Poster Abstracts – Session 2

ICER 2015

We are pleased to introduce a collection of posters as part of the ICER conference program. These posters intend to further expand the ICER community and spark discussion among conference participants. The goal is for these posters to provide a venue in support of both new ideas and newcomers to our community.

Like the main ICER program, the posters cover a wide range of computing education related work, and the presenters are from a variety of different institutions around the world. Short abstracts for each of the posters to be presented at this year's conference in Poster Session 2 can be found on the pages to follow. Note that a subset of these abstracts are also included as lightning talks as applicable.

Leo Porter
2015 Lightning Talks and Posters Chair
UC San Diego

Poster Session 2

1. *Peer-Led Team Learning Supports Computer Science Persistence at Community College*
   Kathryn I. Cunningham
   California State University Monterey Bay, USA

2. *Educating the Next Generation of Great Software Engineers*
   Leigh Ann DeLyser
   NYC Foundation for CS Education, USA

3. *The MoveLab: Supporting Diversity through Self-Conceptions*
   Kayla DesPortes
   Georgia Institute of Technology, USA

4. *Computer Science is "Hard": Looking at the Gender Gap Between Two Computing Programs*
   Brittany Ann Kos
   ATLAS Institute and University of Colorado Boulder, USA

5. *Middle School Students Learn Programming Using Wearable Technologies*
   Victoria Lentfer
   University of Nebraska Omaha, USA

6. *CSTeachingTips.org: Connecting Research and Practice*
   Colleen Lewis
   Harvey Mudd College, USA

7. *Storyteller: Telling Stories by Reflecting on Code Evolution*
   Mark Mahoney
   Carthage College, USA

8. *Breadth, Depth, and Passion of 'DotStar': Transforming CS Students at Illinois*
   Mark Miller, Thomas Reese, Abhishek Modi, and Lawrence Angrave
   University of Illinois at Urbana Champaign, USA

9. *Building Strong Foundations: Computer Science in Elementary School*
   Michael Preston
   CSNYC, USA

10. *Gender Differences in High School Students' Decisions to Study Computer Science and Related Fields*
    Jason Ravitz
    Google, USA

11. *How to Train Your Robot: Computing Introduction for 8-10 Year Olds*
    Chad Williams, Emtethal Alafghani, Antony Daley Jr., Kevin Gregory, and Marianella Rydzewski
    Central Connecticut State University, USA

Peer-Led Team Learning Supports Computer Science Persistence at Community College

Kathryn I. Cunningham
California State University Monterey Bay
kcunningham@csumb.edu

Abstract:
After 6 years, only 39.1% of community college students have earned any degree—Associate's, Bachelor's, or certificate [1]. In an attempt to increase social and academic engagement, and consequently persistence, Hartnell Community College implemented Peer-Led Team Learning workshops [2] for its CS1 course in Spring 2015. The weekly workshops were not required, but students had the opportunity to earn significant course credit (4% of final grade) for consistent attendance. Workshops consisted of two hours of small-group problem solving by students, guided by a more advanced student, and were advertised to students as a better alternative to traditional drop-in tutoring.

PLTL attendance was associated with a higher likelihood of persistence in the course and an increased desire to pursue computer science. 61.1% (22/36) of students completed this offering of CS1, higher than the typical approximately 50% completion rate for CS1 at Hartnell. Of students who completed, 95.5% (21/22) attended 1 or more PLTL workshops, vs. 28.6% (4/14) of students who dropped. Of those who attended 12 or more workshops (the amount needed to gain credit), 91.7% (11/12) indicated they were more likely to pursue a major in computer science than at the beginning of the semester in a post-semester survey. For students who attended 1-11 workshops, 66.7% (6/9) indicated they were more likely to pursue computer science.

PLTL workshops were highly rated by students in end-of semester surveys, with 42.9% of PLTL attendees categorizing PLTL as “extremely helpful” and 38.1% as “Very helpful”. 95% wanted PLTL in their next computer science course. However, no correlation was found between number of PLTL attendances and final course grade.

[1] Shapiro, Doug, et al. "Completing College: A National View of Student Attainment Rates. Signature [TM] Report 4." *National Student Clearinghouse* (2012).
[2] Gosser, David K., et al. "Peer-Led Team Learning: A Guidebook." (2001).

Educating the Next Generation of Great Software Engineers

Leigh Ann DeLyser
NYC Foundation for CS Education
leighann@csnyc.org

Abstract:
Career and technical education (CTE) programs at the K-12 school level provide students with the opportunity to study standard academic courses, as well as specialize in career themed programs of study. In New York City, the Academy for Software Engineering (AFSE) is a CTE school focused on software engineering. As the first of its kind, AFSE has had to construct a program that encompasses academic coursework, rigorous standards with an end-of-program technical assessment, and work based learning, where students engage with industry professionals.

In the fall of 2015, AFSE will submit its program for NY State approval. Over the past three years the program has been developed, classroom tested, and refined to meet the needs of NYC public school students and, at the same time, expectations of partner companies who hire software engineers. Companies such as Google, Morgan Stanley, JP Morgan, Facebook, and others have all participated in the design and evaluation of the program.

To connect the program of study to the ICER community, the presentation will highlight similarities between the three parts of the AFSE program and previous works focusing on the necessary content in software engineering education [1], the tasks performed by novice software engineers [2], and desired qualities of expert software engineers [3]. It is the hope of the author that conversation with the ICER community can influence final program adjustments before the CTE program is submitted for state approval.

[1] Tucker, A. (2003). A Model Curriculum for K--12 Computer Science: Final Report of the ACM K--12 Task Force Curriculum Committee.

[2] Begel, A., & Simon, B. (2008). Novice software developers, all over again. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 3-14). ACM.

[3] Li, P., Ko, A.J., and Zhu, J. (2015) What Makes a Great Software Engineer?, International Conference on Software Engineering. IEEE

The MoveLab: Supporting Diversity through Self-Conceptions

Kayla DesPortes
Georgia Institute of Technology
ksdesportes@gatech.edu

Abstract:
Individuals within underrepresented populations in computing often do not perceive the identity of a *computer scientist* as aligning with their interests or value system. In many cases this leads to rejection of opportunities to participate within the discipline [1–4]. To engage a diversity of individuals in computing, we need to better understand how to support students in forming positive and meaningful identities with computing. This study centers on a community of African American and Latina teenage girls as they began to form self-conceptions with computing and dance during the MoveLab. The MoveLab was a five-day STEAM (Science, Technology, Engineering, Arts, and Mathematics) workshop where students, dancers, choreographers, computer scientists, and engineers collaborated to create a technology enhanced dance performance. Throughout the workshop, we observed students form and reject self-conceptions [5] around computing. We found that supporting diverse roles of participation, creating a trusting and engaging social environment, having leaders model behaviors, and uncovering preconceived ideas were important characteristics for supporting self-conception formation around computing and dance.

[1] James DiSalvo, B. et al. 2011. African American men constructing computing identity. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2011), 2967–2970.

[2] Katz, S. et al. 2003. Gender and race in predicting achievement in computer science. *Technology and Society Magazine, IEEE*. 22, 3 (2003), 20–27.

[3] Margolis, J. 2008. *Stuck in the shallow end: Education, race, and computing*. MIT Press.

[4] Margolis, J. and Fisher, A. 2003. *Unlocking the clubhouse: Women in computing*. MIT Press.

[5] Steinberg, L. and Morris, A.S. 2001. Adolescent Development. *Journal of Cognitive Education and Psychology*. 2, 1 (2001), 55–87.

Computer Science is "Hard": Looking at the Gender Gap Between Two Computing Programs

Brittany Ann Kos
ATLAS Institute
University of Colorado Boulder
Brittany.Kos@colorado.edu

Abstract:
We will be comparing two computing programs at the University of Colorado Boulder, a multidisciplinary program in Technology, Arts, and Media (TAM) and a traditional Computer Science program. Both of these programs have large enrollments (800 and 1100 students in 2014-2015 for TAM and CS, respectively), but different demographics. The TAM program has over 60% women enrolled; while CS only enrolls 25% women. In an ongoing study, we are investigating the introductory classes for these programs, in an effort to gain insight into why these they attract and retain such distinct populations.

We present a preliminary analysis that scrutinizes the way material is presented in class. Through ethnographic observation, we have studied classroom environments and discourse, and discovered a large discrepancy in the way students, professors, and teaching assistants refer to the course material.

The CS classes acknowledge the difficulty of the material being covered and have normalized this type of discourse in the classroom. These classes have common phrases, such as "This is difficult" or "This isn't intuitive", which are usually referring to CS in general or the class overall. Alternatively, the TAM instructors almost never indicate how difficult the material might be, but instead focus on how hard the students will have to work to succeed. When students talk about their hardships, they refer to specific lessons or projects they are doing, not computing in general. This is the first phase in a larger study, investigating the effect this discourse may, or may, not have on student enrollment.

Middle School Students Learn Programming Using Wearable Technologies

Dr. Victoria Lentfer
University of Nebraska Omaha
vlentfer@unomaha.edu

Abstract:
The University of Nebraska - Lincoln and the University of Nebraska at Omaha are partnering with the Nebraska Department of Education to develop, evaluate, and deliver the adoption of a project-based curriculum for middle school formal and informal educators in the content areas of computer programming and engineering. This project utilizes wearable technologies as an instructional context by using microprocessors to introduce the basics of engineering and computing using the Arduino platform. Wearable technology is combined with smart textiles and computer circuitry so that the students can immerse themselves in this hands-on project-based curriculum. This project makes a significant contribution to computer science education by establishing the effectiveness of project-based learning in the study of integrating STEM disciplines of computer programming, engineer design, and electric circuitry. The project has the potential for national adoption, for a direct impact on computer science educational practices, reaching traditional underserved populations and for advancing knowledge in the computer science discipline.

[1] Eric Mazur. 2011. The scientific approach to teaching: research as a basis for course design. In *Proceedings of the seventh international workshop on Computing education research (ICER '11)*. ACM, New York, NY, USA, 1-2.

CSTeachingTips.org: Connecting Research and Practice

Colleen Lewis
Harvey Mudd College
lewis@cs.hmc.edu

Abstract:
This poster will:

- Inform attendees about the CSTeachingTips.org project and our ongoing research,
- Encourage discussion about the interaction between research and practice within our community, and
- Request that attendees share (on the accompanying interactive poster) something from their research that they wish could reach teachers.

CSTeachingTips.org (NSF# 1339404) is a project to document and disseminate tips about teaching CS in an attempt to provide the resources necessary to train a growing field of CS educators. We have currently gathered and posted over 1000 tips about teaching CS at CSTeachingTips.org. Colleen will share insights from the project and some early research regarding gaps between expert and novice CS teachers' perceptions of novice CS teachers' needs.

As a community of CS education researchers, we should be continually discussing and evaluating the structures within the community to bridge research into practice. This lightning talk and poster attempts to spark this discussion by having attendees contribute specific insights from their research that are relevant to CS teachers and posing questions about the current connections between CS education research and practice.

Storyteller: Telling Stories by Reflecting on Code Evolution

Mark Mahoney
Carthage College
mmahoney@carthage.edu

Abstract:
Most significant pieces of software evolve from one small idea into a combination of more ideas, bug fixes, and optimizations. The growth from small to large and simple to complex is a journey. These journeys often have interesting stories that go along with them.

Code comments are usually not an ideal place for telling these stories because some important context may be missing from the final state of the code. It doesn't make sense to document in the comments why an important code refactor was made if there is no evidence left of it in the code itself. These changes may have some valuable lessons in them but there is usually not a good place to write them down for others to learn from.

This lightning talk will describe a tool called Storyteller. Storyteller captures the keystrokes and file operations during multiple development sessions and allows a student to play them back in an animated recreation of the code. The student can then add a narrative to the playback of code over time rather than in the code comments. These narratives, or *stories,* are valuable for students and instructors to explain code evolution over time.

The stories that can be created from the tool can:

- allow a student to reflect on how their code evolved to look for key mistakes and good decisions.
- help a student explain the evolution to a grading instructor. This will help the instructor understand the final product and get a sense of why the code is the way it is.
- help a grading instructor provide feedback by allowing them to create their own stories to highlight the good, the bad, and the ugly.
- help students learn from their peers by watching how others write their code and hearing their stories.

Breadth, Depth, and Passion of "DotStar": Transforming CS Students at Illinois

Mark Miller, Thomas Reese, Abhishek Modi, Lawrence Angrave
University of Illinois at Urbana Champaign, USA
{mrmillr3, reese6, akmodi2, angrave}@illinois.edu

Abstract:
We present "DotStar" – a non-traditional project-based learning model for highly-motivated first-semester Computer Science majors. We have deployed this model as part of a large enrolment class (N>100) over two 16-week semesters at UIUC. Students optionally enroll in DotStar alongside their required introductory CS courses.

DotStar has two main facets. The first, Projects, is organized in a high-energy, incubator-style atmosphere. Students pitch project ideas around which they form teams of three to twelve students. Teams present their progress on a three-week rotation and receive mentorship and management from an experienced CS student. We encourage ambitious and complex ideas for their projects. Previous projects include quadcopters that create 3D models, smartwatch RPGs, and traffic simulations for autonomous vehicles. The complex nature of the projects encourages students to not only gain depth and experience in relevant subfields and technologies but also encourages student engagement [1].

The second, microLessons, is a series of twenty-minute introductions to subfields and technologies. The goal is to provide students with a breadth of knowledge across CS. This helps students judge which concentrations they find useful or interesting, and informs them on how to continue learning them. Addressing unknown unknowns is key to this process. We start the semester with an introduction to programming, then move on to technologies such as iOS and SQL that help students with projects and later introduce subfields such as AI and Security. By the end of the semester students are far more technically informed than other CS freshmen.

Students and faculty alike have received this model positively and students are excited to continue their CS education. Some have even earned internships and research positions as a result of DotStar.

[1] Motivating project-tbased learning: Sustaining the doing, supporting the learning. PC Blumenfeld, E Soloway, RW Marx, JS Krajcik, M Guzdial, A Palincsar. Educational Psychologist 26 (3-4), 369-398.

Building Strong Foundations:
Computer Science in Elementary School

Michael Preston
CSNYC
michael@csnyc.org

Abstract:
The elementary grades offer a formative opportunity to introduce children to technology as tools for creative expression and problem solving. In elementary classrooms, computer science (CS) presents a powerful opportunity for interdisciplinary study and for the development of a range of computational thinking and problem solving skills that will serve as a foundation for later CS learning. Early engagement with CS not only empowers students to be critical users of technology but also demonstrates that learning can be creative, collaborative, and fun.

At present, only a small number of elementary schools offer CS instruction, both in NYC and across the country. Addressing equity issues, as well as longer-term educational and career persistence in technology, requires deeper investments in broader and earlier access to CS. [1] This poster will provide an overview some of the current elementary CS programs in NYC public schools, including "unplugged" activities, programming with block-based languages, building and programming robots, and working with electronics and other forms of physical computing.

CSNYC (the New York City Foundation for Computer Science Education) is a nonprofit dedicated to expanded access to CS in NYC public schools. The purpose of this lightning talk will be to prompt the community to consider the current state of elementary school CS implementation and begin a discussion of how to build out the portfolio of CS programs to reach significantly more students, while maintaining rigor, developmental appropriateness, and fun.

[1] Reporting on Our Efforts to Increase Diversity in Computer Science;
http://codeorg.tumblr.com/post/98856300118/diversity, 2014.

Gender Differences in High School Students' Decisions to Study Computer Science and Related Fields

Jason Ravitz
Google
ravitz@google.com

Abstract:
Computer science is vital to our future -- from the health of nations to an individual's ability to actively engage in the technology that is now embedded in nearly everything we do. Increasing women's participation in computer science (CS) is a critical workforce and equity concern [1]. Researchers, practitioners, and the technology industry have committed to reversing negative trends for women in CS as well as in engineering and information technology "computing" fields [2]. Building on prior research, this study identifies factors that influence young women's decisions to pursue CS-related degrees and the ways in which these factors differ for young men. We conducted a study of 1,739 students across the U.S., surveying high school students and college grads. Using factor analysis to group similar variables into broader factors and logit regression to rate importance of factors, we found that, for high school girls, 95% of the decision to pursue computer science or related fields is comprised of factors that can be influenced: social encouragement, career perception, academic exposure, and self-perception. Encouragement and exposure were particularly huge for girls compared to boys -- with encouragement from parents and other adults significantly more influential. And, for boys, career perception was a larger influencing factor. Self-perception was also important, with no significant difference between both genders. Overall, the most heartening outcome of the study is the limited role that uncontrollable factors play in influencing the pursuit of a CS degree. For high school girls, uncontrollable factors like household income and ethnicity contribute only 4.9% to the explainable factors. The factors most related to female participation in computing fields are actionable. Our presentation will share results of ongoing work to better understand the differences in factors that influence men and women in order to help achieve gender parity.

[1] National Science Foundation. (2012). *Science and Engineering Indicators 2012*. Washington, DC: Author. Retrieved from http://www.nsf.gov/statistics/seind12/c0/c0i.htm

[2] National Center for Women in Information Technology (2015). *Top 10 Ways Families Can Encourage Girls' Interest in Computing*. Boulder, CO: Author. Retrieved from https://www.ncwit.org/resources/top-10-ways-families-can-encourage-girls-interest-computing/top-10-ways-families-can.

How to Train Your Robot:
Computing Introduction for 8-10 Year Olds

Presenter: Chad Williams
Coauthors: Emtethal Alafghani, Antony Daley Jr., Kevin Gregory, and Marianella Rydzewski
Central Connecticut State University
cwilliams@ccsu.edu

Abstract:
Introducing technology to young students has become a critical objective of education programs in today's world. Educators and parents alike are seeking innovative ways to introduce young students to the important skill of computer programming. Within this study, a web application was developed to introduce young students to programming concepts in a fun way through problem solving. By acting as robot trainers, students are introduced to concepts such as: automation, conditions, repetition, and debugging.

The goals of the study were twofold: first, introduce students to how computers work in an enjoyable way and, second examine how they used the introduced programming concepts to solve problems. Two sessions were conducted with eight to ten year olds to evaluate the application. The first session introduced the students to programming concepts in a structured, incremental fashion. The second session challenged them to apply what they learned to create more free form solutions. In addition to surveying participants to understand their knowledge and motivations before and after playing the game, the team also collected analytical data during game play.

This study reports on the trends that were observed in: the programs students developed, debugger use, and variation in the number of attempts and completions between the two sessions. Our results showed the game effectively familiarized eight to ten year old students with both computer language constructs and the essentials of algorithmic thinking, while also increasing the students' own perceived knowledge of computers in a way they enjoyed. Students were quickly able to learn the core programming concepts introduced by the game and apply these concepts to free form solutions. The results demonstrated both males and females enjoyed the game and developed an enthusiastic interest in learning more about computer programming.