

Designing Think-aloud Interviews to Elicit Evidence of Computer Science Pedagogical Content Knowledge

In recent years, several organizations in the U.S. have initiated ambitious programs to drastically increase computer science (CS) course offerings in K-12 schools (e.g., Astrachan, Cuny, Stephenson, & Wilson, 2011; Chicago Public Schools, 2014; San Francisco Unified School District, 2015; Snyder, 2013), requiring the development of a massive workforce versed in both CS content and pedagogy. In-service teachers trained in other disciplines are often assigned to CS classes due to a dearth of certified practitioners, creating a group of educators with varying backgrounds and professional development needs (Gal-Ezer & Stephenson, 2010). Our work focuses on understanding what experiences best support in-service teachers to develop the pedagogical content knowledge (PCK) needed for effective CS instruction. As part of our early stage research, we report on our efforts to create think-aloud interviews that elicit evidence of teachers' CS PCK.

Theoretical Framework

Shulman (1986) proposed the construct of PCK to describe the body of knowledge needed for teaching a particular subject. While researchers have studied PCK extensively in other domains like mathematics (Depaepe, Verschaffel, & Kelchtermans, 2013) and science (Van Driel, Verloop, & De Vos, 1998), PCK research on CS is inchoate. Studies produced by the CS education community have focused on defining, measuring, and developing PCK (e.g., Buchholz, Saeli, & Schulte, 2013; Hazzan, Lapidot, & Ragonis, 2011; Hubwieser, Magenheimer, Mühling, & Ruf, 2013; Liberman, Ben-David Kolikant, & Beerli, 2009; Ohrndorf & Schubert, 2013; Saeli, Perrenet, Jochems, & Zwaneveld, 2012).

Moreover, many of these endeavors have been situated in educational contexts outside the U.S. where CS licensure at the pre-service level is more accessible. The large number of teachers in the U.S. who are transitioning into CS with varying PCK strengths and weakness warrants more research within this unique milieu to determine the feasibility of such a workforce development model.

Selecting methods to explore PCK requires attending to the nature of teacher knowledge. Hashweh (2005) described teacher knowledge as units of knowledge that are private and personal, content-specific, event-based and story-based, and developed through repeated experiences of planning and teaching particular topics. Kagan (1990) noted that teacher knowledge is sometimes stored in metaphors, can reside at an unconscious level, and may not be describable by teachers. Given the often tacit and multifarious nature of teacher knowledge, researchers attempting to elicit PCK need to employ multiple methods and ask teachers to articulate their knowledge (Baxter & Lederman, 1999).

The objective of this paper is to describe our development and initial results of a multi-method approach to gather evidence of in-service teachers' CS PCK. This work supports CS education research by adding to methodologies used to understand teacher knowledge. While we provide an overview of our entire process, we concentrate on one facet of our methodology: think-aloud interviews. Specifically we ask, how can think-aloud interviews be designed to elicit evidence of PCK?

Study Context

Our research project explores CS PCK within the context of TEALS (<http://tealsk12.org>), a multi-year program that pairs volunteers from the high tech field

with teachers to offer CS courses in high schools across the United States. TEALS recruits and trains volunteers to serve as part-time teachers and offers curricular materials and teaching support throughout the year. Multiple partnership models are offered through TEALS. In the co-teaching model, volunteers attend class in person and share instructional responsibilities with high school teachers. In the teaching assistants model, volunteers attend class in person and provide support to high school teachers where needed (e.g., grading assignments). In the remote model, volunteers attend class virtually through teleconferencing software and share instructional responsibilities with high school teachers. Our work focuses on the co-teaching partnership model. In this model, volunteers provide subject-matter knowledge and teachers provide pedagogical expertise. At the beginning of the partnership, volunteers lead CS classes while teachers learn course content. Over time, teaching responsibilities shift from volunteers to teachers, with teachers leading the course independently after two years.

We employ a concurrent triangulation design to study CS PCK within the TEALS program. The quantitative arm of our work consists of background surveys, an attitude survey (Dorn & Tew, 2015), a PCK assessment, and assessments of content knowledge. The qualitative arm of this project involves six case studies with teaching teams located in California. Teams are visited monthly and complete questionnaires about their lesson plans, team preparation, and student difficulties. An observation protocol is used to record classroom activities, instructor roles, and teaching practices. Lastly, teachers participate in an interview focused on teaching CS. In the following sections, we detail the development of these interview protocols and our initial findings.

Methodology

The development of our think-aloud interviews consisted of: (1) identifying types of prompts to include, (2) aligning prompts to curricula, and (3) designing prompts to reflect situations encountered in classrooms. Fourteen interviews were created to evoke teachers' memories of recent pedagogical experiences and drive discussion about their perspectives on instruction. Details on each step are described below.

Types of Prompts

Three types of prompts were selected for the interviews: assessment, student work, and participation. *Assessment prompts* asked participants to review three items and decide which should be included on an exam covering a particular topic (see Appendix A). This prompt type was borrowed from Davis (2004), who used a similar technique to explore science teachers' content knowledge. *Student work prompts* asked participants to review and title three student solutions to a programming problem (see Appendix B). This prompt type was modified from an activity created by Hazzan, Lapidot, and Ragnois' (2011) to support CS educators in examining programming tasks that can be solved in a variety of ways. *Participation prompts* were inspired by equity-based teaching practices identified by Margolis, Goode, Chapman and Ryoo (2014) and topics covered in *Blown to Bits* (Abelson, Ledeen, & Lewis, 2008), which is recommended reading for TEALS courses (see Appendix C). After each prompt, teachers were asked to discuss how their own students might respond to the items. We attempted to alternate prompt types so teachers responded to a different type of prompt each visit.

Aligning Prompts to Curricula

After selecting prompt types, we focused on aligning interviews with topics covered in the AP Computer Science A and Introduction to CS Principles courses. We reviewed

TEALS curricular guides and identified one topic for each unit to address in our interviews. To coordinate interviews with teachers' recent classroom experiences, we asked teachers to identify the topic of their lessons a few days before our visit and we selected the prompt most closely related to that topic.

Designing Realistic Prompts

We also focused on designing tasks that might simulate teachers' everyday experiences. While designing assessment prompts, we drew on a classification of twelve question types used in CS teaching (Hazzan et al., 2011) to vary the types of items presented in each interview. For example, the prompt shown in Appendix A includes items that ask students to complete a solution (item 1), find the purpose of a solution (item 2), and examine the correctness of a solution (item 3). For a subset of student work prompts, we incorporated items reflecting common misconceptions. For example, the prompt shown in Appendix B reflects mental models students have of recursion (Götschi, Sanders, & Galpin, 2003). Lastly, we reviewed recommended course textbooks, AP CS A practice exam guides, a repository of assessments created by prior TEALS participants, and assessment instruments developed by a project advisor. Some interview items were borrowed and modified from these materials while others were created by our team.

Research Findings

Participants and Data

During the 2014-15 school year, six teachers were interviewed a total of twenty-seven times. Cases were selected through convenience sampling due to the rolling nature of our recruitment process. Table 1 summarizes the participants' background:

Table 1. Participant Background Information

Participant	Interviews	Main Subject	Years of Teaching Experience	Course*	Years with TEALS
Teacher 1	5	Science	6	Intro	1
Teacher 2	3	Math	37	Intro	1
Teacher 3	5	Math	12	Intro	2
Teacher 4	3	Math	10	AP	1
Teacher 5	7	Math	10	AP	1
Teacher 6	4	Math Digital Arts	>25	AP	3

**Intro is the Introduction to CS Principles course. AP is the AP Computer Science A course.*

The most frequently used protocols were selected for this first round of analysis. Interviews were conducted between January and May of 2015. Twelve interviews were transcribed and coded; two interviews were analyzed for each participant. The interviews span three protocols including: an assessment prompt focused on variables, functions, and data types that was completed by teachers in the introductory course (Appendix A); a student work prompt focused on recursion completed by teachers in the AP course (Appendix B); and a participation prompt completed by all teachers (Appendix C). The coding scheme was based on PCK components identified in the literature. Interviews were analyzed within cases and then across cases to identify prominent themes. Below we summarize a subset of patterns emerging from our analysis.

TPACK

TPACK is an extension of PCK that includes the knowledge needed to teach with technology (Koehler & Mishra, 2009). Elements of TPACK surfaced in teachers who switched either programming languages or programming environments during the school year. Two teachers needing to extend semester-long courses to a full year switched from teaching Snap! to either Python or JavaScript. For one teacher, this change in

programming languages elicited reflections about how each language supported student learning:

For loops, lists, the difference between the index of the lists, the item of the list [are] easier in Python because you don't see them next to each other...The problem [in Snap!] is when there's a list of numbers, then they'll get confused between the actual number item and the indexed number of that number.

For the second teacher, the change in programming languages seemed deleterious to his content knowledge. He struggled to solve problems with Snap! during the second semester. When deciding whether to include, omit, or revise item 2 of the assessment prompt, he noted:

I'm not sure. If I had just used Snap! yesterday, it would have been a little bit easier. But it's been a few months. I wouldn't know how to fix it up. I can't think of it right now.

Two AP CS teachers used multiple environments (i.e., BlueJ, Eclipse, Processing) to teach Java programming. The change in programming environments provoked comments about how each environment supports learning and motivation to code:

Between the visual component of the BlueJ and the fact that it's easier to install, I might go with BlueJ. But any kid that's used Eclipse doesn't want to go away from it because it does all that auto fill ...and then they also like the idea of using the real thing. Real engineers use Eclipse.

Teacher Preparation

All teachers discussed methods used to learn CS content. Four teachers, following the TEALS model as prescribed, relied mainly on learning content from their volunteers. Two of these teachers, both first-year TEALS participants, felt uncomfortable with course content and expressed a need for more preparation. The other two teachers, returning TEALS participants, felt prepared to teach lessons on their own. In contrast, two other first-year TEALS participants who were actively involved in lesson planning

and delivery from the onset often discussed participating in outside programs focused on CS teaching and culling through multiple references to select lesson materials:

The various materials that I consulted when I was putting together the [recursion] lessons always emphasized the base case and then the part where it calls itself...And I saw some stack diagrams that I might try next year.

Thus teachers may take longer to learn CS content by not being actively involved in teaching the course upfront or by not proactively seeking content knowledge outside of the classroom.

Student Understanding and Engagement

All teachers discussed student learning during their interviews, however, there was variation in the depth of discussion provided. For example, some teachers only identified student issues, while others also discussed how to address those issues. The teacher who has been participating in TEALS the longest was able to discuss common student issues, ways of presenting information conducive to how students learn, and linking content across courses:

Students seem to have a hard time with understanding why you need to do an interface. We try to do a lot of examples with interfaces...but you almost have to have a fairly complex program before it makes sense... maybe next year when we do the Android apps class...maybe it will make more sense then.

Four teachers also discussed their beliefs about CS learning and engagement. All of these teachers emphasized hands-on programming experiences to motivate students and not “*philosophizing about CS*”. Two teachers mentioned a tension between this hands-on approach and demands of the AP CS curriculum: “*Teaching to the AP is entirely different than the idea that I want the kids to explore and have fun and just come away with a love for computer science.*”

Conclusions

In this paper we asked how could think-aloud interviews be designed to elicit evidence of PCK and found our think-aloud interviews were useful in eliciting examples of CS teaching knowledge. The participation prompt elicited many comments about student engagement and beliefs about who should be taking CS courses. It may be useful to create more participation prompts to learn how such beliefs influence instructional decisions. The assessment prompt seemed less useful in encouraging discussion about particular classroom events. Instead, this prompt elicited broader generalizations about CS content, students, and instruction. While the interviews addressed aspects of content knowledge, they did not capture teachers' understanding of course topics. This deficiency might be addressed with additional tools, such as the CoRe (Loughran, Mulhall, & Berry, 2004), that ask teachers to explicitly discuss their thinking around big ideas in a domain, or a content assessment administered separately from the interviews.

The results of our preliminary analysis have informed modifications to our case study work that are currently being implemented in the second year of our project. First, student work prompts have been revised so that each student solution reflects a common misconception or way of thinking. The rationale for this modification was to implicitly encourage more discussion around the ways in which teachers address student difficulties and the ways in which they make sense of computer science content for instructional purposes. Also, we noticed that several participants discussed exchanging knowledge and strategies with other teachers. This gave us pause to consider if our roles as non-practitioner researchers influenced the type of feedback teachers provided for assessment prompts. To mitigate this potential impact, we reframed the assessment prompts so that teachers are asked to imagine they are giving advice to a colleague who is selecting items

for a test. Our next round of analysis will focus on identifying the areas of PCK teachers discuss during our case study visits, how the type of instrument used (e.g., lesson reflection questionnaire, think-aloud interview) relates to the types of PCK elicited, and the depth of knowledge reflected in teachers' comments. We hope this work will provide a better understanding of the PCK development of in-service teachers transitioning into CS classrooms.

References

- Abelson, H., Ledeen, K., & Lewis, H. (2008). *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion* (1 edition). Upper Saddle River, NJ: Addison-Wesley Professional.
- Astrachan, O., Cuny, J., Stephenson, C., & Wilson, C. (2011). The CS10K project: mobilizing the community to transform high school computing. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 85–86). ACM.
- Baxter, J. A., & Lederman, N. G. (1999). Assessment and measurement of pedagogical content knowledge. In *Examining pedagogical content knowledge* (pp. 147–161). Springer.
- Buchholz, M., Saeli, M., & Schulte, C. (2013). PCK and Reflection in Computer Science Teacher Education. In *Proceedings of the 8th Workshop in Primary and Secondary Computing Education* (pp. 8–16). New York, NY, USA: ACM.
<http://doi.org/10.1145/2532748.2532752>
- Chicago Public Schools. (2014). CPS Announces First Schools to Implement District's Comprehensive K-12 Curriculum [Press Release]. Retrieved from http://cps.edu/News/Press_releases/Pages/PR1_03_19_2014.aspx
- Davis, E. A. (2004). Knowledge integration in science teaching: Analysing teachers' knowledge development. *Research in Science Education*, 34(1), 21–53.
- Depaepe, F., Verschaffel, L., & Kelchtermans, G. (2013). Pedagogical content knowledge: A systematic review of the way in which the concept has pervaded

- mathematics educational research. *Teaching and Teacher Education*, 34, 12–25.
<http://doi.org/10.1016/j.tate.2013.03.001>
- Dorn, B., & Tew, A. E. (2015). Empirical validation and application of the computing attitudes survey. *Computer Science Education*, 25(1), 1–36.
<http://doi.org/10.1080/08993408.2015.1014142>
- Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation is critical. *ACM Inroads*, 1, 61–66.
- Götschi, T., Sanders, I., & Galpin, V. (2003). Mental Models of Recursion. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (pp. 346–350). New York, NY, USA: ACM.
<http://doi.org/10.1145/611892.612004>
- Hashweh, M. Z. (2005). Teacher pedagogical constructions: a reconfiguration of pedagogical content knowledge. *Teachers and Teaching*, 11(3), 273–292.
- Hazzan, O., Lapidot, T., & Ragonis, N. (2011). *Guide to Teaching Computer Science: An Activity-Based Approach*. Springer.
- Hubwieser, P., Magenheim, J., Mühlhling, A., & Ruf, A. (2013). Towards a conceptualization of pedagogical content knowledge for computer science. In *Proceedings of the ninth annual international ACM conference on International computing education research* (pp. 1–8). ACM.
- Kagan, D. M. (1990). Ways of Evaluating Teacher Cognition: Inferences Concerning the Goldilocks Principle. *Review of Educational Research*, 60(3), 419–469.
<http://doi.org/10.3102/00346543060003419>

- Koehler, M., & Mishra, P. (2009). What is Technological Pedagogical Content Knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education*, 9(1), 60–70.
- Liberman, N., Ben-David Kolikant, Y., & Beeri, C. (2009). In-service Teachers Learning of a New Paradigm: A Case Study. In *Proceedings of the Fifth International Workshop on Computing Education Research Workshop* (pp. 43–50). New York, NY, USA: ACM. <http://doi.org/10.1145/1584322.1584329>
- Loughran, J., Mulhall, P., & Berry, A. (2004). In search of pedagogical content knowledge in science: Developing ways of articulating and documenting professional practice. *Journal of Research in Science Teaching*, 41(4), 370–391. <http://doi.org/10.1002/tea.20007>
- Margolis, J., Goode, J., Chapman, G., & Ryoo, J. J. (2014). That Classroom “Magic”. *Commun. ACM*, 57(7), 31–33. <http://doi.org/10.1145/2618107>
- Ohrndorf, L., & Schubert, S. (2013). Measurement of Pedagogical Content Knowledge: Students’ Knowledge and Conceptions. In *Proceedings of the 8th Workshop in Primary and Secondary Computing Education* (pp. 104–107). New York, NY, USA: ACM. <http://doi.org/10.1145/2532748.2532758>
- Saeli, M., Perrenet, J., Jochems, W. M., & Zwaneveld, B. (2012). Pedagogical Content Knowledge in Teaching Material. *Journal of Educational Computing Research*, 46(3), 267–293.
- San Francisco Unified School District. (2015). Board Approves Plans to Expand Computer Science Curriculum to All Grades [Press Release]. Retrieved from

<http://www.sfusd.edu/en/news/current-news/2015-news-archive/06/board-approves-plans-to-expand-computer-science-curriculum-to-all-grades.html>

Shulman, L. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher*, 15, 4–14.

Snyder, L. (2013). An Interview with Hadi Partovi. *Commun. ACM*, 56(9), 41–45.
<http://doi.org/10.1145/2500133>

Van Driel, J. H., Verloop, N., & De Vos, W. (1998). Developing science teachers' pedagogical content knowledge. *Journal of Research in Science Teaching*, 35(6), 673–695. [http://doi.org/10.1002/\(SICI\)1098-2736\(199808\)35:6<673::AID-TEA5>3.0.CO;2-J](http://doi.org/10.1002/(SICI)1098-2736(199808)35:6<673::AID-TEA5>3.0.CO;2-J)

Appendix A

Assessment Think-Aloud

Prompt: Imagine you are creating an end-of-unit assessment focused on variables, functions, and data types. I will present you with a set of assessment items. For each item, tell me what learning objectives the item might help you assess, if it aligns with the goals of the unit, and if you would include it on the assessment.

ITEM 1

Consider the following block that simulates a dice game. The block rolls two die, determines if one of the die is greater, and keeps of list of the results:

```

set dice_rolls to list
repeat 5
  set my_die to pick random 1 to 6
  set your_die to pick random 1 to 6
  add [ ] to dice_rolls

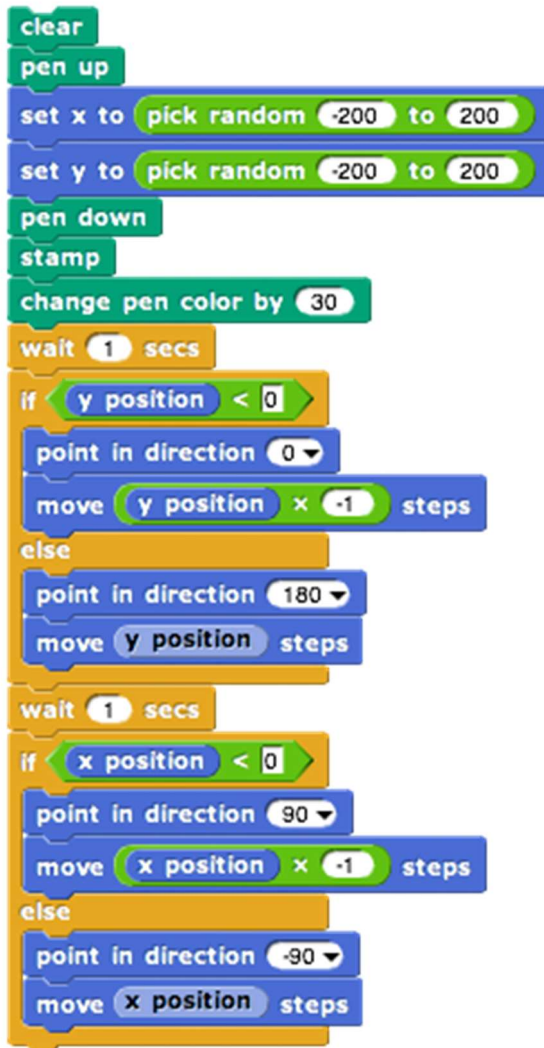
```

Notice that the add block is not complete. Which of the following completed add blocks would make the dice game run correctly?

- a)
- b)
- c)

ITEM 2

Provide an explanation of what this script does:



```
clear
pen up
set x to pick random -200 to 200
set y to pick random -200 to 200
pen down
stamp
change pen color by 30
wait 1 secs
if y position < 0
  point in direction 0
  move y position x -1 steps
else
  point in direction 180
  move y position steps
wait 1 secs
if x position < 0
  point in direction 90
  move x position x -1 steps
else
  point in direction -90
  move x position steps
```

The script performs a random walk on a 2D plane. It starts by clearing the canvas and lifting the pen. It then sets the initial x and y coordinates to random values between -200 and 200. The pen is lowered, and a stamp is placed at the starting point. The pen color is changed by 30 units. A 1-second wait follows. The script then enters a loop where it checks the y-coordinate. If the y-coordinate is less than 0, it points the pen to 0 degrees and moves the pen to the right by the absolute value of the y-coordinate (multiplied by -1). If the y-coordinate is greater than or equal to 0, it points the pen to 180 degrees and moves the pen to the left by the y-coordinate. A 1-second wait follows. The script then checks the x-coordinate. If the x-coordinate is less than 0, it points the pen to 90 degrees and moves the pen down by the absolute value of the x-coordinate (multiplied by -1). If the x-coordinate is greater than or equal to 0, it points the pen to -90 degrees and moves the pen up by the x-coordinate.

ITEM 3

Explain why the following script will not work as intended and propose a fix.

The script is as follows:

```

when clicked
  forever
    if key right-arrow pressed? or key left-arrow pressed?
      glide 1 secs to x: x position - 10 y: y position
      glide 1 secs to x: x position + 20 y: y position
  
```

The callout box contains the text: "Glide forwards then glide backwards when right arrow or left arrow is pressed".

Appendix B

Student Work Think-aloud

Prompt: Three students completed the following exercise. For each solution provided, review the solution, give it a title that captures the essence of the solution, and provide a rationale for your title.

EXERCISE

What value will be returned by Mystery(8)? By Mystery(1)? Show your work.

```
public static int Mystery (int n){
    if(n == 1){
        return 1;
    }
    else{
        return 3 + 4 * Mystery(n/2);
    }
}
```

SOLUTION 1

Mystery(8)	=	3 + 4 * Mystery (4)
	=	3 + 4 * (3 + 4 * Mystery(2))
	=	3 + 4 * (3 + 4 * (3 + 4 * Mystery(1)))
	=	3 + 4 * (3 + 4 * (3 + 4 * 1))
	=	3 + 4 * (3 + 4 * (7))
	=	3 + 4 * (31)
	=	127
Mystery(1)	=	1 (this is the base case)

SOLUTION 2

Mystery(8)	=	3 + 4 * Mystery (4)
------------	---	---------------------

	=	3 + 4 * Mystery(2)
	=	3 + 4 * Mystery(1)
	=	Mystery(1)
	=	1
Mystery(1)	=	1 (this is the base case)

SOLUTION 3

Mystery(8)	=	3 + 4 * Mystery(n/2)
	=	3 + 4 * 1
	=	7
Mystery(1)	=	1

Appendix C

Participation Think-aloud

Prompt 1:

Imagine that you are the sponsor for a new computer science club at your high school. Your school is hosting an after-school fair to help clubs recruit members. Each club is provided a booth to present information about their organization. You have access to any equipment you might need. What might you present at your booth to recruit students to the club?

Prompt 2:

You solicit the help of students enrolled in a computing course and ask them to submit ideas for the booth presentation related to their recent coursework. The class just read chapter 2 from the book *Blown to Bits* on privacy in the digital world. For each idea presented, discuss how the presentation might support or hinder student recruitment for the new club.

SOLUTION 1

I propose we do a presentation on how our privacy is being compromised every day in our community. I think we should identify some of the businesses in our neighborhood that we share digital data with. For example, we can talk about the printers at Kinko's that encode serial numbers and dates on documents we have printed. Or we could talk about the RFID tags they put in the books at the local bookstore. In addition to showing all the places where we leave digital footprints, we should also list some of the risks we put ourselves under by sharing that data. Then, we should have another section in the booth that describes how knowing computer science can help us address these issues.

SOLUTION 2

I propose we include a hands-on demo at our booth, something like a mini-hackathon. When people visit the booth, we can have them modify a program that we create ahead of time related to privacy issues. For example, we could create a program that encrypts messages and have the visitors adjust the encryption algorithm. Before they leave the booth, we can email the visitors the programs they modified.

SOLUTION 3

I think we should interview a few computer scientists who work on digital privacy issues and play the interview recordings at our booth. Our TEALS volunteer is friends with Latanya Sweeney, who was mentioned in chapter 2. We could interview her about the research she does on security. Also, one of my neighbors is a member of SHPE (the Society of Hispanic Professional Engineers) at the local college. We can interview him about the courses he has taken on cyber security and his internship at DARPA last summer.