www.ucl.ac.uk/ioe/research/projects/scratchmaths. Accessed March 29, 2018.

Papert, S. (1972) Teaching Children to be Mathematicians Versus Teaching About Mathematics, *Int. J. Math Ed in Science and Tech.*, Vol. 3, No. 3, pp. 249–262.

Papert, S. A. (1980) *Mindstorms: Children, computers, and powerful ideas*. New York City, NY: Basic Books.

Rachwani, J., Hoch, J. E., and Adolph, K. E. (in press). Action in development: Variability, flexibility, and plasticity. In C. S. Tamis-LeMonda and J. J. Lockman (Eds.). *Handbook of infant development.* Cambridge University Press. Preprint formatted by authors. Retrieved from http://www.psych.nyu.edu/adolph/publications/RachwaniHochAdolph-inpress-ActionInDevelopment.pdf, October 28, 2019.

Sendova, E. (2013) Assisting the art of discovery at school age: The Bulgarian experience. In *Talent Development Around the World* (pp. 39–98). Mérida, Yucatán.

Sendova, E., and Sendov, B. (1994) Using computers in school to provide linguistic approaches to mathematics: A Bulgarian example. *Machine-Mediated Learning*, *4*(1), 27–65.

Wing, J. (2006) Computational thinking. *Communications of the ACM, 49*(3), 33–35.