Co-Teaching Computer Science Across Borders: Human-Centric Learning at Scale

Chris Piech
Stanford University
Stanford, CA, USA
piech@cs.stanford.edu

Lisa Yan
Stanford University
Stanford, CA, USA
yanlisa@stanford.edu

Lisa Einstein
Stanford University
Stanford, CA, USA
lisae@stanford.edu

Ana Saavedra
Stanford University
Stanford, CA, USA
anamar@stanford.edu

Baris Bozkurt
Izmir Demokrasi Universitesi
Izmir, Turkey
baris.bozkurt@idu.edu.tr

Eliska Sestakova
Czech Technical University
Prague, Czech Republic
eliska.sestakova@fit.cvut.cz

Ondrej Guth
Czech Technical University
Prague, Czech Republic
ondrej.guth@fit.cvut.cz

Nick McKeown
Stanford University
Stanford, CA, USA
nickm@stanford.edu

ABSTRACT

Programming is fast becoming a required skill set for students in every country. We present CS Bridge, a model for *cross-border co-teaching* of CS1, along with a corresponding open-source course-in-a-box curriculum made for easy localization. In the CS Bridge model, instructors and student-teachers from different countries come together to teach a short, stand-alone CS1 course to hundreds of local high school students. The corresponding open-source curriculum has been specifically designed to be easily adapted to a wide variety of local teaching practices, languages, and cultures.

Over the past six years, the curriculum has been used to teach CS1 material to over 1,000 high school students in Colombia, the Czech Republic, Turkey, and Guinea. A large majority of our students continue on to study CS or CS-related fields in university. More importantly, many of our undergraduate student-teachers stay involved with teaching beyond the program. Joint teaching creates a positive, high-quality learning experience for students around the world and a powerful, high-impact professional development experience for the teaching team—instructors and student-teachers alike.

Author Keywords

Course-in-a-box; co-teaching; CS for All; international education

L@S '20, August 12–14, 2020, Virtual Event, USA.

ACM ISBN 978-1-4503-7951-9/20/08 ...$15.00.
http://dx.doi.org/10.1145/3386527.3405915

CCS Concepts

•Social and professional topics → Computing education; CS1;

INTRODUCTION

Computer science education has made substantial progress towards the goal of CS for All in the United States, online, and in some regions of the world. However, there is mounting evidence of a growing global digital divide, where access to CS education is heavily dependant on which region you were born in [5]. Despite progress, out of the more than 1.3 billion K-12 students in the world [40], a small fraction will have the opportunity to learn to code. Though online learning gives access to CS, the post-mortem on massive online open-access courses (MOOCs) suggests that human teaching remains an essential element for a successful education. The CS education community is thus faced with a challenge: How can we scale high-quality *in-person* and culturally relevant education more evenly across the globe—for students of all genders, diverse socio-economic backgrounds, and different countries of origin?

Scaling quality *in-person* CS education globally has two major barriers: First, good learning ideas have trouble spreading across borders; one teacher's great classroom idea will experience substantial friction when adapted to other classrooms [12]. These challenges are exacerbated across teaching contexts in different countries. As a result, cultural, pedagogic, and organizational know-how may be siloed within countries and institutions. Second, many students live in communities without trained CS teachers, resulting in a "chicken and egg" problem—the lack of teachers with the required content knowledge prevents students from becoming future teachers [3, 37]. We note that both of these challenges exist in all countries—including in the United States [41].

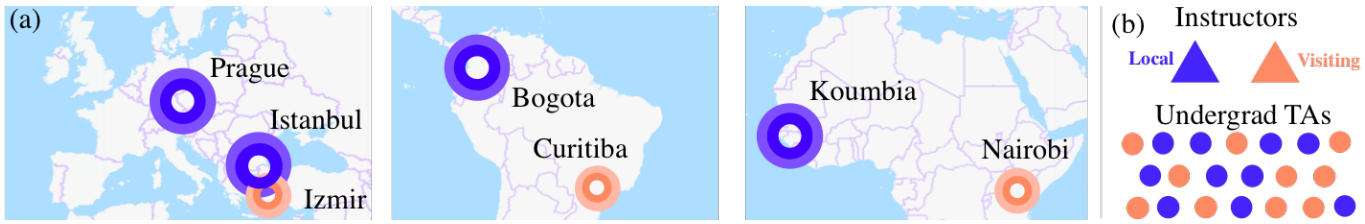


Figure 1. (a) CS Bridge locations. Courses based on CS Bridge; (b) A typical cross-border co-teaching team, comprised of an equal number of local and visiting teachers.

With the goal of CS for *All Countries*, we developed CS Bridge, a curriculum designed to support cross-border co-teaching. The course is jointly taught to students in diverse communities around the world to reinforce and bootstrap local communities of CS practice, as well as to reduce the friction preventing the spread of curricula and pedagogy across borders. We discuss two key contributions in this paper: We provide a course-in-a-box based on a prominent US R1 university's CS1 course that instructors can adapt to a multitude of contexts.[1] The course is designed to be linguistically simple, to foster creativity through programming, and to be compatible with both high school- and university-level classroom learning. Our second contribution is a course centered around *cross-border co-teaching*, where local instructors and student-teachers (i.e., undergraduate teaching assistants, or UTAs) actively participate in shaping the course and teaching it to students in their own communities. By combining a high-quality CS1 curricula with a blended group of motivated instructors (Figure 1b), we aim to boost and sustain interest in studying and teaching computer science.

This paper was written by eight instructors from around the world who were brought together by CS Bridge. Since 2014, we have taught CS Bridge ten times in three countries to over 1,000 high school students, with a team of over 100 UTAs and lecturers. Our goal is to add one new host country per year—as we learn from each other and our students, we adapt the curriculum and the way we teach. In this paper, we evaluate the effectiveness of the curriculum and highlight educational outcomes for the students and UTAs. One big lesson for us has been the enthusiasm it has fostered among the UTAs, who have built a thriving community of their own, many of whom have returned to teach again, or have gone on to teach elsewhere. This last outcome is perhaps the most scalable effect of small-group, in-person, cross-border co-teaching.

While most of our effort has focused on three university-based programs in Istanbul, Turkey; Bogotá, Colombia; and Prague, Czech Republic, other teachers have successfully adopted and adapted the CS Bridge curriculum—ranging from Peace Corps Volunteers, to a course in Guinea, to a community college and a large research university. We close our paper with a case study of Koumbia, Guinea, where we adapted CS Bridge to a non-university, rural context.

[1]Our course-in-a-box is available at http://course.csbridge.org.

PRIOR WORK

CS Bridge presents a novel *combination* of ideas: co-teaching, how to create inclusive and engaging curricula, and international CS teaching. As such, it builds upon a large body of work from several fields of thought in the computer science education community.

Learning to program for non-English speakers presents substantial challenges [7, 13, 43]. There have been many projects to teach programming outside the US—the Bermuda project and AddisCoder, among others [2, 23]. App Inventor, Code.org, and Scratch have been three large-scale efforts to support multilingual resources in block-based programming languages for K-12 students around the world [16, 32, 42, 44]. In particular, App Inventor's ability to deploy student code to Android phones is a powerful feature. The design and pacing of the curriculum is then left to local instructors to integrate into their courses. Our goal in *co*-teaching CS1 is to use curricula that local instructors and UTAs could envision in their own classrooms. Many international universities have introductory courses in languages like Java, C, C++, or Python, and we sought to teach a course that local teachers were familiar with, where they could actively contribute content and incorporate ideas back into their classrooms.

Co-teaching has a rich literature [14] especially with respect to inclusive education [21, 38]. We believe it could be one way to address the identified issues in diffusion of technology [1, 22, 33].

Service-learning—where students travel to other countries for community service projects—is well-practiced as a way to enhance social good in computing education [4, 6, 11, 15, 28]. While service-learning can be beneficial for students to learn about social responsibility and empathy, it can have negative impacts on the host community [24]. Cross-border co-teaching brings many of the same benefits for the hosts and guest community. However, CS Bridge is different from a service project as it is a *joint* project between colleagues from different countries; mitigating the costs.

MAIN CONTRIBUTIONS

CS Bridge Course-in-a-Box

The CS Bridge curriculum is an open-source Java-based course built around the popular ACM Java libraries and informed by decades of shared ideas in the CS education community [34]. The course assumes no prior knowledge and is built so that all students can—by the end—build the Breakout game from

scratch [26] and make their own creative final project. The primary learning goal is to inspire students to choose to study and practice computer science. It is intentionally designed to be a high-quality course for any country; unlike traditional curricula, CS Bridge has been especially tailored to be linguistically simple and easy to translate and localize. The contribution of this paper includes CS Bridge as an open-source and free to use CS1 curriculum-in-a-box: assignments, a clonable website, slides, lesson plans, and more. It is built for expert and novice teachers alike.

Cross-Border Co-Teaching

In CS Bridge, university instructors and undergraduate teaching assistants from different countries join together to collectively teach a course in each country. We argue that co-teaching is a surprisingly effective way to facilitate the open transfer of pedagogical ideas, cultural understanding, and technical know-how. Our model bridges communities that otherwise would have few channels for sharing CS teaching insight.

Motivation and Course Philosophy

Participation in CS for All

Every country in the world should be working towards fostering a thriving software industry within its economy. While tremendous strides have been made in online education, large swaths of the world lack the prerequisite skills and technology to access the benefits of online education. CS Bridge can serve as a catalyst or spark for people to begin engaging in independent learning. Doing so requires work across borders. We choose to *join* together to collaboratively teach a course—as opposed to having outsiders dictate decisions to local hosts. To that end, our model is built for sharing knowledge both ways.

Inspiring + Practical Hands on Learning

Decades of work in the broader CS education community have produced CS1 assignments that enable students to learn by doing, while simultaneous inspiring them to want to learn more [17]. Engaging coursework can demystify CS and—if presented at the right time—can help students choose to pursue computer science as a profession. We want to inspire beginner students to not only use software, but also to create it. Our initial intention was to bring these experiences to a broad set of learners from around the world in a culturally appropriate manner.

Fostering Near-Peer Teaching

The near-peer student teachers approach has been used in a wide variety of CS1 contexts [9, 30, 31, 35]. This teaching model has two advantages for students—cognitive and social congruence. In CS Bridge, we use undergraduate teaching assistants (UTAs) as near-peer teachers. Because UTAs themselves are in the process of becoming experts, they often can explain difficult problems to students in the same social and cultural language. Moreover, they can also serve as close role models. The support communities that form around near-peer teaching can be of critical importance for learners.

We believe that teaching is an exciting opportunity to foster interest in CS education. We hope that UTAs experience the joy of teaching, so that they consequently identify as—and aspire to become—professional CS instructors. By offering the program in the same location over consecutive years, we also grow a community of driven, experienced UTAs who are willing to teach each other how to teach. We coach UTAs by adapting best practices from an existing, year-round UTA program [35]. Throughout the 2-week course, we also work with the local campus to assess if near-peer teaching can be integrated into university CS courses during the academic year.

Transfer of Organizational Know-How

There is a pressing need for better methods of sharing pedagogical ideas and (the equally important) associated institutional knowledge and technical infrastructure. In CS education, good ideas surrounding pedagogy, accessibility to CS, and institutional organization are often lost in translation, as much of an idea's effectiveness is often in the details and situational context. The need is especially dire for large-audience university courses run by a large staff team. While the US-based authors are affiliated with a top R1 university that has taught CS1 to tens of thousands of students, it is unclear how well this curriculum would translate, if at all, to a different university context of similar scale. We want to effectively channel educational expertise through a collaborative implementation of a single short course.

Applied Contexts

We emphasize that CS Bridge is designed to be as broadly useful as possible. We have directly implemented the course and corresponding model in four specific contexts (Figure 1), of which three were taught at universities and the fourth in a village in Guinea, West Africa.

Turkey taught (in English and Turkish) four out of six times at Koç University in Istanbul, Turkey; 200 students per course from high schools across the country.

Colombia taught (in Spanish) once at Universidad de Los Andes in Bogotá, Colombia; 100 students per course from high schools across the country.

Czech Republic taught (in English and Czech) twice at Czech Technical University in Prague, Czech Republic; 100 students per course from high schools across the country.

Guinea taught (in French) once at Koumbia High School in Koumbia, Guinea; 20 students. The village had a newly built computer cluster but with intermittent electricity and internet.

It is important to note that the universities listed above are some of the most prestigious institutions in their respective countries. This choice is mostly deliberate: to draw students from diverse communities to these city centers. These universities are also more likely to have experienced lecturers, strong UTAs, adequate classroom and technological resources, as well as residential options close to campus. We anticipate more flexibility as CS Bridge grows as a program—particularly as we gain experience in effective training programs for UTAs and grow our global teaching community.

External Courses

The curriculum is free-to-use, and as such it has featured in a series of other courses in Izmir and Istanbul, Turkey; Curitiba, Brazil; and Nairobi, Kenya. The CS Bridge curriculum was itself inspired by CS106A, a course taught at Stanford University in the United States to around 1,800 students a year. The improved course material that arose from cross-border co-teaching has now been integrated back into CS106A.

Pacing

All four courses were taught over the Northern Hemisphere summer in an intensive 2–3 week, full-day course. In Colombia, we encountered the logistical challenge of matching out-of-phase school calendars between the Northern and Southern Hemisphere because of out-of-phase academic calendars, but many countries have a multi-week holiday sometime in June–August. In many settings, this two-week structure best fit the needs of the host and instructors as well as the local students. The pacing of the course can easily be modified, and a majority of the external courses that have used the curriculum have taught the material over a longer period of time.

INTERNATIONAL CS1-IN-A-BOX

The course assumes no prior CS knowledge and aims to equip students with a sound understanding of basic concepts of programming and the skills to program an interactive game within two weeks, which can be extended as needed. Given the limited time, the program is highly efficient. Since the core learning goal is to inspire students to further pursue computer science, it is presented as a series of engaging, educational coding tasks.

Our course-in-a-box—as the name suggests—has all the resources that a novice or expert instructor would need to teach their own instance of the class. It comes with a personalizable website, code, slides, assessment tools, and of course assignments. Moreover we include tools to help run a team of section leaders and lesson plans with fun activities away from the computer.

Course Plan

The course covers Karel, Console, Graphics and Events to accelerate students to the point where they can code exciting graphical programs (see Figure 2 for course units and assignments). We fit the course into a intensive two-week program. The first day introduces control structures, loops and method/function writing via Karel the Robot [27]: a kick-start, variable-free API for a beginner, that allows for engaging problem solving from day one. The rest of the first week covers all the required concepts to start designing the Atari Breakout game [26]. The second week is dedicated to implementing Breakout and building a creative project. All assignments are designed with optional, creative extensions for students to go above and beyond the core goals (a path which most students take). This design facilitates course pacing across a heterogeneous group of students, and it encourages creativity early on in learning programming.

In the two-week course, the daily schedule includes a morning and an afternoon lecture, a morning discussion *section*—where

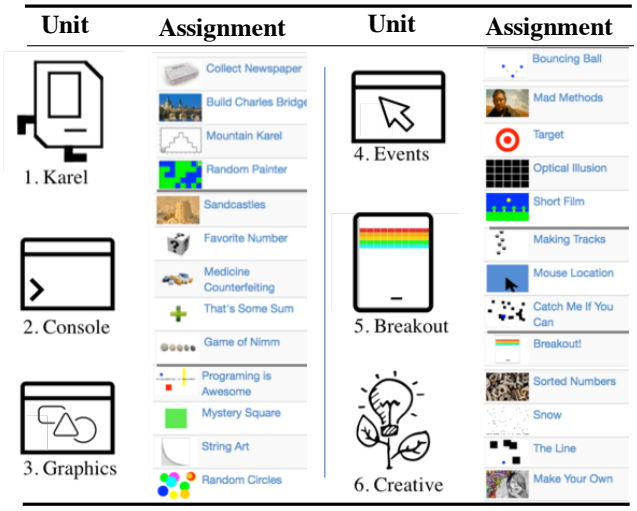


Figure 2. Two-week curriculum from Karel to open-ended Graphics.

UTAs lead a meeting with students for conceptual discussion—and morning and afternoon labs. Every day, an average of five programming tasks are implemented by the students during the labs. One of the most critical activities is the section: UTAs meet for an hour with 8–10 students as a group to consider possible strategies for a programming task and to review and discuss course concepts. These sections are essential to build a small community united towards the same target goals—learning the concepts and solving a problem—under the guidance of a mentor UTA. It is also the activity that trains the UTA as an effective educator, communicator, and future instructor.

In the second week of the course, the program hosts invited talks on different topics to increase student exposure to the broader domain of CS. We include modules on the inner workings of the internet, artificial intelligence, and a gentle introduction to JavaScript for web development. We also host chats where students can get to know the local and visiting instructors, as role models. This is an opportune time to encourage women in computing.

Built for Localization

The language of instruction plays a key role in guaranteeing equitable access. Students learn CS better when it is taught in their mother tongue [7, 8, 13, 19] and, as such, CS Bridge teaching and learning materials are designed to adapt easily to the local language of the participating students. The core material is written in English and comes with a program (powered by Google Translate, and adjustable by human translators) to translate the entire course into non-English languages. The base curriculum from which we translate is *low word count*: assignments, APIs, documentation and slides are all as visual as possible, and any explanation is written in Simplified Technical English [18]. This makes it easy to (1) translate and (2) read for non-English speakers. We have translated the entire course into Spanish, French, and Turkish. For Spanish and French, we also fully translated the student Java APIs (Karel, console, and graphics) from the ACM Java task force libraries

[34, 36]). For example, in Spanish the method `println()` became `imprimir()`. We have published this automatic translating tool, from which anyone can then override the APIs or text and thus translate into a new language.

The course activities are designed *and can be redesigned* to be relevant and interesting in different countries. K-12 students are motivated in surprisingly similar ways around the world—for example, Karel is loved everywhere. For certain projects the assignments are written to be culturally translated. As an example in the Czech Republic, students have Karel repair Charles Bridge,[2] whereas in Turkey, they repair Efes. In Guinea, they build the Grand Mosque of Conakry. The challenge is the same, but the narrative is localized.

CO-TEACHING ACROSS BORDERS

Because human teaching remains an essential element for successful education, it is important to collectively develop our ability to educate. Early-career instructors usually begin by teaching in the same way as they were once taught; improving and innovating upon teaching style tends to be a slow process of trial and error.

On the other hand, co-teaching can be more efficient and faster way to spread high-quality pedagogy, possibly across cultural boundaries. Teachers can compare, discuss and even inspire among themselves. Moreover, less experienced teachers can do more than just learn. With their new and young point of view, the young teachers can inspire their experienced counterparts as well.

We implement co-teaching for both instructors and UTAs (Figure 1b). All teachers come together as a team to help make the course as effective as possible for the students. The instructors select the material and alternate giving each lecture. Visiting and local UTAs run labs together and are jointly responsible for helping students when they get stuck. In Prague and Istanbul, the class is offered in a mix of simplified English and the local language, and as such, students need basic proficiency in English. In Colombia and Guinea, we require that members of the teaching team are fluent in both English and the local language.

Financial Considerations

The CS Bridge model is built to be as economical as possible. The financial requirements are primarily salary for both instructors and UTAs (hired at a student-teacher ratio of 1:10), followed by travel and housing expenses for visiting teachers and visiting UTAs. We note that employing UTAs dramatically reduces the cost, among providing other pedagogical benefits. The host campus provides access to computer labs and facilities at no cost. To minimize additional costs, we suggest that visiting teachers use home-stays or any accommodation the host university may have available.

The total budget for teaching a two-week, full-time class of 200 students is approximately $15,000 in each country—though this cost varies substantially between countries. We do not charge students as we want to increase access to CS for All; in many countries, the host university also provided accommodations for students who did not live locally. Our budget thus far has been graciously covered by a private benefactor who believes that the in-person co-teaching model is a highly impactful use of charitable funds.[3]

What it Takes to Get Started

Since the course-material is ready to go, the biggest step to starting your own instance is to develop a working relationship between the host and visiting community, around one year prior to the course. Three of our partnerships were created by students/colleagues currently at Stanford University who were from the host country. One of our partnerships was created by "cold-emailing" a university. Once the partnership is established, there are many paths to developing a healthy course. Our next step was to select instructors from both universities and to have them remotely plan the course together. Each party individually recruits UTAs; the local partner recruits students. The course-in-a-box includes a detailed list of timelines that we use before the course starts.

EDUCATIONAL OUTCOMES

In this section, we evaluate student learning by analyzing their work on the Breakout project; we use pre-post surveys to evaluate student perception of computer science before and after the program; and we share a few student anecdotes and final projects. However, the course benefits more than just the students—it is productive for the teachers and universities as well. We also how this experience impacts UTAs and university-level CS instruction beyond the summer program.

Student Outcomes

Assignment Learning Goals

The student projects are designed such that as students program, their work is saved to a local git repository. When students upload their final submission, they also submit the entire process by which they solved the problem. This offers us a rich and unique way to measure if students are achieving the learning goals. We analyzed the git repositories for Breakout submitted by high school students in Prague ($n = 73$ valid repositories out of 99 students) and Bogotá ($n = 87$ valid out of 99) during CS Bridge 2019 and plot assignment completion times in Figure 3. In all cases, the completion rate was high (95% in both programs); some repositories also had corrupted data and were not analyzed. More impressively, when we compared against repositories of Stanford University students, on average the CS Bridge students from Prague ($\mu = 5.8$) and Bogotá ($\mu = 6.2$) completed their assignments in significantly ($p < 0.001$) less time than Stanford students did ($n = 417, \mu = 7.4$). After the CS Bridge program, Stanford incorporated the improved pedagogy that came from co-teaching and time to complete Breakout subsequently dropped ($n = 487, \mu = 6.7$). This is noteworthy, especially considering that students at the university had substantially more self-reported prior experience.

[2] Karel the Robot is named after a Czech playwright; the name "Karel" is the Czech equivalent of "Charles." Charles Bridge is a historic bridge in Prague.

[3] CS Bridge became an official Stanford program starting in late 2019.

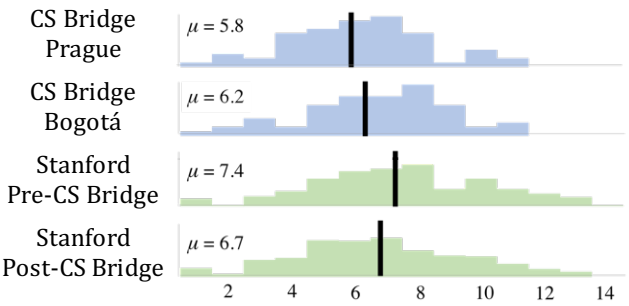
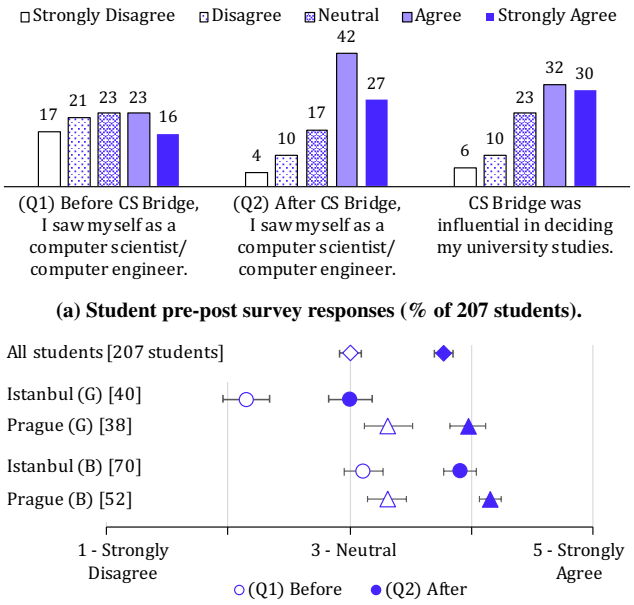


Figure 3. Histograms of Breakout completion times, in hours. Lower completion times means students finished the assignment faster.



(a) Student pre-post survey responses (% of 207 students).

(b) Average responses to Q1 and Q2 (±1 SE) by student demographic.

Figure 4. Student experience with the two-week university program. On average, students report a positive change in perceived identity in CS, though there were significant differences among sub-groups of students.

Interest in CS

To further understand the impact we were having on students and teachers we conducted a series of surveys. In the Istanbul course from 2015 to 2017, students reported immediately post-course that the course had impact: Most students enjoyed the course (96% strongly agree or agree on a 5-point Likert scale), were confident that they could program Breakout on their own (88%), and planned to continue programming (85%). From personal anecdotes, we found that many students were positively affected by their CS Bridge experience. During our first offering, a female student firmly stated on the first day that she had no idea why she was there, had no idea what CS was, and wanted to be a doctor. By the end of the course, she declared, "I've decided what I want to do: I want to be a computer scientist."

In 2018, we performed a mandatory pre-post survey in the Istanbul and Prague courses to assess perceived identity in CS with the statement, "I saw myself as a computer scientist/computer engineer," both before (Q1) and after (Q2) the program. We also surveyed student response to other course components, such as influence on university study and the UTA experience. We share 207 student responses to a subset of these questions on a 5-point Likert scale in Figure 4. On average, students reported a 0.77 point difference in perceived identity immediately after the course, though this difference was not significant ($p \approx 0.5$). In Figure 4b, we also report the average responses to Q1 and Q2 split by four different student demographics: self-identified boy/girl, and Istanbul/Prague courses. None of these groups reported a significant difference in perceived identity in CS before/after the program ($p \approx 0.5$ for all groups). However, we note that compared to girls in the Prague course, girls in the Istanbul course reported a significantly lower perceived identity both before the program (Istanbul $\mu = 2.15$, Prague $\mu = 3.31$, $p < 0.001$) and after (Istanbul $\mu = 3.00$, Prague $\mu = 3.97$, $p < 0.001$).

From our survey, 55% of students felt that having a UTA was important to their CS Bridge experience. 62% of students agree that the program was influential in deciding university study. We share additional long-response student comments in Table 1.

We conducted an additional, optional survey twice—in 2017 and 2019—to assess if the program had lasting impact on university study. 115 students responded, all of whom completed the program between July 2016 and July 2019. In Table 2 we report the university majors of current university students and intended majors of current high school students. While we observe a large fraction of students intending to pursue a degree in CS or computer engineering, we cannot conclude from this optional survey whether CS Bridge was the main factor in their decision. As a result of these surveys and personal anecdotes, we can at least gather that we have succeeded in giving the students a memorable, positive experience and *exposing* them to the field of computer science, perhaps facilitating a more informed decision about their futures.

	Describe one impact that CS Bridge had on you.
1	"I saw a new and interactive educational approach, which was vastly different from what I was used to within the Turkish curriculum."
2	"I formed one of the strongest relationships in my life with a person I've never met."
3	" I know I won't be a computer engineer, but I will be a doctor who loves CS."
4	"…that it is never too late to start programming, which encouraged me to choose CS at university."
5	"…that a computer scientist is a universal job. … one cannot say 'I don't use that website because the person who [coded it] was a woman or was from [some] country.' I think that is a very happy thought."

Table 1. Selected student survey comments.

University Major (# Students)	Current (76)	Intended (39)
Computer Science/Computer Eng.	53%	38%
Electrical Eng.	11%	8%
Other STEM	26%	36%
Other	11%	18%

Table 2. Surveyed distributions (%) of university major.

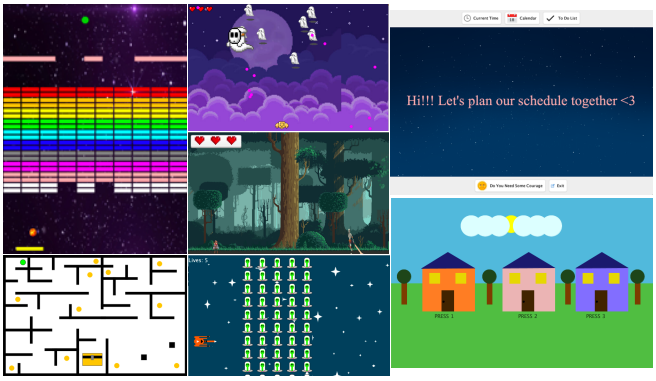


Figure 5. Sample final projects (Istanbul, 2019).

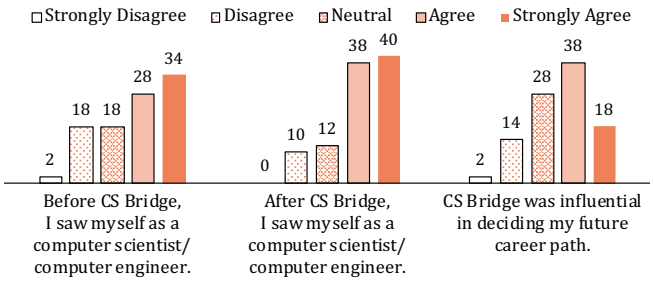
Final Projects

In the second week of the two-week course, students design and build a final, creative project. Individuals who hadn't programmed before were able to program complex, creative Java projects. Many students build interactive games (Figure 5), where they utilize graphics and web images to demonstrate a strong understanding over programming logic and design. Some students incorporate inside jokes from the CS Bridge program into their projects. The current direction of our curriculum explains the abundance of games, though some students elect to create schedule planners, text-based choose-your-own adventure games, and calculators for scientific applications. We find anecdotally that students are especially proud of their submissions to this open-ended project; at the end of the course, instructors share especially creative, innovative submissions with the group.

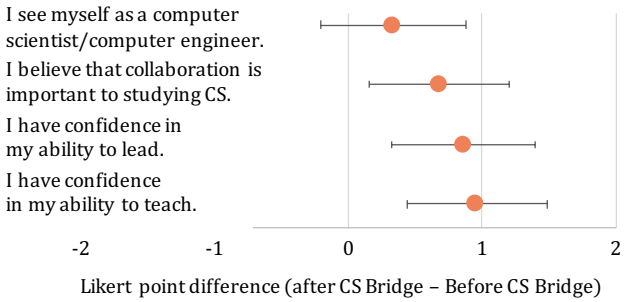
Undergraduate Teaching Assistant Growth

We surveyed 50 UTAs in late 2019 to understand how the program affected their career choices (Figure 6). Their experience was overwhelmingly positive, and they enjoyed the co-teaching community. As part of the survey, we asked UTAs to reflect on their experiences before and after the program. Figure 6b illustrates the mean change in Likert item responses (with bootstrapped standard error) after the program, compared to before the program. The results demonstrate small improvements in identity and perceived collaboration in the field of CS, and a large increase in teaching and leadership confidence, though none of these differences were significant ($p \approx 0.5$ for all questions).

Anecdotal comments reveal that the experience has been formative to boosting confidence CS and has inspired many UTAs to continue teaching (Table 3). The UTAs form friendships that shape their teaching identity. Many UTAs indicated interest in teaching in the future, either as an academic or as community service, and some mentioned that the UTA group was incredibly supportive and fun. In fact, almost half of our UTA staff from year to year and several graduated UTAs working in industry take vacation time to help with CS Bridge in following years.



(a) UTA pre-post survey responses.



(b) Average change (±1 SE) in UTA survey responses after the program.

Figure 6. UTA experience with the two-week university program.

How has CS Bridge impacted your career decision?	
1	"I would love to continue teaching. This is one of my strong reasons [for applying to a] PhD."
2	"I would like to be part of CS teaching activities even if I work in industry."
3	"I really started thinking about being a teaching member at university in the future."
4	"I can use my CS knowledge for [things outside of] research, like for community service."

Table 3. Selected UTA survey comments.

Four of the United States UTAs have gone on to become full-time lecturers in universities, both public and private. Several UTAs are current contributors to open-source teaching materials.

Instructor Outcomes

Co-teaching has proven to be a great way of spreading high-quality pedagogy and inspiration.

Clear Communication Improvement

In the CS Bridge course, it is crucial to explain each topic in the simplest way as possible, because we are teaching HS students from a range of cultural and linguistic backgrounds. Co-teaching has been a great way of finding the simplest explanations efficiently. This was especially apparent in the

development of metaphors for teaching CS1 concepts—e.g., variables are like boxes; methods are like toasters; pointers are like URLs; and so on. By co-teaching, we can also easily discuss challenges in teaching and together come up with better way of explaining difficult concepts.

Inspiration for Enjoyable Teaching and Learning

Co-teachers inspire each other to develop new and better ideas how to prepare lectures attractive to students. For example, this year the Czech lectures used special lecture themes that were then incorporated into future offerings of the course. High school students from every country appreciate fun and excitement during lecture—like the experience of using function calls to guide a classmate role-playing Karel the Robot around a physical map in the classroom. Instructors and UTAs encouraged each other to include jokes and personal experiences as part of teaching. In the future, we want to make our classrooms more inclusive by incorporating cultural differences in classroom dynamics into teacher training for handling difficult situations, such as fielding questions from advanced students without alienating beginners.

The results of this incredible professional development experience are largely intangible: we are all now connected to a growing community of teachers who care about the craft of CS1 education. The instructors of CS Bridge are writing this paper together as a testament to what we learned.

University Outcomes

For all universities, a major outcome is an increase in studying CS among high school students—which implies an increase of students enrolling in CS1 at the university level.

At Koç University, Istanbul, following the first two successful CS Bridge offerings, the university decided to adapt the UTA program to their CS1 course of 180 engineering students during the academic year. In order to design a system appropriate for their own cultural context, the instructors visited Stanford University to observe in-person the UTA program. The local instructors and UTAs involved in co-teaching in CS Bridge consequently formed the team to run the very first UTA program in Turkey, which has run successfully in Koç's CS1 course for four semesters. Consequently, the university is now planning to integrate UTAs in other engineering courses.

In addition, the Koç UTA group has formed a close-knit student group dedicated to designing and running new teaching activities. The student group has carried out two teaching and social responsibility projects: a weekend-based CS Bridge external program for local high school students and an unofficial CS Bridge-inspired CS1 course in Python for non-engineering Koç undergraduates. Instructors noted that all UTAs exhibited high individual growth—increased self-reliance, improved communication skills, and improved coding style.

TEACHING IN NON-UNIVERSITY CONTEXT: GUINEA CASE STUDY

Does CS Bridge work outside of a university context? In Koumbia we had the opportunity to adapt the program to an especially high-need, low-resource context. One of the authors of this paper had extensive experience teaching science in Koumbia, a farming village in rural Guinea—a 12-hour drive from the capital Conakry—where most families live without electricity or running water on less than a dollar a day. While students rarely touch computers, smart phones are ubiquitous.

Students and teachers alike expressed a desire to learn to use their technology productively both for the sake of learning and in order to have access to job opportunities. As such, in 2018 the local Association des Jeunes pour la Défense des Droits des Enfants (AJDE) installed a solar-powered technology center with 10 laptop computers.[4] CS Bridge collaborated with AJDE to expand their technology trainings to coding trainings, executing a *three*-week CS Bridge camp for 20 high school students ages 13–20. The teaching team consisted of one of the authors as instructor and three UTAs (one Guinean recent university graduate, one Guinean advanced high school student, and one current Peace Corps Volunteer in Koumbia).

Given the unique context, the structure differed from the other three CS Bridge programs: The curriculum was ported offline, and there were more adjustments to adapt assignments to local scenarios. For example, the first programming task involves teaching Karel to pick up a Newspaper—but in Koumbia there are no newspapers. The program was consequently changed to have Karel get water from a well. In Guinea, we paced the CS Bridge program around the existing infrastructure, computer proficiency, and cultural realities including daily power outages, and student's familial responsibilities. The curriculum was completed through the Events segment, at an average of two new concepts and two new programming tasks daily. While students did not build Breakout, they still enjoyed independent projects—creating their own creative Graphics, Karel, and Console projects.

Students were happy to gain employable skills, learn more about computers and build things; they were also excited to teach their friends and have fun. Several expressed interest in teaching for-pay in their own computer learning labs. UTAs similarly appreciated the chance to augment their learning. The program inspired two new iterations of CS Bridge in Guinea, and Peace Corps Guinea held a preliminary training session to train its volunteers to be code-instructors.

We emphasize that for future adaptations of CS Bridge to sociocultural contexts like Koumbia's, a slow and flexible pace is key. By pacing the course to the students, even those with basic computer skills (e.g., typing and mouse use) were able to complete programming tasks and benefit from CS Bridge.

DISCUSSION

Cross-Cultural Insights and Exchange

In some countries (e.g., Czech Republic, Turkey, and Guinea), the teacher-student relationship tends to be very formal. In the case of the Czech Republic, this phenomenon is connected with the Czech language, in which the border between formal and informal way of speaking is strict and context-dependent.

[4] One of the authors is a co-founder of AJDE. AJDE hypothesized that if parents saw their children learning job-marketable skills such as coding at school they would be motivated to support their children's education. In the 2019 school year, high school students learned typing, Excel, and Word.

Although Czech students do ask questions during lectures and other teaching forums, this cultural-induced barrier may be one reason why there is less interactivity during lectures compared to a United States-based environment. Teachers in all four teaching locations outside of the United States initially expressed doubts that UTAs would be accepted by students. However students and UTAs formed similar, positive near-peer relationships in all countries.

All universities learned pedagogy from each other and as such improved their CS1 offerings. Moreover, we all learned about the nuances of CS education in each other's countries, including university entrance requirements, regional differences in access, and more.

Curriculum Changes

The CS Bridge course-in-a-box in its current form uses Java to teach key introductory concepts and inspire creativity in CS. We acknowledge that there are many language options for teaching university-level CS1, and we have plans to implement a Python-based course as early as Summer 2020. Experiencing this change can be beneficial for local university instructors, who are also considering alternate CS1 languages and software tools. Overall, we stress that what is more important than our choice of programming language is our curriculum designed around open-ended tasks and graphical output. Such a course is accessible and interesting to a diverse set of learners [20, 26, 39].

Diversity and Inclusion

In all of our programs, we have emphasized the need for diversity and inclusion in both the teaching team and the student body. In the Istanbul course, we have achieved gender balance among not only students but also UTAs, who were also from a variety of STEM degree programs. The Turkish secondary education system has many strong public science and private schools across the country, meaning that there are fewer barriers to recruiting students from different regions. In Colombia, on the other hand, there are striking differences in access to education and quality of schooling between rural and urban areas [10, 29]. There was an active effort to invite, fund, and house students from outside of Bogotá (20% of students) to participate in CS Bridge. At the same time, girls composed only 20% of students in the program—and Colombia has one of the largest gender achievement gaps in math and science in the OECD [25]. Stanford, the visiting university, regularly recruits a gender- and background-diverse set of UTAs and instructors to co-teach CS Bridge—but we can all do better. In the future, we want to increase our commitment to *cross-border* education: fostering a diverse community around CS education in all countries, through active efforts in equal representation and access to education. The benefits of a diverse group of educators and learners is best experienced first-hand, and our program is an ideal starting point.

External Use

CS Bridge has inspired spinoffs in Kenya (NaiCode), Guinea (gncode.org), and Brazil (cs106r.com). One of the authors of this paper, an instructor who taught in two CS Bridge-Istanbul events, is also initiating a UTA program in his university's Electrical Engineering department. In collaboration with Peace Corps Guinea, CS Bridge had planned a national camp to train teachers and students for July 2020. Unfortunately, the program had to be put on hold due to the evacuation of Peace Corps Volunteers during the COVID-19 pandemic. However, we plan to continue our partnership once Peace Corps Volunteers are reinstated. We invite anyone to use and extend our work.

CONCLUSION

In this paper, we presented CS Bridge, a CS1 curriculum-in-a-box translatable to different languages and adaptable to multiple cultural contexts. When paired with our model for cross-border, cross-cultural co-teaching, we move one step closer to the goal of CS for All Countries. Our experience with CS Bridge—while qualitative—has demonstrated a positive impact on students and teachers alike, and it is a strong indication that joint teaching is an fruitful direction to pursue in the future. We are most excited about the idea of a global UTA community, where small, high-quality teaching experiences can scale access to CS education. We hope that our longest-lasting contributions to the international growth of CS education are in the form of in-person, human connections.

REFERENCES

[1] Philip Achimugu, Oluwatolani Oluwagbemi, and Adeniran Oluwaranti. 2010. An evaluation of the impact of ICT diffusion in Nigerias higher educational institutions. *Journal of Information Technology Impact* 10, 1 (2010), 25–34.

[2] BJ Barron, CK Martin, and ES Roberts. *DESIGNING A COMPUTER SCIENCE CURRICULUM FOR BERMUDAS PUBLIC SCHOOLS*. Technical Report. Stanford University.

[3] Sigrid Blömeke, Nils Buchholtz, Ute Suhl, and Gabriele Kaiser. 2014. Resolving the chicken-or-egg causality dilemma: The longitudinal interplay of teacher knowledge and teacher beliefs. *Teaching and Teacher Education* 37 (2014), 130–139.

[4] Christopher H. Brooks. 2008. Community Connections: Lessons Learned Developing and Maintaining a Computer Science Service-Learning Program. *SIGCSE Bull.* 40, 1 (March 2008), 352356. DOI: http://dx.doi.org/10.1145/1352322.1352256

[5] Wenhong Chen and Barry Wellman. 2004. The global digital divide-within and between countries. *It&Society* 1, 7 (2004), 39–45.

[6] Randy W. Connolly. 2012. Is There Service in Computing Service Learning? In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE 12)*. Association for Computing Machinery, New York, NY, USA, 337342. DOI: http://dx.doi.org/10.1145/2157136.2157238

[7] Sayamindu Dasgupta and Benjamin Mako Hill. 2017. Learning to Code in Localized Programming Languages.

In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale (L@S 17)*. Association for Computing Machinery, New York, NY, USA, 3339. DOI: http://dx.doi.org/10.1145/3051457.3051464

[8] Sayamindu Dasgupta and Benjamin Mako Hill. 2018. How "Wide Walls" Can Increase Engagement: Evidence From a Natural Experiment in Scratch. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI 18)*. Association for Computing Machinery, New York, NY, USA, Article Paper 361, 11 pages. DOI: http://dx.doi.org/10.1145/3173574.3173935

[9] Paul E. Dickson. 2011. Using Undergraduate Teaching Assistants in a Small College Environment. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE 11)*. Association for Computing Machinery, New York, NY, USA, 7580. DOI:http://dx.doi.org/10.1145/1953163.1953187

[10] Luis Gamboa and Erika Londoño. 2015. Assessing Educational Unfair Inequalities at a Regional Level in Colombia. *Lecturas de EconomÃa* (12 2015), 97 – 133. http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0120-25962015000200004&nrm=iso

[11] Michael Goldweber, John Barr, Tony Clear, Renzo Davoli, Samuel Mann, Elizabeth Patitsas, and Scott Portnoff. 2013. A framework for enhancing the social good in computing education: a values approach. *ACM Inroads* 4, 1 (2013), 58–79.

[12] Ruth Graham. 2018. The global state of the art in engineering education. In *Massachusetts Institute of Technology (MIT) Report*. Massachusetts, USA.

[13] Philip J. Guo. 2018. Non-Native English Speakers Learning Computer Programming: Barriers, Desires, and Design Opportunities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI 18)*. Association for Computing Machinery, New York, NY, USA, Article Paper 396, 14 pages. DOI:http://dx.doi.org/10.1145/3173574.3173970

[14] Charles Hannon, Manfred Huber, and Lisa Burnell. 2005. Research to Classroom: Experiences from a Multi-Institutional Course in Smart Home Technologies. *SIGCSE Bull.* 37, 1 (Feb. 2005), 121125. DOI: http://dx.doi.org/10.1145/1047124.1047395

[15] Leah H. Jamieson. 2002. Service Learning in Computer Science and Engineering. *SIGCSE Bull.* 34, 1 (Feb. 2002), 133134. DOI: http://dx.doi.org/10.1145/563517.563391

[16] Filiz Kalelioğlu. 2015. A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior* 52 (2015), 200–210.

[17] Judy Kay, Michael Barg, Alan Fekete, Tony Greening, Owen Hollands, Jeffrey H Kingston, and Kate Crawford. 2000. Problem-based learning for foundation computer science courses. *Computer Science Education* 10, 2 (2000), 109–128.

[18] Joseph Kimble. 1994. Answering the critics of plain language. *Scribes J. Leg. Writing* 5 (1994), 51.

[19] Thomas Andrew Kirkpatrick. 2011. Internationalization or Englishization: Medium of instruction in today's universities. (2011).

[20] Lucas Layman, Laurie Williams, and Kelli Slaten. 2007. Note to Self: Make Assignments Meaningful. In *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education (SIGCSE '07)*. ACM, 459–463. DOI: http://dx.doi.org/10.1145/1227310.1227466

[21] David Lowe, Steve Murray, Lothar Weber, Michel de la Villefromoy, Archie Johnston, Euan Lindsay, Warren Nageswaran, and Andrew Nafalski. 2009. LabShare: Towards a national approach to laboratory sharing. In *Proceedings of the 20th Annual Conference for the Australasian Association for Engineering Education*. The School of Mechanical Engineering, University of Adelaide, 458–463.

[22] Jimmy K.N. Macharia and Theunis G. Pelser. 2014. Key factors that influence the diffusion and infusion of information and communication technologies in Kenyan higher education. *Studies in Higher Education* 39, 4 (2014), 695–709.

[23] Jelani Nelson. 2020. AddisCoder. (2020). Retrieved June 3, 2020 from https://www.addiscoder.com/.

[24] Grace Ngai and Stephen C.F. Chan. 2015. How Much Impact Can Be Made in a Week? Designing Effective International Service Learning Projects for Computing. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education (SIGCSE 15)*. Association for Computing Machinery, New York, NY, USA, 645650. DOI: http://dx.doi.org/10.1145/2676723.2677267

[25] OECD. 2020. *Results for countries and economies*. 288 – 299 pages. DOI:http://dx.doi.org/https://doi.org/https://doi.org/10.1787/b9935c8e-en

[26] Nick Parlante, Steven A. Wolfman, Lester I. McCann, Eric Roberts, Chris Nevison, John Motil, Jerry Cain, and Stuart Reges. 2006. Nifty Assignments. *SIGCSE Bull.* 38, 1 (March 2006), 562563. DOI: http://dx.doi.org/10.1145/1124706.1121516

[27] Richard Pattis. 1995. Karel J Robot. (1995).

[28] Tarsem S. Purewal Jr, Chris Bennett, and Frederick Maier. 2007. Embracing the social relevance: computing, ethics and the community. *ACM SIGCSE Bulletin* 39, 1 (2007), 556–560.

[29] Thomas Radinger, Alfonso Echazarra, Gabriela Guerrero, and Juan Pablo Valenzuela. 2018. *OECD Reviews of School Resources: Colombia 2018*. 304 pages. DOI:http://dx.doi.org/https://doi.org/https://doi.org/10.1787/9789264303751-en

[30] Stuart Reges. 2003. Using Undergraduates as Teaching Assistants at a State University. *SIGCSE Bull.* 35, 1 (Jan. 2003), 103107. DOI: http://dx.doi.org/10.1145/792548.611943

[31] Stuart Reges, John McGrory, and Jeff Smith. 1988. The Effective Use of Undergraduates to Staff Large Introductory CS Courses. *SIGCSE Bull.* 20, 1 (Feb. 1988), 2225. DOI: http://dx.doi.org/10.1145/52965.52971

[32] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. 2009. Scratch: Programming for All. *Commun. ACM* 52, 11 (Nov. 2009), 6067. DOI: http://dx.doi.org/10.1145/1592761.1592779

[33] Amal Rhema and Iwona Miliszewska. 2014. Analysis of student attitudes towards e-learning: The case of engineering students in Libya. *Issues in informing science and information Technology* 11, 1 (2014), 169–190.

[34] Eric Roberts, Kim Bruce, James H. Cross, Robb Cutler, Scott Grissom, Karl Klee, Susan Rodger, Fran Trees, Ian Utting, and Frank Yellin. 2006. The ACM Java Task Force. *SIGCSE BULLETIN* 38, 1 (2006), 131.

[35] Eric Roberts, John Lilly, and Bryan Rollins. 1995. Using Undergraduates as Teaching Assistants in Introductory Programming Courses: An Update on the Stanford Experience. *SIGCSE Bull.* 27, 1 (March 1995), 48–52. DOI:http://dx.doi.org/10.1145/199691.199716

[36] Eric Roberts and Antoine Picard. 1998. Designing a Java graphics library for CS 1. *ACM SIGCSE Bulletin* 30, 3 (1998), 213–218.

[37] Lee S. Shulman. 1986. Those who understand: Knowledge growth in teaching. *Educational researcher* 15, 2 (1986), 4–14.

[38] Michael Solis, Sharon Vaughn, Elizabeth Swanson, and Lisa Mcculley. 2012. Collaborative models of instruction: The empirical foundations of inclusion and co-teaching. *Psychology in the Schools* 49, 5 (2012), 498–510.

[39] Elizabeth Sweedyk, Marianne deLaet, Michael C. Slattery, and James Kuffner. 2005. Computer Games and CS Education: Why and How. In *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education (SIGCSE '05)*. ACM, 256–257. DOI: http://dx.doi.org/10.1145/1047344.1047433

[40] UNESCO. 2019. UIS.Stat. http://data.uis.unesco.org/. (2019). [Online; accessed 19-July-2019].

[41] David W. Valentine. 2004. CS Educational Research: A Meta-Analysis of SIGCSE Technical Symposium Proceedings. *SIGCSE Bull.* 36, 1 (March 2004), 255259. DOI:http://dx.doi.org/10.1145/1028174.971391

[42] Amber Wagner, Jeff Gray, Jonathan Corley, and David Wolber. 2013. Using App Inventor in a K-12 Summer Camp. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE 13)*. Association for Computing Machinery, New York, NY, USA, 621–626. DOI: http://dx.doi.org/10.1145/2445196.2445377

[43] Ben Williamson, Annika Bergviken Rensfeldt, Catarina Player-Koro, and Neil Selwyn. 2019. Education recoded: policy mobilities in the international learning to codeagenda. *Journal of Education Policy* 34, 5 (2019), 705–725.

[44] David Wolber. 2011. App Inventor and Real-World Motivation. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE 11)*. Association for Computing Machinery, New York, NY, USA, 601–606. DOI: http://dx.doi.org/10.1145/1953163.1953329