2016

Creative Computation for CS1 and K9-12

Dianna Xu
Bryn Mawr College, dxu@brynmawr.edu

Ira Greenberg

Deepak Kumar
Bryn Mawr College, dkumar@brynmawr.edu

Ursula Wolz

Custom Citation

Xu, D. et al. 2016. Creative Computation for CS1 and K9-12 Envisioning the Future of Undergraduate STEM Education: Research and Practice Symposium (AAAS EnFuse), April 27-29, 2016, Washington, DC.

Creative Computation for CS1 and K9-12

Dianna Xu
Computer Science
Bryn Mawr College
Bryn Mawr, PA
dxu@brynmawr.edu

Ira Greenberg
Computer Science & Engineering
Southern Methodist University
Dallas, TX
igreenberg@smu.edu

Deepak Kumar
Computer Science
Bryn Mawr College
Bryn Mawr, PA
dkumar@brynmawr.edu

Ursula Wolz
RiverSound Solutions
Montclair, NJ
http://riversoundsolutions.com
riversoundsol@gmail.com

ABSTRACT

We present the design and development of a new approach to teaching the introductory computing course (CS1), at both the college-level as well as K9-12, using the context of digital art and creative computation. Creative computation is a highly interdisciplinary area combining theory and methodology from computer science and engineering with aesthetic principles and creative practices from the arts. Using the Processing programming language, students create a portfolio of aesthetic visual designs that employ basic programming constructs and structures typically taught in traditional CS1 courses. The goal of this approach is to bring the excitement, creativity, and innovation fostered by the context of creative coding. We have developed a web portal containing an extensive set of resources for adoption by others. A comprehensive textbook has also been published in 2013 [Greenberg *et al* 2013].

We present results from a comparative study involving multiple offerings of the new course at the two lead institutions as well as several other partner institutions. We also describe the success of bringing creative computation via Processing into two very different high schools that span the range of possibilities of grades 9-12 in American education. We report on how contextualized computing that supports integration of media arts, design, and computer science can successfully motivate students to learn foundations of programming and come back for more. The work of two high school teachers with divergent pedagogical styles is presented. They successfully adapted a college-level creative computation curriculum to their individual school cultures providing a catalyst for significant increases in enrollment and female participation in high school computer science.

1. INTRODUCTION

Contextualized introduction to computing provides a means through which a diverse student population gains insight into the foundations of computer science [Cassel & Wolz 2013]. At the college level this includes media computation [Guzdial 2004], robots [Summet *et al* 2009, Kumar *et al* 2008], games/animation [Xu *et al* 2008, Wolz *et al* 2007, Bayless & Stout 2006], and music [Beck *et al* 2011]. Extensions into the K-12 curriculum are beginning to occur, but the primary emphasis is on computing principles applied to a sampling of disciplines rather than immersion in an interdisciplinary domain.

There have been various successful attempts at incorporating graphics and creativity in introductory computing courses (CS1); most notable examples include media computation [Guzdial 2004] and Alice [Cooper *et al* 2003, Moskal *et al* 2004]. The context that distinguishes our project is *creative computation* -- a highly interdisciplinary area combining theory and methodology from computer science and engineering with aesthetic principles and creative practices from the fine arts [Maida 2001]. The ultimate goal is to radically *recontextualize* computer code – from an applied math notation to a creative medium, on par with charcoal, paint, clay, etc. Creative coding is an exploratory and aesthetically driven approach, where students build visual designs and artworks iteratively as they expand their programs. In addition, the driving principle of the project is to create a curriculum that does not require a change of programming language away from Java or any modification of core CS1 topics covered in a traditional CS1 classroom.

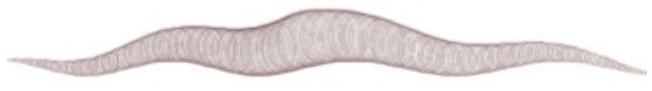
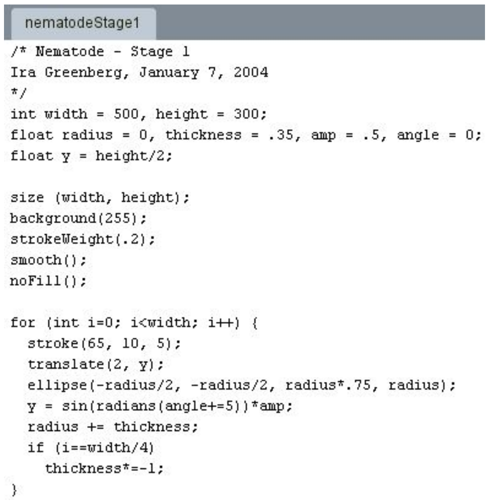


Figure 1: Nematode sketch (see Figure 2 for program listing)

We have chosen to work with the programming language Processing, a robust and full-featured language built on top of Java, that uses a simplified syntax and a graphics programming model. Compared to Java, it has a very flat learning curve with a simple, intuitive, and easy-to-use IDE – an excellent choice for novice programmers. It is fully integrated in that straight Java code can be embedded freely in any Processing program, and every Processing program can be exported to a Java applet as well as a Java application for Linux, Mac and Windows, and mobile devices. Because Processing is fully Java-based, transition to full Java from Processing takes 1-2 weeks, and happens relatively seamlessly either at the end of CS1 or beginning of CS2. This is an important feature for K-12 students aiming for Advance Placement (AP) in computer science for college-level credit, or college departments who teach Java-based CS2s. Processing was designed for the construction of 2D and 3D visual forms. Its IDE is well-suited for the kind of rapid proto-typing needed for dynamic visual work. Novice programmers respond well to programming environments where small snippets of code can be quickly tested and minimal effort is needed to run code. Despite the ease with which beginners take to Processing, it is a full featured programming language capable of rendering stunning graphics and animations, ones that rival the capabilities of OpenGL and other standard Graphics libraries, with little learning curve and much less actual code. For example, Figure 2 shows a complete Processing program [Greenberg 2007] that generated the sketch in Figure 1.



```
/* Nematode - Stage 1
Ira Greenberg, January 7, 2004
*/
int width = 500, height = 300;
float radius = 0, thickness = .35, amp = .5, angle = 0;
float y = height/2;

size (width, height);
background(255);
strokeWeight(.2);
smooth();
noFill();

for (int i=0; i<width; i++) {
  stroke(65, 10, 5);
  translate(2, y);
  ellipse(-radius/2, -radius/2, radius*.75, radius);
  y = sin(radians(angle+=5))*amp;
  radius += thickness;
  if (i==width/4)
    thickness*=-1;
}
```

Figure 2: Nematode program in Processing

2. CREATIVE COMPUTATION FOR CS1

Our Introduction to Computing (CS1) course based on creative computation using Processing was developed and evolved over the past six years as NSF TUES Type 1 & 2 projects (NSF DUE-0942626, NSF DUE-0942628, DUE-1323463 and DUE-1323305 Collaborative: Bryn Mawr College, Bryn Mawr, PA and Southern Methodist University, Dallas, TX). Over the course of these projects, our CS1 course has been offered over twenty times by a dozen different faculty. Classes at both institutions created overwhelmingly positive enrollment and retention trends, as well as strong student interest and motivation. Due to the successful results and its popularity this course has now been integrated into the sustainable regular offerings at the two institutions. Results and assessment data were reported in SIGCSE 2012 [Greenberg *et al* 2012] and SIGCSE 2016 [Xu *et al* 2016].

The general philosophy of our design is to teach the core CS1 topics as we would in an average traditional Java-based CS1. The only difference is that we show the application of these principles with creative computation. In other words, we teach the same materials, but instead of solving for roots of polynomials or simulating gas station/cash registers, we create graphics, interactive media and visualizations and introduce students to contemporary, diverse examples of computing in a modern context. Besides standard programming constructs, data structures and algorithms, we also introduce students to data visualization and other advanced areas not typically accessible in CS1.

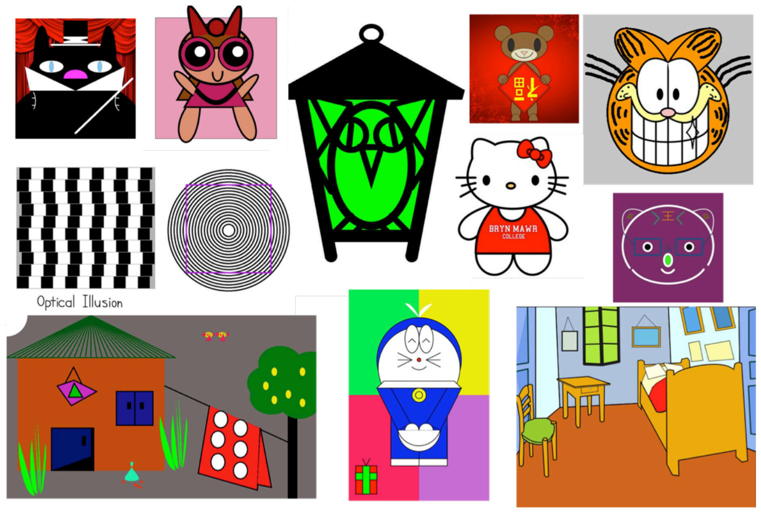


Figure 3: Student submissions for assignment 1, after 1 week of lecture

Since we started offering this CS1 course in Fall 2010, Bryn Mawr's pre-registration numbers consistently exceeded available seats by 100% and lotteries are always needed to keep out half of the students who pre-register. Bryn Mawr is not able to offer more sections due to staffing constraints. All Bryn Mawr Processing classes have 100% or near 100% retention rates. Currently, Bryn Mawr is offering 3-5 sections of this course each year enrolling nearly 10% of our entire undergraduate student body.

SMU also experienced high demand in their Processing-based CS1 sections, including exceeding enrollment caps. In addition, the success of the Processing-based CS1 has led SMU to start offering an entire curriculum in Creative Computing[1], including establishing a new major and minor. The Creative Computing major is the first program in the country that demands rigor and concentration in both engineering and the arts with an emphasis on Computer Science. A new graduate program in Creative Computing is also being developed.

Identical surveys were collected from pilot classes at both Bryn Mawr College and SMU, which are very different institutions. Bryn Mawr is a small all-women's liberal arts college with 1,300 undergraduate students while SMU offers a more conventional CS and Engineering program with a much larger student body (11,000 total with 6,000 undergraduates). SMU also offers an Introduction to CS for Non-majors (CS0) while Bryn Mawr does not, making SMU's CS1 less likely to attract non-majors, particularly non-STEM majors. Bryn Mawr's CS1 tends to be 90% non-majors or undecided. In addition, near identical surveys were also given to traditional Java-based CS1 sections running concurrently at SMU as a control, with two art and/or Processing specific questions taken out. Major observations from the data are the following:

1. Students in the Processing sections are more positively inclined to take additional CS courses.
2. Students in the Processing sections are much more likely to spend extra time on a homework assignment for "fun".

[1] http://www.smu.edu/Meadows/AreasOfStudy/CreativeComputation

3. A large percentage of the Bryn Mawr women (62%) did not expect that they would ever have to write another program in any language after this class (likely non-STEM majors), yet the same Bryn Mawr students also showed the most interest in creating self-initiated programming projects outside of class. We believe this is a particularly encouraging sign that the art and creative coding context motivates women and attracts non-STEM or undecided majors to computer science.

20% of Bryn Mawr's all-female freshmen class now takes CS1 (lottery still required), while SMU reports 41% female in Processing-based CS1 classes and 50% women majors in its new Creative Computation program. A dozen different instructors have successfully offered our curriculum more than twenty times. In addition, we have led several national faculty training workshops annually since the start of the projects and supported the adoption of our course at many institutions.

We have developed extensive course materials to support widespread use and adoption of creative computing for teaching introductory computing courses. These include a rich collection of course exemplars and a gallery of examples on our web portal (www.cs.brynmawr.edu/visual). We have published a comprehensive textbook [Greenberg *et al* 2013] for in these courses. In addition, for those wishing to do formal studies in assessing their own adoptions, we provide our custom-designed pre- and post- surveys. In the next section, we summarize our progress on implementing our approach in colleges and high schools across the United States.

3. BEYOND THE LEAD INSTITUTIONS

The curriculum outlined above has been in place within undergraduate introductory courses at Bryn Mawr College and Southern Methodist University for over six years. The two lead institutions have experienced a significant increase in introductory computing enrollments prior to the more recent national surge. These surges are directly attributable to the creative computation approach (as reported in [Greenberg *et al* 2012, Greenberg *et al* 2013]). In fall 2013, a TUES Type 2 project was awarded to focus on nation-wide adoption and expansion into K9-12.

During this project, we have partnered with nine institutions whose CS faculty are adopting our curriculum to their local classrooms and conducting formal assessment and data collection during the pilot course offerings. The partners include: one two-year community college, two four-year liberal arts colleges, two 4-year large public universities and four high schools. Only one partner has completed the course adoption and assessment so far while the remaining are offering the course now or in spring 2016. More formal analysis of a larger set of institutions is underway.

We recruited two lead high schools with experienced CS teachers to take on the task of adapting the college-level creative computation curriculum into high school classrooms. The two high school partners were chosen as the initial lead participants both because of their school cultures and for the marked difference in teacher pedagogical styles. James Martin High School (MHS), located in the Arlington School District in Dallas, TX, is a large (> 3,000 student) public high school (grades 9-12) with rigorous standardization at the state level. Sidwell Friends School (SFS), located in Washington, DC is an east-coast elite private PK-12 day school with a total enrollment of approximately 1,000 and a deep Quaker tradition. The two teachers in both schools emphasize extensive student-initiated problem solving. Aaron Cadle (MHS) provides a highly structured, more traditional delivery of instruction style using a huge corpus of materials he has developed; Darby Thompson (SFS) approaches the material through quick face-to-face delivery followed by extensive one-on-one and small group interaction.

A new set of high school curricular materials has since been developed and taught at both MHS and SFS, leading to similar reports of increased enrollments and student motivation at both high schools. In addition, significant increase in female participation is reported from SFS and sustained high female participation rate in pre-AP and increased female participation in AP and beyond are reported from MHS during the project's first two years.

These results, despite school differences, suggest that creative computation can be adapted across a broad spectrum of school cultures by teachers with a range of organizational and accountability constraints and levels of expertise. Detailed experience and success of these teachers are reported in our SIGCSE 2016 paper [Xu *et al* 2016]. More formal analysis of a larger set of schools is also forthcoming.

4. SUMMARY

The Creative computation approach suggests that a broader cohort of students can be engaged in computer science, and appears effective in attracting women and non-STEM majors to computer science. Our experience at both the college and high school suggests that the immediate feedback provided by graphic visualizations supports student competencies in the essential skills of programming, as well as a deeper understanding of foundational computer science concepts. Lessons learned include (1) the importance of fostering individual creativity in solving a problem, as demonstrated first at Bryn Mawr and SMU, (2) in classroom coaching as demonstrated at Sidwell Friends, (3) extensive material support to allow students to pursue highly individualized solutions, as demonstrated at Martin High School. Creative computation curricula and outcomes demonstrate that introductory computing can be highly motivating because it presents interesting problems to students that tap into their personal interests and creative expertise.

5. ACKNOWLEDGMENTS

This project is supported in part by grants from The National Science Foundation (DUE-0942626, DUE-0942628, DUE-1323463, DUE-1323305, CCF-0939370, and CCF-1140489), and from Bryn Mawr College and Southern Methodist University.

6. REFERENCES

[1] [Bayless & Stout 2006] Jessica D. Bayless and Sean Strout. Games as a "Flavor" of CS1. In *proceedings of SIGCSE 2006*. ACM Press 2006.

[2] [Beck *et al* 2011] Robert E. Beck, Jennifer Burg, Jesse M. Heines, and Bill Manaris. Computing and Music: A Spectrum of Sound. *Special Session, SIGCSE 2011*. Dallas, TX, March 2011.

[3] [Cassel & Wolz 2013] Cassel, L. and Wolz, U Interdisciplinary Computing, Successes and Challenges. In *Proceedings of SIGCSE 2013*. ACM Press 2013.

[4] [Cooper *et al* 2003]. Stephen Cooper, Wanda Dann, Randy Pausch. Teaching Objects-first in Introductory Computer Science. In Proceedings of SIGCSE 2003. ACM Press 2003

[5] [Greenberg 2007] Ira Greenberg. *Processing: Creative Coding and Computational Art*. Friends of Ed, 2007.

[6] [Greenberg *et al* 2012] Ira Greenberg, Deepak Kumar and Dianna Xu. Creative Coding and Visual Portfolio for CS1. In *Proceedings of SIGCSE 2012*. ACM Press 2012.

[7] [Greenberg *et al* 2013] Ira Greenberg, Dianna Xu, and Deepak Kumar. *Creative Coding and Generative Art in Processing 2.0*. friends Of ed/Apress 2013.

[8] [Guzdial 2004] Mark Guzdial. *Introduction to computing and programming with Python: A Multimedia Approach*. Prentice-Hall, 2004.

[9] [Kumar *et al* 2008] Deepak Kumar, Doug Blank, Tucker Balch, Keith O'Hara, Mark Guzdial, Stewart Tansley, Engaging Computing Students with AI and Robotics. *Symposium on Using AI to Motivate Greater Participation in Computer Science*, 2008.

[10] [Maida 2001] John Maeda *Design by Numbers*, MIT Press 2001.

[11] [Moskal *et al* 2004] Barb Moskal, Deborah Lurie, Stephen Cooper. Evaluating the Effectiveness of a New Instructional Approach. In Proceedings of SIGCSE 2004. ACM Press, 2004

[12] [Summet *et al* 2009] Jay Summet, Deepak Kumar, Keith O'Hara, Daniel Walker, Lijun Ni, Doug Blank, and Tucker Balch. Personalizing CS1 with Robots. In *Proceedings of ACM SIGCSE 2009*. ACM Press 2009.

[13] [Wolz *et al* 2007] Wolz, U., C. Ault and T. M. Nakra, "Teaching Game Design through Cross-Disciplinary Content and Individualized Student Deliverables", *The Journal of Game Development*, adapted based on invitation from presentation at the 2nd Annual Microsoft Academic Days Conference on Game Development , February 22 - 25, 2007

[14] [Wolz *et al* 2011] Wolz, U., K. Pearson, S.M. Pulimood, M. Stone, M. Switzer. Computational Thinking and Expository Writing in the Middle School: A novel approach to broadening participation in computing, *Transactions on Computing Education*, 2011, Volume 2, article 9.

[15] [Xu *et al* 2008] Dianna Xu, Douglas Blank, and Deepak Kumar. Games, Robots and Robot Games: Complementary Contexts for Introductory Computing Education. In *Proceedings of Third International Conference on Game Development in Computer Science Education* (GDCSE'08), 2008.

[16] [Xu *et al* 2016] Dianna Xu, Aaron Cadle, Darby Thompson, Ursula Wolz, Deepak Kumar, and Ira Greenberg. Creative Computation in High School. In *Proceedings of SIGCSE 2016*. ACM Press 2016. To appear.