MyCS at 5: Assessing a Middle-years CS Curriculum

Brenda Castro, Terrence Diaz, Marissa Gee, Rebekah Justice, David Kwan,
Preethi Seshadri, and Zachary Dodds
Harvey Mudd College
301 Platt Blvd.
Claremont, CA 91711
{bcastro, tdiaz, mgee, rjustice, dkwan, pseshadri, dodds}@hmc.edu

ABSTRACT
This paper shares the five-year development and deployment of MyCS, a CS curriculum for "middle-years students," roughly in US grades 4-10. Consistent with many middle-years curricula, MyCS promotes positive individual identification with its field, CS, especially as it intersects with other facets of students' maturing identities. A detailed assessment of students' MyCS experiences reveal significant positive outcomes relative to a control group, as well as many neutral (no-distinction) results relative to the control. Feedback from teacher and administrators have refined MyCS and, perhaps more importantly, built curricular bridges to both elementary- and high-school CS. By tracing MyCS's assessment and evolution, this work highlights how two districts used a middle-years CS foothold from which to establish deeper, district-wide changes in identity.

Categories and Subject Descriptors
Social and professional topics - Professional topics - Computing education - K-12 education

Keywords
Middle-school CS; Computational Identity; K-12 CS; MyCS

1. MOTIVATION
Middle-years Computer Science, or *MyCS*, is an 18-week introductory CS curriculum targeting students in US grades 4-10. As described in [1], MyCS synthesizes well-established resources [2,3,4,5] along with original material in a semester-long course.

Successful approaches have integrated CS within existing facets of the middle-years curriculum, e.g. algebra [6] and science [7]. In contrast, MyCS offers a complete course that districts use as a stand-alone elective. Alternatively, its modules can supplement existing tech curricula, nudging students toward confidence in composing, as well as to consuming, computational resources. MyCS's primary goal is to promote a positive computational identity among its students.

Concretely, MyCS succeeds when its students say *"CS is something people like me do."* To date, over 4,000 students have taken MyCS-based courses in the curriculum's first five years; a number larger than that total are taking MyCS in '15-'16. In hopes of supporting other districts seeking to adopt or adapt CS for their middle-years students, this paper shares lessons distilled from MyCS's five-year evolution:

- MyCS's flexibility has enabled it to reach a more diverse student group than identify with CS at older ages. It supports a positive computational identity equitably across the student cohorts it has served.
- MyCS's modularity and graduated online resources support tentative teachers in building confidence in an unfamiliar subject -- at a pace they can tune to their liking.
- Far larger than the differential pre/post impacts between MyCS students and the control groups are the pre/post impacts it has had on its districts. Adopters have developed positive ***institutional*** identities vis-a-vis computing.

We next summarize MyCS's core curriculum and share the results of its assessments. Influenced by partnering districts' growing identification as "*places where **we** do CS*," MyCS's curriculum has recently piloted transition materials that reach outward from its target age-range. From those efforts, we reflect on the conditions through which the "middle-years" might offer an especially fertile starting ground from which to grow a positive district-wide CS identity and CS program.

2. CURRICULUM
MyCS's curriculum emphasizes CS's breadth, opportunities for creative expression, and experiences in computational problem solving. For the past five years, MyCS has refined a computing course featuring conceptual, hands-on, and on-screen activities.

Often, the eighteen-week curriculum is expanded to a full year through deliberate pacing or through combining its material with an existing tech class. Professional development workshops have built enthusiasm and confidence for teaching CS in partnering districts, PUSD and KUSD, of Pomona, CA and Kauai, HI.

Detailed more thoroughly elsewhere [1], we offer here a brief summary of each of MyCS's six three-week modules. The first, "*What is CS?*," builds students' intuition about CS by making more deliberate and explicit their substantial prior experiences -- and assumptions -- about computers and software. The second module, "*A-maze-ing Scratch*," introduces students to Scratch through a gently graduated set of 100 mazes. "*Codes and Data*" challenges students to encode and decode messages of many forms, culminating in using binary to represent – and rebuild – a Lego tower of their own design. The fourth unit, "*Interactive Scratch,*" returns to programming by guiding message-sharing, story-staging, and game-creation. The fifth unit, "*Problem Solving and Algorithms*," offers a screen-free, hands-on introduction to algorithmic thinking, including searching and sorting. The final unit again spirals back: "*Algorithms in Scratch*" scaffolds students through decomposing problems and implementing solutions in what, we hope by term's end, is a programming environment in which they feel comfortable and confident.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
SIGCSE '16, March 2–5, 2016, Memphis, TN, USA.
© 2016 ACM ISBN 978-1-4503-3685-7/16/03…$15.00 .
DOI: http://dx.doi.org/10.1145/2839509.2844643

Figure 1. Prompts and averages for *all* survey-takers (using a 1-6 Likert scale, with N=5475)

Date	Computing is fun.	I am good at computing.	Programming is Hard (Reversed)	I can become good at computing.	I'm interested in a career in computing.	I like the challenge of computing.	Computer jobs are boring. (Reversed)	I'm interested in more computing classes.	I'm more interested in computing now than I was last year.
Sep.	4.83	4.13	3.26	4.71	3.37	3.80	4.19	3.85	4.10
Dec.	4.80	4.08	3.40	4.66	3.41	3.83	4.21	3.90	4.20
May	4.81	4.23	3.37	4.61	3.45	3.88	4.17	3.90	4.26

3. ASSESSMENTS

Partnering districts' pilot and formal deployments of the course, both in part and in whole, have supported an assessment of MyCS student experience – especially as it overlaps with computational identity-building. In 2014-15, over 6000 student surveys were collected. Additionally, in the summer of 2015, MyCS ran three professional development workshops, reaching a total of 52 teachers, also pre/post surveyed.

3.1 Surveys and methods

Three times in 2014-15, in Sep., Dec., and May, partnering teachers asked their students to complete a nine-question survey elaborating computational identity. Figure 1 shows both the prompts and each season's average Likert responses on a scale from 1 to 6. Seven of the statements were phrased positively. We reversed the scores of the two negatively-phrased prompts, "*Programming is hard*" and "*Computer jobs are boring*" to provide a uniform measure of *positivity* in students' computational identity. For all of the analysis, higher scores represent a more positive view of CS. We note that we did *not* seek to measure students' pre/post changes in computational knowledge or skillsets.

The 6115 student responses were then anonymized, duplicates removed, and other invalid submissions deleted.[1] If a student submitted multiple responses in a single setting, we kept only the last. The cleaned data set consisted of 5475 responses.

During the week-long professional development workshops, pre/post surveys also tracked teachers' confidence with computing concepts and skills. Teachers reflected daily on the MyCS curriculum and its relevance/applicability to their classrooms.

3.2 Overall Results

As seen in Figure 1, responses were more often positive than negative, though not dramatically so. In general, even though students agreed that programming is difficult, they expressed enthusiasm about CS (being fun) and confidence in their own skills – and, more importantly, with even greater confidence in their *future* skills! While their attitudes did not change significantly over time, responses remained favorable throughout the three time periods.

We also separated all 5475 responses by self-identified gender; Figure 2 summarizes the results across time for each prompt. With MyCS's goal of inviting *all* identity-groups to take ownership of computing, the results are heartening. Young women consistently averaged significantly more positive responses to the computational identity questions, with $p<1.0e-6$. We note that, though the trend is positive, the results did not show significant changes over time between the young women's and men's cohorts.

It is crucial to note that the results of Figure 1 and Figure 2 *do not separate students who took MyCS from a control group*. Rather, we include those results here as a baseline snapshot – including some stereotype-defying distinctions – of the views of two quite diverse districts' middle-years students. Next we turn to how equitably MyCS reached different demographics within those districts, along with the differences (or lack thereof) between the students taking MyCS and the control cohort.

3.3 Comparative Results

To obtain the 5475 responses, our partnering teachers asked *all* of their students to share their reactions to the computational-identity prompts. This approach provided a large control group (whose classes included no computational components at all), as well as a cohort actively engaged in MyCS and/or some of its modules. In addition, it provides a natural control for variance between individual teachers: each teacher had students within the MyCS group and outside of it.

Among the students surveyed, 613 took a class that included the entire MyCS curriculum or several full units from it. We designated this group as the MyCS cohort. The control cohort comprised 3360 survey-takers who were in classes with little or no computational content. The other 1502 students were in classes with many full lessons from the MyCS curriculum, woven through existing curricula: these students were not included in the comparative results summarized in Figure 3. As the side-by-side distributions suggest, the MyCS cohort did report a significantly ($p<1.0e-10$) more positive computational identity than those in the control group.

However, the results showed *no* significant differences in the **gains** by the MyCS cohort relative to the control group. Reinforcing the data of Figures 1 and 2, ***all*** students, whether taking MyCS or not, increased their personal positive affiliations with computing over the past year – to about the same extent.[2] In fact, the one notable dip in positive sentiment among the MyCSers appeared for the prompt *"Programming is hard,"* i.e., students who spent considerable time programming did develop an appreciation for its challenges. Here, however, it seems that MyCS has helped those students keep programming's challenge in perspective: other facets of a positive computational identity remain.

[1] If either Taylor Swift or Iggy Azalea *did* take MyCS with us last year, we're afraid our analysis does not include their responses.

[2] Coincidentally, tech market indices followed this same trend.

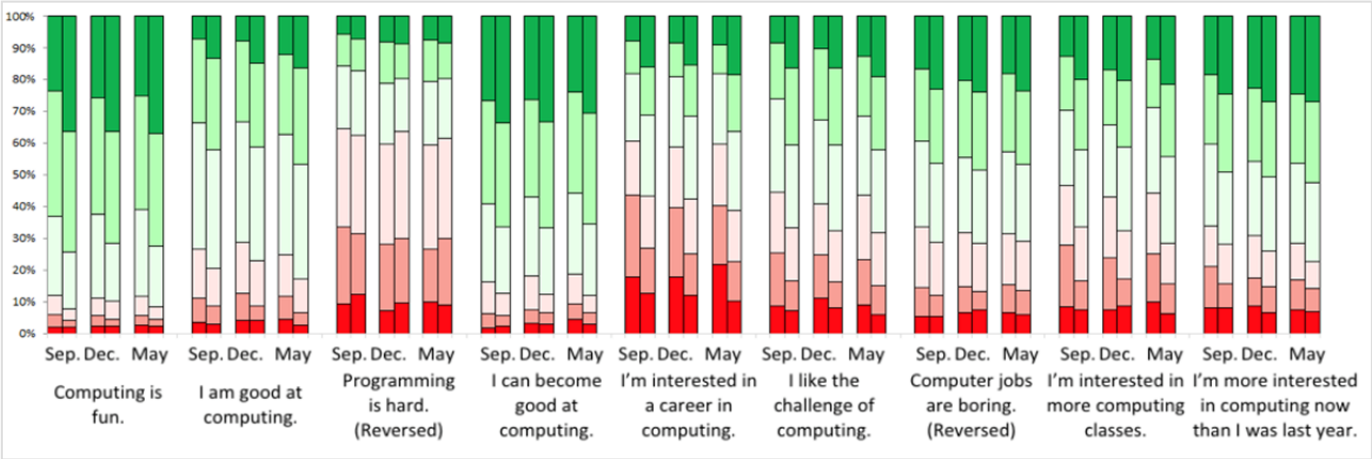

Figure 2. Computational identity responses over time separated by gender. Each column shows the distribution of Likert responses to each prompt, with the most positive scores (6) at top in dark green and the most negative (1) at bottom in dark red. Within each contiguous pair of columns, young women's answers are at right and young men's are at left. Averaging across prompts, female students expressed a significantly more positive computational identity ($p < 1.0e\text{-}6$)

One interpretation of these data is that overall positive sentiment toward CS was growing, perhaps particularly among adolescents. MyCS, then, offered an opportunity for especially enthusiastic students to pursue their computing interests, but did not lead to greater changes than the overall trend.

MyCS's long-term goal, however, is to reach students with less favorable views of computing and invite them to reconsider the field in a more personally positive way. More work remains.

3.4 Accessibility Results

Further, the possibility that MyCS is providing a home for computing-interested students is a cause for concern -- ***if*** it is perpetuating or reinforcing too-common stereotypes that only *some* groups of people identify with CS. Although MyCS's partnering districts serve a large number of students from backgrounds poorly represented in post-secondary CS, past research has demonstrated that simple co-location does *not* suffice to broaden CS's pipeline [8,9].

To assess how MyCS relates to the broadening *accessibility* to computing, we compare the demographics of students in the MyCS cohort with those in the control in Tables 1 and 2.

Table 1. Raw numbers and percentage-wise measures of the number of male and female students in MyCS and non-MyCS courses. MyCS's balance did not match the overall group's**.**

Gender	Control: Raw	Control: Percentage	MyCS: Raw	MyCS: Percentage
Female	1603	47.7%	263	42.9%
Male	1757	52.3%	350	57.1%

Table 1 shows that the students taking MyCS are to a larger extent male than female, even relative to the control cohort. This difference is likely due to individual student choices. At some schools MyCS is an option competing with other possible electives; at other schools, it is part of a technical curriculum that all students take. It is worth noting that, on average, the young women surveyed did not have less personal affiliation for computing, as Figure 2 attests. In addition, the 43/57 split, while not even, is noticeably more balanced than in some of the district's other electives, as reported by the teachers in MyCS's professional development workshops.

Table 2 shows that MyCS reached a larger proportion of students from racial and ethnic groups underrepresented (UR) in post-secondary CS. This testifies, in part, to the energy the districts and the teachers have invested in creating an inviting pathway into not only MyCS but all technical fields.

Table 2. Raw numbers and percentage-wise measures of the racial/ethnic background of students in MyCS and non-MyCS courses.

Race/ Ethnicity	Control: Raw	Control: Percentage	MyCS: Raw	MyCS: Percentage
Asian	310	9.2%	32	5.2%
Black/African American	67	2.0%	26	4.2%
Latina/o	1689	50.3%	386	63.0%
Native American	46	1.4%	9	1.5%
White	359	10.7%	59	9.6%
Multiracial	461	13.7%	22	3.6%
Other	429	12.8%	79	12.9%

We do note, albeit without a good explanation, that students identifying as *multiracial* did not participate in MyCS proportional to their overall numbers. Even so, the rate of participation across almost all of these demographic identities does suggest that MyCS can help provide an *intra*curricular opportunity for computing participation that avoids some of the segregation seen by other computing curricula and by some extracurricular or business-based delivery models [8,9].

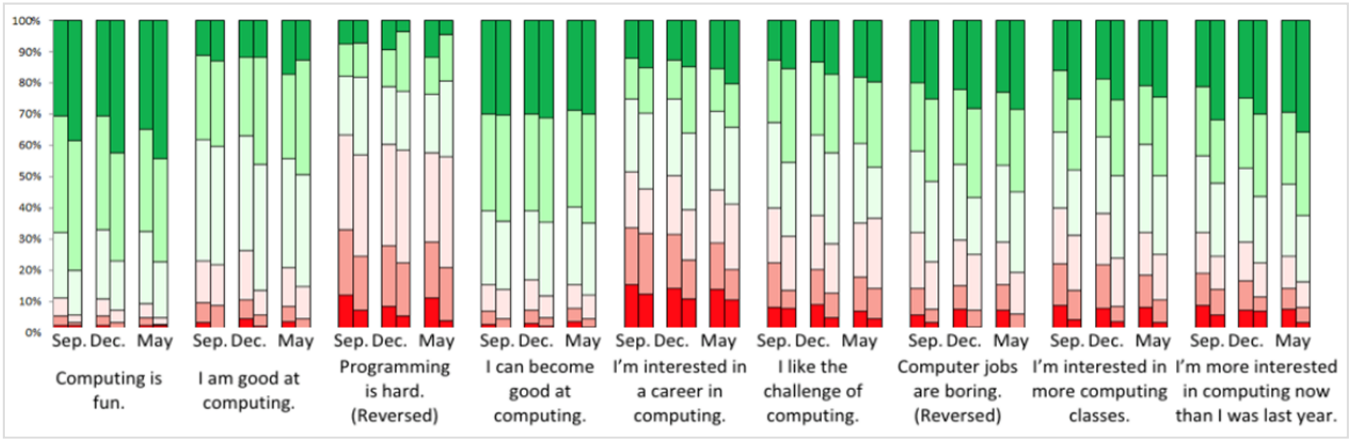

Figure 3. Changes in computational identity responses over time separated by whether the student was in a MyCS course (right-hand columns) or not (left-hand columns). Students in MyCS courses were significantly more positive than their peers about the field, but the gains (increased positivity) for MyCS students were *not* significantly greater than the gains for the control group.

3.5 Assessing the MyCS impact on teachers

One facet of the MyCS curriculum that, we believe, helps support a diverse group of students is its flexibility. Short term, most middle-years computing teachers will have a computing background almost wholly drawn from their professional-development (PD) experiences and their classroom experiments with teaching computing. Because of this, the primary goal of MyCS's PD workshops is teachers' confidence and comfort with *easing* into teaching computing.

Surveys of the 52 summer '15 PD workshop, summarized in Figure 4's distributions and Figure 5's numeric summaries, show that the group of teachers gained significantly in confidence with computing ($p<0.01$) for each skill queried.

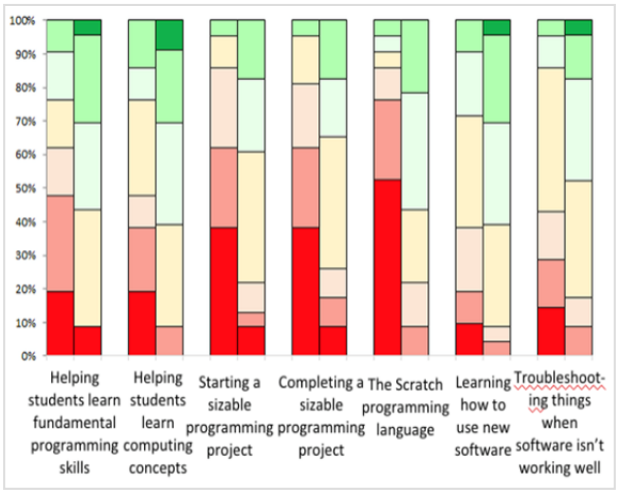

Figure 4. Teacher responses to surveys given before (left-hand columns) and after (right-hand columns) 2015's MyCS PD workshops. For all eight skills, teachers' confidence significantly increased over the course of the week.

In addition, teachers often depart the workshop reporting a desire to build their skills and understanding further. As one teacher commented on the final day, "This has opened a new world for us to expose to our students. We feel that preliminarily we can teach and integrate some of the concepts. We would love another year or two for more information and practice with computer science." Having a few teachers who are this enthusiastic can have an impact on entire schools and the numerous students they serve.

After a workshop, teachers are more enthusiastic about CS, and they express intent to share what they have learned with their school. One teacher shared, "MyCS and [this program] have had a huge positive impact on Fremont Academy. MyCS has help[ed] our Engineering teachers see the relevance of CS and it has shown them how to implement CS in the classroom. Just amazing!!! Thank you."

Although we realize that it's possible only in light of CS's still-maturing standards for external reporting at the middle-years level, we do strongly encourage teachers to adapt MyCS's resources to their specific situation and subjects. As a result, teachers have interlaced MyCS through all types of curricula, whether the class focuses on technology or not. Looking over the full five-year set of PD participants (N=145), of the teachers who attended the workshops and completed the surveys, a majority, 58%, have integrated some amount of MyCS in their classrooms – often in inspiring ways. For instance, one woodshop teacher has added MyCS's Arduino unit, part of the curriculum's bridge modules, asking his students to add programmatic motion and sensing to their projects. Other teachers dedicate a day a week from an existing tech-applications class, *CS Fridays*. Some science and math teachers integrate the screen-free lessons involving encoding and decoding into their science or math classes. English teachers have used the Scratch storytelling module to guide pairs of students in creating a scene summarizing a chapter from the current novel, resulting in a series of scenes following the book from beginning to end. Still other teachers use MyCS activities as warm-ups to start class or as rewards for students who finish work early!

3.6 Integrating qualitative teacher feedback

Through the workshops' daily routine of qualitative feedback, this diversity of teachers' voices and experiences has helped refine and expand MyCS's ability to reach a wide audience. One of the most difficult facets of MyCS for some teachers is developing confidence in using Scratch. Several teachers in 2015's early-summer PD noted this discomfort, e.g., "The Scratch game-building may be too difficult, and I would need training to teach this. It would be easier to start off with someone else's code," echoed by many other comments, as well.

Averages by questions	Helping students learn fundamental programming skills	Helping students learn computing concepts	Starting a sizable programming project	Completing a sizable programming project	The Scratch programming language	Learning how to use new software	Troubleshooting things when software isn't working well
Before	3.05	3.33	2.24	2.29	2.00	3.71	3.33
After	4.65	4.83	4.13	4.00	4.48	4.83	4.43
Difference	1.60	1.49	1.89	1.71	2.48	1.11	1.10

Figure 5. Average scores on a Likert scale from 1-6 of MyCS PD workshop participants' self-reported confidence in each of seven skills. Here, N=52 and the third row's differences are significant at the $p<0.01$ level for each skill.

To be sure, the "*Interactive Scratch*" module is MyCS's programming foundation, seeking to promote self-directed design, logical thinking, and collaborative expression. We updated the online curriculum using Scratch's new step-by-step and video tutorials, supplemented by the original starter projects. The curriculum now also leverages Colleen Lewis's Offline Scratch [offline] materials to support sound, art, and storytelling lessons, building gradually from others' code to creating one's own. The late-summer PD benefitted from these changes, e.g., with comments like, "Scratch: tutorials are easy to use. There's lots of potential there," and a large group of teachers leaving excited about using Scratch with their students.

MyCS's screen-free algorithms module (unit 5), in part, introduces searching and sorting. In early PD experiences the sorting activity moved too quickly, and many teachers reported feeling they did not grasp the explanations or examples, or purpose. Moreover, in an attempt to show the power of sorting algorithms, we increased the difficulty too quickly and left teachers feeling frustrated. The subsequent workshop recalibrated this experience, showing only two sorting algorithms, gradually building from basic examples to illustrate their processes and differences. After the workshops, the late-summer cohort reported a stronger grasp and greater confidence in this unit's material, in contrast to the early-summer group, who reported an impression of why sorting and searching were important but little confidence in their details.

4. BRIDGING BEYOND MyCS

One of the unexpected results of deploying a middle-years CS curriculum through two districts has been the change in the computational identities *of the districts themselves*. In fact, as Figures 2 and 3 attest, even absent significant intra-year gains in students' computational identities, inter-year gains in *institutional* computational identity have been substantial:

- After 2-3 years, both partnering districts sought to deploy MyCS at the elementary-school level; in 2015 Pomona opted to send a full PD cohort of 20 elementary teachers in order to start a district-wide rollout of computing.

- One district, too, found that the middle-years starting point was a gentle, low-overhead means to catalyze CS at the high-school level, resulting in a demand for a 2014 and 2015 "bridge" PD workshop, in which MyCS instructors added Python and Arduino materials to their repertoire. Those materials, in turn, are seeding an organic process of secondary-level CS development.

In this section, we share some of the "curricular bridges" that have grown out of the MyCS effort to bring in-school computing to both elementary and secondary classrooms of our partnering districts.

4.1 Textual programming: Python

As a result, 2014 saw the introduction of a week-long Python PD workshop run in parallel with PD for new MyCS instructors. Each unit of the Python curriculum begins with an interactive discussion, followed by hands-on practice with programming. Unit 1 of the Python curriculum provides an overview of data types and conditionals through creating multiple versions of rock-paper-scissors. An optional text adventure utilizes concepts such as user-input, strings, conditionals, and the `random` module. Unit 2 uses the Turtle library and loops, with activities ranging from composing functions to generate basic shapes to constructing more complex drawings such as checkerboards, targets, faces, wreaths, spirals, and pixelated grids.

Unit 3 parallels MyCS's encoding and decoding challenges with string and list operations to produce a "Pig Latin Generator," "Acrostic Decoder," "Letter Shifter Function," and "Prime Number Challenge." Again, similar to the latter units of MyCS, Python's Unit 4 consolidates the skills from previous units to discuss problem-solving algorithms in order to construct games such as a maze and tic-tac-toe. Hands-on challenges for the various units range from SET as a visual matching game that incorporates conditionals, to the human-pen drawing activity serving as a live preview of Turtle graphics to demonstrate how a computer will execute the code.

Python feedback Python piqued teacher engagement during our workshop by fostering anticipation and maintaining interest. As one teacher noted, "The Tic Tac Toe lesson would be very relatable to my students. This would be a fun and engaging lesson". We also incorporated numerous paper and discussion-based challenges to underscore the logical, step-by-step procedure behind computational processes. As with MyCS, teachers reacted positively to that balance, e.g., "I like the 'screen-free' stuff. The hands-on activities give the students a more tangible way of accessing the material."

The greatest challenge we faced was our daily time limit; even carefully structured, it is impossible for teachers to completely hone all the skills. Perhaps this is an equally important realization! One teacher remarked, "Everything was interesting. We just need more time." Another teacher admitted, the "most difficult [area] is the SYNTAX, SYNTAX, SYNTAX! The next most difficult thing is the SYNTAX still."

4.2 Hands-on Hardware: Arduino

Drawing from backgrounds across a variety of shop and professional-skills curricula, many teachers sought a MyCS extension that overlapped hands-on technical projects with project-based pedagogy and scheduling. The Arduino is a natural fit: it provides a pathway for creativity and hardware expression and, at heart, is a computational interface. Also a plus was the

Arduino's and its components' low cost, e.g., relative to Vex or Lego kits. All of the accessories in Figure 6 are part of the sequence of challenges provided. With servo motors, students can experiment with different types of wheels and determine which ones work better for their goals and chassis, i.e., a faster vehicle versus a sturdier vehicle.

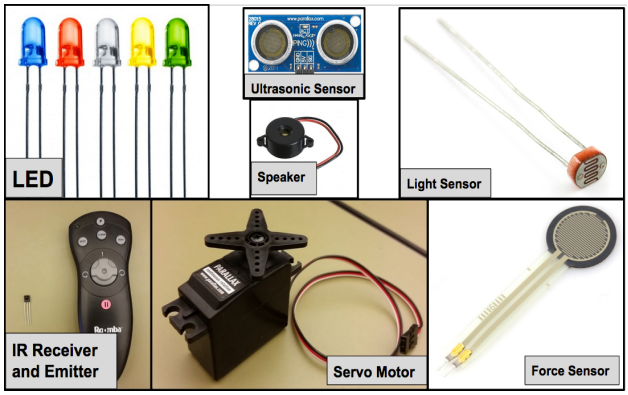

Figure 6. Hardware accessories compatible with the Arduino and used in the MyCS Arduino bridge-curriculum tutorials. Returning PD teachers can opt to supplement the content with Arduino or Python work. This, in turn, has led to deepening *institutions'* computational identities.

Accessibility via Ardublock To leverage the skills used in MyCS's Scratch-based curriculum, MyCS's Arduino curriculum uses Ardublock, a Scratch-like language in which syntax errors are not possible. Ardublock offers an important feature that Scratch does not, however: each time a block of code is compiled in Ardublock, it is converted into a text file (in Arduino's native Processing language) and then run. Far from a burden, this intermediate step exposes students to the underlying Processing code – and gives them the opportunity to re-edit and re-run that text file as many times as a student might wish. This provides an inviting – and unintimidating transition from block-based to text-based programming.

We have observed in workshops that attendees start with the block-based language and, at whatever pace they feel comfortable, start to modify the textual language rather than the block code. Usually, that experimentation leads to questions and exploration about how to use the text-based interface more effectively, as many students (and their teachers) find they are so much more efficient to specify and modify programs using text than blocks!

Arduino Feedback The MyCS Arduino tutorials provide scaffolded challenges that introduce hardware accessories in a natural order. First, LEDs support exploring programmed light patterns, then the photosensor enables creation of a light-meter or passing messages from one pair's Arduino circuit to another with both LEDs and photosensors. Servomotors mounted underneath the Arduino create a tethered robot, and an IR sensor/emitter allows the robot to be untethered but still remote controlled with an ordinary TV remote. The force sensor, speaker, and ultrasonic distance sensor add capabilities that allow student teams to build creative projects including theremins, piano-like instruments, or an alarm that someone's approaching or a door has moved.

During the workshops, teachers responded well to this progression, saying "My favorite [part] was the first day and learning about the very basics of the Arduino." An Arduino exhibition is a natural capstone to the unit; as one teacher noted, "the general idea to create a project is [an] excellent strategy."

5. PERSPECTIVE

In its adopting districts, the MyCS curriculum has evidenced both strengths and weaknesses relative to its original goals:

- By offering MyCS, partering districts have provided an opportunity to build computing skills and knowledge *in-school* to many more middle-years students than before it.
- Survey feedback to date does **not** show MyCS bolstering students' positive computational identities to a greater extent than a control cohort, but it **does** show that MyCS offers a home for engaging those students – from a representative cross-section of the surrounding district – who identify significantly more positively with computing.
- MyCS teachers and PD attendees are significantly more confident with computing skills after the experience, and are more likely than not to include computing in their classes.

Perhaps most importantly, MyCS has led to a deeper *institutional* identification with computing across its partner districts. The middle years already carry an expectation of transition, identity exploration, and, frankly, feeling awkward. As a result, those years are a safe place from which a district can experiment with the undeniable awkwardness of computing – and from which it can grow in the directions that best suit that district's teachers' and administrators' ambitions for their schools. In 2015-16 half of San Fransisco's middle schools will pilot MyCS, and we look forward to that chance to more deeply understand how to foster a positive computing identity among all middle-years students.

6. ACKNOWLEDGEMENTS

We thank the teachers, students, and administrators who have helped develop and refine MyCS. We thank Google's CS4HS initiative for supporting the pilot years of MyCS, and the NSF, though project #1240939, for enabling a careful assessment of the curriculum and substantially expanding its reach.

7. REFERENCES

[1] Schofield, E., Erlinger, M., and Dodds, Z., MyCS: CS for Middle-years Students and Their Teachers SIGCSE '14, Atlanta, GA, USA, pp. 337-342. www.cs.hmc.edu/MyCS

[2] Lewis, C. 2013. Scratch Offline. colleenmlewis.com/scratch

[3] Goode, J., Chapman, G., Margolis, J. (2012). Beyond Curriculum: The Exploring Computer Science Program. ACM Inroads. 3(2), 47-53.

[4] Bell, T., Witten, I., and Fellows, M. 2010. Computer Science Unplugged. URL: www.csunplugged.org

[5] code.org, accessed 10/23/2015

[6] Schanzer, E., Fisler, K., Krishnamurthi, S., and Felleisen, M. Transferring Skills at Solving Word Problems from Computing to Algebra Through Bootstrap SIGCSE '15, Kansas City, MO, USA, pp. 616-621.

[7] Lee, I., et al. Project GUTS URL: www.projectguts.org

[8] Sankar, P., Gilmartin, J., and Sobel, M. 2015. An Examination of Belongingness and Confidence among Female Computer Science Students. SIGCAS, 45 (2), Jul 2015, pp. 7-10.

[9] Margolis, J. Stuck in the Shallow End MIT Press, 2008.