

# The Journal of Computing Sciences in Colleges

**Papers of the 26th Annual CCSC  
Central Plains Conference**

April 3-4, 2020  
St. Charles Community College  
Cottleville, MO

Baochuan Lu, Editor  
Southwest Baptist University

Bin "Crystal" Peng, Regional Editor  
Park University

**Volume 35, Number 6**

**April 2020**

*The Journal of Computing Sciences in Colleges* (ISSN 1937-4771 print, 1937-4763 digital) is published at least six times per year and constitutes the refereed papers of regional conferences sponsored by the Consortium for Computing Sciences in Colleges.

Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

## Table of Contents

<b>The Consortium for Computing Sciences in Colleges Board of Directors</b>	<b>5</b>
<b>CCSC National Partners</b>	<b>7</b>
<b>Welcome to the 2020 CCSC Central Plains Conference</b>	<b>8</b>
<b>Regional Committees — 2020 CCSC Central Plains Region</b>	<b>9</b>
<b>Reviewers — 2020 CCSC Central Plains Conference</b>	<b>11</b>
<b>Invited Talks</b>	<b>12</b>
<i>Andy Nelson, Paul Barham, Frank Alaniz</i>	
<b>Using a Short Textbook in CS 1 to Improve Student Reading</b>	<b>13</b>
<i>David Toth, Thomas Allen, Centre College</i>	
<b>An Alternative to Programming Contests</b>	<b>22</b>
<i>Timothy Urness, Drake University</i>	
<b>Design of Virtual Labs for an Ethical Hacking Course</b>	<b>31</b>
<i>Xiaodong Yue, Hyungbae Park, University of Central Missouri</i>	
<b>Discrete Math: To Blend or Not Blend</b>	<b>39</b>
<i>Charles Hoot, Northwest Missouri State University</i>	
<b>DIVAS at Three: Image Processing Outreach</b>	<b>46</b>
<i>Mark M. Meysenburg, Tessa Durham Brooks, Erin Doyle, Doane University, Raychelle Burks, St. Edward's University</i>	
<b>A Course-Based Undergraduate Research Experience (CURE) in Computer Science: An Experience Report</b>	<b>56</b>
<i>Fahmida Hamid, Grinnell College</i>	
<b>LAGradebook: A Tool for Course-Level Comparative Learning Analytics</b>	<b>66</b>
<i>Christopher Phillips, Jesse Eickholt, Central Michigan University</i>	

<b>Engaging Early Programming Students with Modern Assignments Using BRIDGES</b>	<b>74</b>
<i>Allie Beckman, Matthew Mcquaigue, Alec Goncharow, David Burlinson, Kalpathi Subramanian, Erik Saule, UNC Charlotte, Jamie Payton, Temple University</i>	
<b>Using JShell in CS1</b>	<b>84</b>
<i>Joseph Kendall-Morwick, Missouri Western State University</i>	
<b>Is It Getting Foggy in Here? Cloud Computing in the Classroom</b>	<b>92</b>
<i>Denise M. Case, Michael P. Rogers, Northwest Missouri State University</i>	
<b>Effect of User Involvement in Information Systems Capstone Course: A Case Study</b>	<b>107</b>
<i>Cindy Zhiling Tu, Joni Adkins, Northwest Missouri State University</i>	
<b>Introduction to Alexa Programming</b>	
— <b>Conference Tutorial/Workshop</b>	<b>117</b>
<i>Jay Canty, Edgar Cerna, Wen-Jung Hsin, Park University</i>	
<b>Real-World Data, Games and Visualizations in Early CS Courses Using BRIDGES — Conference Tutorial/Workshop</b>	<b>118</b>
<i>Kalpathi Subramanian, Erik Saule, UNC Charlotte, Jamie Payton, Temple University</i>	
<b>Short Modules for Introducing Heterogeneous Parallel Programming — Conference Tutorial/Workshop</b>	<b>119</b>
<i>David P. Bunde, Knox College</i>	
<b>Error Detection and Correction Using Hamming Code</b>	
— <b>Nifty Assignment</b>	<b>121</b>
<i>Rad Alrifai, Northeastern State University</i>	
<b>Drawing With A Turtle — Nifty Assignment</b>	<b>122</b>
<i>Saty Raghavachary, University of Southern California</i>	
<b>TwHeatmap: Visualizing Sentiment Analysis of Tweets</b>	
— <b>Nifty Assignment</b>	<b>125</b>
<i>Evelyn Brannock, Robert Lutz, Georgia Gwinnett College</i>	



## The Consortium for Computing Sciences in Colleges Board of Directors

Following is a listing of the contact information for the members of the Board of Directors and the Officers of the Consortium for Computing Sciences in Colleges (along with the years of expiration of their terms), as well as members serving CCSC:

**Jeff Lehman**, President (2020), (260)359-4209, jlehman@huntington.edu, Mathematics and Computer Science Department, Huntington University, 2303 College Avenue, Huntington, IN 46750.

**Karina Assiter**, Vice President (2020), (802)387-7112, karinaassiter@landmark.edu.

**Baochuan Lu**, Publications Chair (2021), (417)328-1676, blu@sbuniv.edu, Southwest Baptist University - Department of Computer and Information Sciences, 1600 University Ave., Bolivar, MO 65613.

**Brian Hare**, Treasurer (2020), (816)235-2362, hareb@umkc.edu, University of Missouri-Kansas City, School of Computing & Engineering, 450E Flarsheim Hall, 5110 Rockhill Rd., Kansas City MO 64110.

**Judy Mullins**, Central Plains Representative (2020), Associate Treasurer, (816)390-4386, mullinsj@umkc.edu, School of Computing and Engineering, 5110 Rockhill Road, 546 Flarsheim Hall, University of Missouri - Kansas City, Kansas City, MO 64110.

**John Wright**, Eastern Representative (2020), (814)641-3592, wrightj@juniata.edu, Juniata College, 1700 Moore Street, Brumbaugh Academic Center, Huntingdon, PA 16652.

**David R. Naugler**, Midsouth Representative (2022), (317) 456-2125, dnaugler@semo.edu, 5293 Green Hills Drive, Brownsburg IN 46112.

**Lawrence D'Antonio**, Northeastern Representative (2022), (201)684-7714, ldant@ramapo.edu, Computer Science Department, Ramapo College of New Jersey, Mahwah, NJ 07430.

**Cathy Bareiss**, Midwest Representative (2020), cbareiss@olivet.edu, Olivet Nazarene University, Bourbonnais, IL 60914.

**Brent Wilson**, Northwestern Representative (2021), (503)554-2722, bwilson@georgefox.edu, George Fox University, 414 N. Meridian St, Newberg, OR 97132.

**Mohamed Lotfy**, Rocky Mountain Representative (2022), Information Technology Department, College of Computer & Information Sciences, Regis University, Denver, CO 80221.

**Tina Johnson**, South Central Representative (2021), (940)397-6201, tina.johnson@mwsu.edu, Dept. of Computer Science, Midwestern State University, 3410 Taft Boulevard, Wichita Falls, TX 76308-2099.

**Kevin Treu**, Southeastern Representative (2021), (864)294-3220, kevin.treu@furman.edu, Furman University, Dept of Computer Science, Greenville, SC 29613.

**Bryan Dixon**, Southwestern Representative (2020), (530)898-4864, bcdixon@csuchico.edu, Computer Science Department, California State University, Chico, Chico, CA 95929-0410.

**Serving the CCSC:** These members are serving in positions as indicated:

**Brian Snider**, Membership Secretary, (503)554-2778, bsnider@georgefox.edu, George Fox University, 414 N. Meridian St, Newberg, OR 97132.

**Will Mitchell**, Associate Treasurer, (317)392-3038, willmitchell@acm.org, 1455 S. Greenview Ct, Shelbyville, IN 46176-9248.

**John Meinke**, Associate Editor,

meinkej@acm.org, UMUC Europe Ret, German Post: Werderstr 8, D-68723 Oftersheim, Germany, ph 011-49-6202-5777916.

**Shereen Khoja**, Comptroller, (503)352-2008, shereen@pacificu.edu, MSC 2615, Pacific University, Forest Grove, OR 97116.

**Elizabeth Adams**, National Partners Chair, adamses@jmu.edu, James Madison University, 11520 Lockhart Place, Silver Spring, MD 20902.

**Megan Thomas**, Membership System Administrator, (209)667-3584, mthomas@cs.csustan.edu, Dept. of Computer Science, CSU Stanislaus, One University Circle, Turlock, CA 95382.

**Deborah Hwang**, Webmaster, (812)488-2193, hwang@evansville.edu, Electrical Engr. & Computer Science, University of Evansville, 1800 Lincoln Ave., Evansville, IN 47722.

## CCSC National Partners

The Consortium is very happy to have the following as National Partners. If you have the opportunity please thank them for their support of computing in teaching institutions. As National Partners they are invited to participate in our regional conferences. Visit with their representatives there.

### Platinum Partner

*Turingscraft*

*Google for Education*

*GitHub*

*NSF – National Science Foundation*

### Silver Partners

*zyBooks*

### Bronze Partners

*National Center for Women and Information Technology*

*Teradata*

*Mercury Learning and Information*

*Mercy College*

## Welcome to the 2020 CCSC Central Plains Conference

2020 ushers in the 26th year of the CCSC Central Plains Region conference and we are proud that St. Charles Community College (SCC) has been selected to host. Every year that we have attended the conference we have taken back to the classrooms something new and exciting to help teach and energize out students. Whether it is finding a new tool or technology or partnering with one of our peers, the conference has something for everyone.

This year's conference at SCC we will have some exciting events on the schedule, specialty tracks for vendors, students and K-12 teachers to focus in on what interests you most.

All professional papers, panels, tutorials, and nifty assignments go through a double-blind review process. This year we were able to accept 53%. We would like to extend our appreciation to the authors who submitted their work for our consideration and to the highly talented group of reviewers that exerted a tremendous amount of time and effort to review all the submissions. Our conference would not be as remarkable without our National Partners, Sponsors, and Vendors for their continued support of our organizations. Thank you very much, all!

Finally, this conference would be nowhere near as successful without the dedication and support from the volunteers and committee members that help support the CCSC efforts throughout the year to bring you this wonderful event. Without their tireless work and dedication, we would not have been able to plan and build a conference this great. This includes the support from past conference chairs. Thank you all for letting us be "pests" throughout the year of planning.

We hope you have a wonderful time at the CCSC-2020 conference, gain some enlightenment and take back something new to enrich your classroom and research.

Deepika Jagmohan and Dayu Wang  
St. Charles Community College  
Conference Chair

## 2020 CCSC Central Plains Conference Steering Committee

### Conference Chair

Dayu Wang ..... St. Charles Community College

### Conference Co-Chair

Deepika Jagmohan ..... St. Charles Community College

### Conference Publicity

Tom Mertz ..... Kansas State Polytechnic

Michael P. Rogers ..... Northwest Missouri State University

### Keynote Speakers

Deepika Jagmohan ..... St. Charles Community College

Michael P. Rogers ..... Northwest Missouri State University

Scott Sigman ..... Drury University

### Pre-Conference Workshop

Judy Mullins ..... University of Missouri Kansas City

Michael P. Rogers ..... Northwest Missouri State University

Dayu Wang ..... St. Charles Community College

### Papers, Panels, Tutorials, Workshops

Scott Bell ..... Northwest Missouri State University

Ron McCleary ..... Retired

Mahmoud Yousef ..... University of Central Missouri

### Nifty Assignments

Mahmoud Yousef ..... University of Central Missouri

Scott Bell ..... Northwest Missouri State University

### Lightning Talks

Diana Linville ..... Northwest Missouri State University

### K-12 Outreach

Nicole Nunfaren ..... St. Charles Community College

Mahmoud Yousef ..... University of Central Missouri

Scott Bell ..... Northwest Missouri State University

Belinda Copus ..... University of Central Missouri

Wen Hsin ..... Park University

Michael P. Rogers ..... Northwest Missouri State University

### Student Paper Session

Scott Sigman ..... Drury University

Michael P. Rogers ..... Northwest Missouri State University

Ajay Bandi ..... Northwest Missouri State University

### Student Poster Competition

Joseph Kendall-Morwick ..... Missouri Western University

Ajay Bandi ..... Northwest Missouri State University

Tim DeClue .....	Southwest Baptist University
<b>Student Programming Contest</b>	
Charles Riedesel .....	University of Nebraska-Lincoln
Charles Hoot .....	Northwest Missouri State University
<b>Two-Year College Outreach</b>	
Rex McKanry .....	St. Charles Community College
Belinda Copus .....	University of Central Missouri
Wen Hsin .....	Park University
Mahmoud Yousef .....	University of Central Missouri
<b>Career Fair</b>	
Rex McKanry .....	St. Charles Community College
<b>Submission System</b>	
Scott Bell .....	Northwest Missouri State University
Rex McKanry .....	St. Charles Community College

## Regional Board — 2020 CCSC Central Plains Region

<b>Regional Rep &amp; Board Chair</b>	
Judy Mullins .....	University of Missouri Kansas City
<b>Registrar Membership Chair</b>	
Ron McCleary .....	Retired
<b>Current Conference Chair</b>	
Dayu Wang .....	St. Charles Community College
Deepika Jagmohan .....	St. Charles Community College
<b>Next Conference Chair</b>	
Brian Hare .....	University of Missouri Kansas City
<b>Past Conference Chair</b>	
Rex McKanry .....	St. Charles Community College
<b>Secretary</b>	
Diana Linville .....	Northwest Missouri State University
<b>Regional Treasurer</b>	
Denise Case .....	Northwest Missouri State University
<b>Regional Editor</b>	
Bin “Crystal” Peng .....	Park University
<b>Webmaster</b>	
Michael P. Rogers .....	Northwest Missouri State University

## Reviewers — 2020 CCSC Central Plains Conference

Scott Sigman	Drury University, Springfield, MO
Brian Hare	University of Missouri-Kansas City, Kansas City, MO
Wen Hsin	Park University, Parkville MO
Michael Rogers	Northwest Missouri State University, Maryville, MO
Ron McCleary	(retired) Independence, MO
Richard Scott Bell	Northwest Missouri State University, Maryville, MO
Henry Walker	Grinnell College, Grinnell, IA
Mahmoud Yousef	University of Central Missouri, Warrensburg, MO
Judy Mullins	University of Missouri-Kansas City, Kansas City, MO
Carol Spradling	Northwest Missouri State University, Maryville, MO
Baochuan Lu	Southwest Baptist University, Bolivar, MO
Timothy Urness	Drake University, Des Moines, IA
William Siever	Washington University in St. Louis, St. Louis, MO
Ajay Bandi	Northwest Missouri State University, Maryville, MO
Denise Case	Northwest Missouri State University, Maryville, MO
Jamil Saquer	Missouri State University, Springfield, MO
Rad Alrifai	Northeastern State University, Tahlequah, OK
Dabin Ding	University of Central Missouri, Warrensburg, MO
Aziz Fellah	Northwest Missouri State University, Maryville, MO
Charitha Hettiarachchi	Northwest Missouri State University, Maryville, MO
Beth Arrowsmith	University of Missouri-St. Louis, Saint Peters, MO
Ken Vollmar	Missouri State University, Springfield, MO
Jose Metrolho	Polytechnic Institute of Castelo Branco, Portugal
David Furcy	University of Wisconsin Oshkosh, Oshkosh, WI
Ernest Ferguson	Northwest Missouri State University, Maryville, MO
George Dimitoglou	Hood College, Frederick, MD
John Buerck	Saint Louis University, St. Louis, MO

## Invited Talks

### Friday Opening Keynote Speaker: Andy Nelson

Andy is a software engineer and manager with a strong passion for cybersecurity. Originally from Indiana, he graduated from Purdue University in 2008. He went to work immediately at Cerner Corporation predominantly developing java applications and services for his first 8 years. For the last 3 years he has been focused on creating tools that identify and report on security vulnerabilities for software at Cerner. Aside from leading a team of engineers, he is also responsible for Cerner's monthly cybersecurity meetup, creating internal engineering focused cybersecurity training, and creating a culture of engineering security.



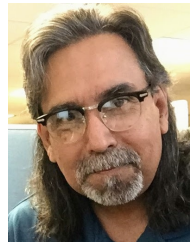
### Friday Banquet Keynote Speaker: Paul Barham

Paul has been Captain of the Kansas City Brigade of Code for America for the past five years. During that time, he has fostered a place for people to work on civic projects for nonprofits, city government and the UMKC School of Law. He has been a team member of the Community KC, Address API, and Expungement projects. In the past, he has been involved in the leadership of other computer organizations, and he has been a software developer for over 30 years.



### Saturday Keynote Speaker: Frank Alaniz

Frank Alaniz is an Air Force Veteran, Mentor and Project Manager with over a decade of successful experience in project management, public speaking and workforce program development. Frank specializes in workforce programs, human resource technologies and workforce analytic tools. He was recognized by the National Conference of Mayors for his development of workforce programs for Veterans in transition and the State of Missouri for his outreach to the business community and workforce program development for individuals in transition. Frank regularly develops hiring campaigns to assist employer and jobseeker interaction and is a contributor/speaker to the Missouri Association for Workforce Development professionals (MAWD), Business Persons Between Jobs (BBJ), and Beyond Networking-STL. Frank can typically be found coding in LAU and G-Code for his 3d printers or in his shop perfecting his metalsmithing techniques.





# Using a Short Textbook in CS 1 to Improve Student Reading \*

*David Toth, Thomas Allen*  
*Computer Science*  
*Centre College*  
*Danville, KY 40422*  
*{david.toth,thomas.allen}@centre.edu*

## Abstract

Over the last several years, most of our CS 1 students did not read the textbook despite us giving reading quizzes in class. This resulted in students not learning the material as well. In an attempt to remedy that to try to improve student learning, we wrote a concise textbook for our CS 1 class. We ran two sections of the course each semester for a year, one with the traditional textbook we had been using and the other with the concise book. We found that the students in the course with the concise book read the assigned pages regularly and spent more time reading the book and trying the code in it.

## 1 Introduction

Our institution is a small liberal arts college, near the top 50 schools in the U.S. News Rankings. Our CS 1 course is taken not only by students who want to major or minor in computer science, but also by many students who are just using the course to fill a general education requirement. Our course is taught in two-hour blocks Monday, Wednesday, and Friday, giving us 6 contact hours per week. The first hour each day is nominally for lecture and the second hour is for lab activities. Most of the students in our CS course in the last several years had not been reading the assigned pages before class, even though we gave reading quizzes in class on the assigned sections. Because of this, most

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

of the students didn't learn any of the material before coming to class, which resulted in us having to lecture more and teach even the simpler topics the students could have picked up from the reading. The lecturing took more than the desired hour, which meant students had less time lab time to engage with the material. This resulted in students not learning the material as well as we wanted, taking longer to finish labs and almost always having to finish them at home, which resulted in student dissatisfaction.

We believed that part of the reason students didn't do the reading was because they felt it took too long, as readings were typically about 20 pages per class. Recognizing the short attention span and over-scheduling of today's students, we thought that if we could condense the material we wanted students to learn before class into 3-5 pages, we might be able to persuade them to do the reading. We hoped this would result in students learning the easier material on their own, so we could cover the more challenging topics with them in class, freeing up more of the two-hour block for lab time. Although we looked through many textbooks and trade books, we could not find one we liked where the material was organized in such a way that we could easily make reading assignments without students needing to jump around in the book much, and thus we wrote our own 63-page book for the course [1]. Our book presents only the basic concepts and syntax, leaving more challenging concepts out of the book, so they must be covered in class. The book also only has a subset of the Python 3 syntax, in an attempt to pull the focus of the course back to the problem-solving aspect and not sidetrack students with lots of unnecessary syntax and portions of the language they do not need.

## **2 Related Work**

Others have tried using different minimalist approaches to teaching. Edgcomb, Vahid, and Lysecky studied the effect of using less text to teach the same concept and found that resulted in better learning [2]. Nasrawt and Lam used a custom-built procedural language that has less syntax than a full language like Java or C++ [3]. Although they did not do a formal study, their observations showed that students without prior knowledge of their language were able to solve problems using it with only limited assistance.

## **3 How the Course with the Concise Book Was Run**

We set the expectations for the section of the course that used the concise book on the first day of class with the students, explaining that every day at the end of class, if there was a reading assignment for the next class, it would be posted on Moodle and announced in class. Students were told they were expected to

do the reading before class and that the lecture and lab topics in class would build off the reading, so if the students did not do the reading they would not understand the material in class. In addition to doing the 3-5 pages of reading before class, the students were to do the small number of graded homework problems at the end of the reading, and turn them in electronically before the upcoming class. At the beginning of the upcoming class, there would be a quiz on the reading and homework. However, the professor would not answer questions at the beginning of class, so students were expected to do the reading and come ask questions in the professor's office hours before the class period. This was to ensure students read the material and assimilated it before class. Because the students were not going to be able to ask questions in class before the quiz, the professor made sure the students knew that he was available not only during his 6 scheduled office hours per week with some every day, but he would make extra time for students if they could not make it to the office hours.

## 4 Our Study

In order to determine if using the concise book was more effective at getting the students to do the reading, we created a survey for students and had it approved by our institutional review board (IRB). The survey was given to the students near the middle of the semester and then again near the end of the semester in both the fall and the spring semesters. We did this in case reflection at the end of the semester changed students' perceptions of the effectiveness of the textbooks. The survey questions were:

1. Name (this will be removed before the instructors are given the survey).
2. Please select the option below that most accurately reflects how frequently you read the assigned chapters in the book thoroughly and carefully.
  - Never or almost never
  - Infrequently
  - About half the time
  - Most of the time
  - Always or almost always
3. Please select the option below that most accurately reflects how frequently you type in the sample code from the book and run it.
  - Never or almost never
  - Infrequently

- About half the time
- Most of the time
- Always or almost always

4. Approximately how much time do you spend working with the book (reading and trying code samples in the assigned reading) each week. \_\_\_\_hours

5. How effective do you think the textbook has been in helping you learn the concepts in the course?

- Extremely ineffective
- Somewhat ineffective
- Neither effective or ineffective
- Somewhat effective
- Extremely effective

6. What changes would make the book better?

Questions 7 and 8 were based on the book the students used for their course. Students who were in the course using the concise book were given these questions:

7. Do you think a longer, more comprehensive book would have been noticeably more useful? Why or why not?

8. If you had a more comprehensive book with longer readings (about 20 pages each time instead of 3-5 pages) how likely would you have been to read the assigned readings thoroughly and try the code examples?

- Extremely unlikely
- Somewhat unlikely
- Neither unlikely or likely
- Somewhat likely
- Extremely likely

Students who were in the course using the regular book were given these questions:

7. Do you think a shorter, less comprehensive book would have been noticeably less useful? Why or why not?

8. If you had a shorter and less comprehensive book with shorter readings (about 3-5 pages each time instead of 20 pages) how likely would you have been to read the assigned readings thoroughly and try the code examples?

- Extremely unlikely
- Somewhat unlikely
- Neither unlikely or likely
- Somewhat likely
- Extremely likely

The survey was administered by the professor teaching the other section of the course, so section A's professor gave the survey to section B's students and vice-versa. The survey was given during the middle of the two-hour class period. Students were informed that the professor teaching their section would not see the results until (1) after grades were submitted and (2) until the names had been removed from the data. The professor teaching a section left the classroom while the other professor administered the survey. We asked for names on the surveys so we could correlate the answers from the midterm survey with the answers from the end of semester survey to see if a given student's opinion had changed. However, once the surveys were done, we stapled the two surveys from each student together and removed the names to anonymize the data. The surveys were optional for the students and they didn't receive any compensation for participating. They were conducted during class time so as not to require students to give up time outside of class to participate. All 56 students in the courses participated. They were split between the sections of the course as shown in Table 1. However, 5 students did not fill out the second survey in a semester due to absences or choosing not to do so.

Table 1: Student Participation in the Study

Class	Number of Participating Students
Fall Class Short Book	10
Spring Class Short Book	7
Fall Class Normal Book	24
Spring Class Normal Book	15

## 5 Results

The results of the surveys are shown in Tables 2-7. We did not include the answers to questions 6 and 7 from the survey, as these were intended to get feedback on the book to improve it over the upcoming year, but is not relevant to this study. We converted the answers the students gave to numerical values so we could take the average of them to get an average response.

The values for questions 2 and 3 were:

- 0: Never or almost never
- 1: Infrequently
- 2: About half the time
- 3: Most of the time
- 4: Always or almost always

The values for question 5 were:

- 0: Extremely ineffective
- 1: Somewhat ineffective
- 2: Neither effective or ineffective
- 3: Somewhat effective
- 4: Extremely effective

The values for question 8 were:

- 0: Extremely unlikely
- 1: Somewhat unlikely
- 2: Neither unlikely or likely
- 3: Somewhat likely
- 4: Extremely likely

Table 2 shows that the students in the sections of the course that used the concise book read the book thoroughly and carefully more frequently than the students in the other sections of the course.

Table 2: Frequency of Reading the Book Thoroughly and Carefully

Class	Mid-Semester Survey	End-of-Semester Survey
Fall Class Short Book	3.5	3.6
Spring Class Short Book	3.9	3.9
Fall Class Normal Book	1.6	1.3
Spring Class Normal Book	2.1	2.2

Table 3 shows that the students in the sections of the course that used the concise book typed in and ran the code in the book more frequently than the students in the other sections of the course.

Table 4 shows the number of hours students in both sections of the course spent working with the book. We note that the students in the course sections

Table 3: Frequency of Typing in and Running the Code

Class	Mid-Semester Survey	End-of-Semester Survey
Fall Class Short Book	2.9	2.8
Spring Class Short Book	3.6	3.7
Fall Class Normal Book	0.6	0.5
Spring Class Normal Book	1.2	1.0

Table 4: Hours Spent Working with the Book

Class	Mid-Semester Survey	End-of-Semester Survey
Fall Class Short Book	3.5	4.0
Spring Class Short Book	4.8	3.9
Fall Class Normal Book	1.5	1.7
Spring Class Normal Book	2.3	3.3

that used the concise book spent more time than the students in the sections that used the longer book.

Table 5 shows that the students in the sections of the course that used the concise book felt the book was more effective than the students in the other sections of the course felt their book was.

Table 5: Effectiveness of the Book

Class	Mid-Semester Survey	End-of-Semester Survey
Fall Class Short Book	3.5	3.4
Spring Class Short Book	3.7	3.6
Fall Class Normal Book	2.8	2.4
Spring Class Normal Book	2.5	2.3

Table 6 shows the likelihood of students of students in the class with the concise book doing the readings and trying the code if the readings had been 20 pages from a longer book instead of 3-5 pages. Students indicated that they were neither unlikely or likely to do the readings and try the code examples. This is in contrast to them actually doing the 3-5 page readings most of the time for the course.

Table 7 shows the likelihood of students of students in the class with the normal book doing the readings and trying the code if the readings had been 3-5 pages from a shorter book instead of 20 pages. Students indicated that they were somewhat likely to do the readings and try the code examples. This

Table 6: Likelihood of Doing the Readings and Trying the Code with Longer Readings

Class	Mid-Semester Survey	End-of-Semester Survey
Fall Class Short Book	1.9	2.1
Spring Class Short Book	1.0	1.2

is in contrast to them actually doing the 20 page readings only about half the time and trying the code examples infrequently for the course.

Table 7: Likelihood of Doing the Readings and Trying the Code with Shorter Readings

Class	Mid-Semester Survey	End-of-Semester Survey
Fall Class Normal Book	3.3	3.0
Spring Class Normal Book	3.3	3.3

## 6 Observations and Conclusions

From a data-driven perspective, the students using the concise book felt the book was effective more than the students using the normal book felt the book was effective. The students using the concise book did the readings and worked with the code samples in the book at a significantly higher frequency than the students using the normal book. The students using the concise book also spent more time working with it than the students using the normal book. Students with the concise book said they were less likely to do the readings and work with the sample code if they had the normal book with longer reading assignments. Students with the normal book indicated they were likely to do the readings and work with the code if they had shorter readings, whereas they did the readings in the normal book about half the time or less.

Anecdotally, the author who taught the sections using the concise book has taught it a number of times using the normal book, and he observed that students seemed more prepared for class in general when using the concise book. He did not have to teach everything from scratch, but was able to skip right to the more challenging topics for the day, since the students had learned the basics before coming to class. In that sense, the concise book was definitely a success. Given our data and observations, we have begun using the concise book in all of the sections of the course.



## References

- [1] David Toth. *Programming and Problem-Solving with A Subset of Python 3, Beta Edition*. Linus Learning, Ronkonkoma, NY, 2018.
- [2] Alex Edgcomb, Frank Vahid, and Roman Lysecky. Students learn more with less text that covers the same core topics. In *Proceedings of the 2015 IEEE Frontiers in Education Conference, FIE '15*, pages 1–5, 2015.
- [3] Zamua O. Nasrawt and Michael O. Lam. Less-Java, more learning: Language design for introductory programming. *J. Comput. Sci. Coll.*, 34(3):64–72, 2019. <http://dl.acm.org/citation.cfm?id=3306465.3306476>.

# An Alternative to Programming Contests\*

*Timothy Urness*

*Department of Mathematics and Computer Science*

*Drake University*

*Des Moines, IA 50311*

*`timothy.urness@drake.edu`*

## Abstract

Programming contests, such as the International Collegiate Programming Contest (ICPC) have many benefits for undergraduate students studying computer science. These include encouraging collaboration amongst teammates and providing an incentive to study of algorithms and data structures. However, these multi-hour, time-pressured contests also have the potential to discourage students and promote bad habits in software development (e.g. rushed development, non-reusable code, and the lack of documentation). In this paper, we describe an alternative to traditional programming contests that involve student-led projects that span an entire semester. These projects, which include long-term data science competitions and student-led research projects, have the benefit of exploring the cutting-edge of technology while giving students the opportunity to collaborate, learn from mistakes, and develop robust software that incorporates aspects of software engineering.

## 1 Introduction

Programming contests, such as the International Collegiate Programming Contest (ICPC) have been a popular activity amongst undergraduate computer science programs. The contests can benefit students in many ways. The ICPC website (<https://icpc.baylor.edu/regionals/abouticpc>) states:

“The contest fosters creativity, teamwork, and innovation in building new software programs, and enables students to test their ability

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

to perform under pressure. The contest has raised aspirations and performance of generations of the world's problem solvers in the computing sciences and engineering."

We have seen, firsthand, many of these benefits described above. The contest provides an atmosphere where students are excited to compete and debrief after the contest has concluded. The contest itself can motivate students to join a club to develop skills and deepen their knowledge of computer science algorithms and data structures in order to be more successful in future competitions.

However, we have also witnessed situations where students leave the contest dejected for a number of reasons. Primarily, students can be frustrated that code would pass all of the examples provided, but not pass all of the hidden testing examples. The contest rules stipulate the ranges of boundary cases that are not thoroughly supplied in the testing examples provided. While this is an important component of the contest, the feedback given to the students is limited. This black-box testing leaves little room for constructive feedback and has caused students to leave the contest feeling frustrated, inadequate, and under-educated. Instead, a more important skill that we would like to encourage is the dedication to utilize available resources to find a solution after a problem is encountered. The short-term nature of the competition has the potential to discourage students that would otherwise thrive if time and resources were available.

Another practice that is unintentionally encouraged by time-pressured programming contests is the development of hastily-developed code designed to run only once. Much of the code that is constructed in programming contests does not adhere to well-accepted software engineering practices (e.g. documentation, object-oriented programming, reusable code). Short-term contests do not necessarily promote the skills of software developers that computer science professors are often encouraging thorough the curriculum.

Lastly, IBM has recently discontinued as the ICPC corporate sponsor. At the 2018 and 2019 North Central North America Regional contest, participation required a \$25 fee per participant. The fees may prove to be a burden to departments wishing to open the contests to all interested students.

## 2 Related Work

There have been other alternatives to programming contests proposed. Many of these references make a point that the ICPC stresses skills that are not always in line with modern computer science and software engineering degree programs. Competitions can play a productive role in education and competitive desires can be utilized for educational motivation [4], though competition

in education can be good or bad [3]. The best results are often obtained when competition is combined with cooperation. Furthermore, competitions should not be the only factor in assessment and is best utilized as a complement to standard teaching [3].

The College of Charleston student chapter of the ACM hosted a contest in which the judging criteria included both technical and artistic merit[1]. The contest consisted of problems that included some working scaffolding code and a *syntax master expert*, who could answer any student question about syntax. Furthermore, the problem statements encourage a design phase, an implementation phase, a testing phase, and a final submission. In addition to technical correctness, the judging rubric included several sections that evaluated the quality of the submitted code, testing cases, and results.

Constantinescu et. al describes a contest that contains a component aimed at boosting students' creativity and involves a presentation of a finished project to the judges [2]. A high-scoring balanced solution considers multiple facets of the contestants' talent, effort, and results.

In this paper, we provide another alternative to a multi-hour, time-pressured programming contest.

### 3 An Alternative to the Programming Contest

As an alternative to the programming contest, we've developed semester-long, student-led research groups that have faculty mentors. The groups are designed to be a place where students can explore exciting and cutting-edge aspects of computer science and develop skills over the course of several months. The groups are student-led where faculty involvement can be minimal and is more akin to a mentor or coach than a research director.

The groups are not associated with any credit-bearing course or internship. Instead, we incentivize students by stressing the fun nature of exploring the cutting-edge of technology, the collegiality of being involved with a group, and the potential to contribute to a published research paper. In addition, we stress that participation in a group has the strong potential to increase the strength and warmth of a recommendation letter that a professor could write for a student.

#### 3.1 Forming Groups

During the first week of the semester, we invite all students studying computer science, mathematics, or data analytics to an informative meeting to introduce the idea of student-led research groups. We present many possible ideas for areas of study. In the past, project ideas have included: virtual reality, game development, robotics, mathematical modeling, machine learning, computer

science theory, hardware, 3D printing, data science competitions, and data mining. These topics align roughly with faculty interests, but also include topics at the suggestions of students. After the initial meeting, we have students separate into smaller groups of common interests.

### 3.2 Student-Led Projects

As part of smaller groups, students are tasked with developing concrete goals for the semester. This critical juncture of the projects may require the most professor guidance as the specific goals will determine the long-term success of a project. For example, the transition to a vague interest in a topic such as virtual reality to a specific project (e.g. creating a VR museum) has the potential to cause students to sustain interest or dissuade involvement based on the individual preferences. Students in the smaller groups also determine the best time to meet and accomplish short-term goals. The short-term goals often times include identifying tutorials, development environments, and small tasks to bring to the larger group.

### 3.3 Online Competitions

Another option for the research groups is to work on an online data science competition. Table 1 shows several examples of data science competitions from kaggle.com that were open in fall of 2019. The prize money is likely out of range for an undergraduate team, particularly with little experience, but the potential of competing for prizes may be the positive motivation that drives students to learn more [4].

Table 1: Fall 2019 kaggle.com Contest Examples

<b>Title</b>	<b>Description</b>	<b>Prize Money</b>
NFL Big Data Bowl	How many yards will an NFL player gain after receiving a handoff?	\$75,000
ASHRAE - Great Energy Predictor III	How much energy will a building consume?	\$25,000
Understanding Clouds from Satellite Images	Can you classify cloud structures from satellites?	\$10,000
House Prices: Advanced Regression Techniques	Predict sales prices and practice feature engineering, RFs, and gradient boosting	Training Set Only

The learning curve for these competitions is rather gradual as most com-

petitions on kaggle.com also come with introductory tutorials. A group of students can follow a tutorial that will allow them to make a submission to the competition. Then, they can continue to work, refine the algorithm, learn new techniques and re-submit and see if they are able to climb the leaderboard. This long-term learning process rewards collaboration, elaborate solutions, and the exploration of current technology.

There are many other online competitions that groups of students or individuals can utilize to the same effect as kaggle.com. For example, Hacker Rank (<https://www.hackerrank.com>) provides competitive programming challenges. Programmers are ranked on a leaderboard and can earn badges based on accomplishments. Another online site that provides interesting problems that could be solved in a similar fashion to a programming contest is Project Euler (<https://projecteuler.net>).

### **3.4 Goals**

The goals for the computer science department at Drake University include providing opportunities for students to participate in beyond-the-classroom projects. We feel that this provides learning opportunities that can complement the traditional curriculum and develops recognition for the program. Concepts such as team-programming, self-motivated learners, code repositories, using code libraries, and learning new concepts by exploring and synthesizing online content are skills that students will need to develop as part of today's technological industry. We feel that students' educations can be enhanced with more dedicated practice with these important skills and concepts, both in and out of the classroom.

Furthermore, if the department can establish an environment where out-of-classroom learning becomes the norm, the research groups can be self-sustaining and produce outcomes after the students that started a particular group have graduated.

### **3.5 The Role of the Professor**

One of the main motivating factors for starting student-led projects is to allow any student, regardless of their experience, to contribute to a research project. In the past, professors would typically select the top few students from a class to join a research project which they directed. If a student was not fortunate enough to get the attention of a professor, his or her education might lack this enrichment opportunity. To give all students an opportunity, we wanted to provide an experience that we could honestly market to all prospective students and not just the students with the potential to be amongst the top few percent.

However, opening up research opportunities to all students has the potential to monopolize a willing professor's time and resources. In order to mitigate this, we set up the initial meetings and opportunities to take place within an hour on Friday afternoons. This is when willing professors will make themselves available for consultation on various projects.

We intentionally have named the projects "student-led" projects and remind students that the main thrust of motivation must come from the students, and not the professor. In this context, professors are seen as coaches and mentors, and not the driving force of motivation behind the projects [5].

## 4 A Case Study

### 4.1 Semester-Long Research Group and a Programming Contest Group

During the academic year of 2018-2019, a group of students at Drake University participated in both the student-led research groups. A different group of students participated in the ICPC programming contest in the fall.

The research group met approximately one to two hours a week either independently or with a faculty member over the course of both semesters. The programming contest team only met once or twice prior to the programming contest event held in the fall.

We had over 40 students participate in student research groups and 9 students participate in the programming contest. At the end of the academic year, we asked both groups to anonymously answer two questions on a Likert-like scale from 1 to 5 to assess the student satisfaction with the experience as well as the student's likelihood to participate against next year. The survey also allowed for students to give optional written feedback. The questions posed to the students were as follows:

- How satisfied were you in participating in the event (programming contest or the research group)?
- If you are able, how likely are you to participate next year?
- (optional) Do you have any additional comments?

Twelve students responded for the research group survey, and five students responded for the programming contest survey. Valuable insights into the effectiveness of both groups is available. The results of the satisfaction question are shown in Table 2 and Figure 1. The results of the likelihood of participating in the next year are shown in Table 3 and Figure 2.

Table 2: Satisfaction Survey Results

Group	1 (not satisfied)	2	3	4	5 (very satisfied)
Research Projects	0%	0%	33%	33%	33%
Programming Contest	0%	0%	20%	40%	40%

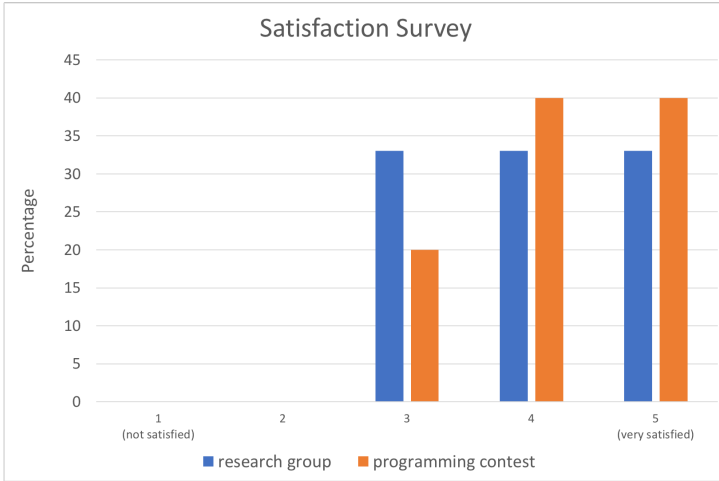


Figure 1: Visualization of the data displayed in Table 2. Answers to the question, "How satisfied were you in participating in the event?"

## 4.2 Observations and Insights

In aggregate, both groups of students had positive experiences. Each student that completed the survey indicated that they were satisfied with the experience, indicating their satisfaction at a level of 3 (satisfied), 4, or 5 (very satisfied) out of a 5-point scale. A vast majority of the students also indicated that they were likely or very likely (5 out of 5) to participate in a group again in the next academic year. The survey also allowed for a few deeper insights.

The student-led research groups had a higher percentage (66.7% vs 40%) of students rate their likelihood of participating again next year as "very likely" (5 out of 5). However, there were also a few students that rated their likelihood in participating in the research experience next year as "not likely" (1 out of 5) and another student rate their satisfaction as 2 out of 5. The open-ended comments in the survey indicated that at least one student felt that the research experience could be improved by better articulating short, medium, and long-



Table 3: Likelihood of Returning Survey Results

Group	1 (not likely)	2	3	4	5 (very likely)
Research Projects	8.3%	8.3%	8.3%	8.3%	66.7%
Programming Contest	0%	0%	0%	60%	40%

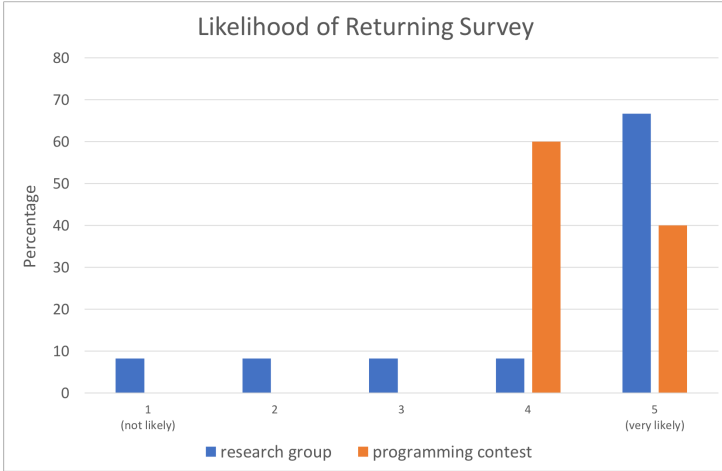


Figure 2: Likelihood of Returning Survey. Visualization of the data displayed in Table 3. Answers to the question, "If you are able, how likely are you to participate next year?"

range goals for each of the students. As the participation in these projects does not hold external incentives in the form of college credit or grades, making the other incentives (e.g. student camaraderie, future letters of recommendation, exploring the cutting-edge of a discipline) may need to be emphasized.

The results from the programming contest survey were surprisingly positive. After the contest, students' reactions were initially mixed. On the ride home, student verbally commented about their frustration with the level of feedback allowed, the length of the multi-hour contest, and the contest structure itself. However, those that responded to the survey indicated they were largely satisfied and likely to participate again next year. We feel that the programming contest team experience could also be improved by treating the group as a club, with regularly-scheduled meetings, practice sessions, and mentoring opportunities designed to mitigate the potentially negative characteristics of a programming contest identified in section 1.

## 5 Conclusion

In this paper, we've described an alternative to multi-hour, time-pressured programming contests by providing students with opportunities to get involved in semester-long student-led research groups. These groups have advantages over programming contests as they can promote the skills of software developers, facilitate discovery of new concepts not introduced in standard courses, and still provide a competition that can motivate students. A survey of students that participated in either the research groups or the programming contest at Drake University in 2018-2019 found that students found both kinds of extra-curricular involvement satisfying, and a vast majority of those that responded will plan to do the events in the future. The student-led research groups had a higher percentage of students rate their likelihood of participating again next year as "very likely" (5 out of 5), but also had a few students that will likely not participate in the future.

Overall, the student-led research groups provide an alternative, and not necessarily a replacement, to the traditional programming contest for some students. These projects, which include long-term data science competitions and student-led research projects, have the benefit of exploring the cutting-edge of technology while giving students the opportunity to collaborate, learn from mistakes, and develop robust software that incorporates aspects of software engineering.

## References

- [1] James F. Bowring. A new paradigm for programming competitions. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '08, pages 87–91, New York, NY, USA, 2008. ACM.
- [2] Zoran Constantinescu, Simona Nicoara, Monica Vladoiu, and Gabriela Moise. Computer science student contests: individuals or teams. In *Proceedings of the 16th RoEduNet International Conference: Networking in Education and Research*, Petru-Maior University of Targu Mures, Romania, 2017.
- [3] Fredrik Kristensen, Olof Troeng, Mohammadhassan Safavi, and Prakash Narayanan. *Competition in higher education—good or bad?* Lund University, 2016.
- [4] Tom Verhoeff. The role of competitions in education. *Future world: Educating for the 21st century*, pages 1–10, 1997.
- [5] Matthew Zwier and Timothy Urness. Just in time research: The advantages and pitfalls of a student-led interdisciplinary undergraduate research experience. *J. Comput. Sci. Coll.*, 33(5):179–185, May 2018.

# Design of Virtual Labs for an Ethical Hacking Course\*

*Xiaodong Yue and Hyungbae Park*  
*Department of Computer Science*  
*University of Central Missouri*  
*Warrensburg, MO 64093*  
*{yue,park}@ucmo.edu*

## Abstract

The cybersecurity job is booming in many regions of the country. According to the Department of Labor Bureau of Labor Statistics, employment of information security analysts is projected to grow 32% from 2018 to 2028, much faster than the average for all occupations. In response to the high demand of cybersecurity professionals, many higher education institutions are adding Cybersecurity programs and/or courses. In this paper, a case study is presented based on the lessons learned and experiences gained from the design of virtual labs for an ethical hacking course. The findings and recommendations summarized in this paper could be adopted in a similar setting by other institutions, especially those with limited resources.

## 1 Introduction

Demand for information security analysts is expected to be very high, as these analysts will be needed to create innovative solutions to prevent hackers from stealing critical information or causing problems for computer networks/infrastructures. To help higher education institutions to develop cybersecurity curriculum, the Association for Computing Machinery (ACM) released the Curriculum Guidelines for Post-Secondary Degree Programs in Cybersecurity in 2017. In 2018, the Accreditation Board for Engineering and Technology

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

(ABET) also developed accreditation criteria for undergraduate Cybersecurity programs.

The National Security Agency (NSA) and the Department of Homeland Security (DHS) co-sponsored the National Centers of Academic Excellence in Cyber Defense (CAE-CDE) Program which provides a guideline for institutions to prepare a workforce to defend the Nation’s information infrastructures. NSA defined a set of Knowledge Units (KUs) as the criteria to evaluate the CAEs. A class that focuses on teaching ethical hacking concepts and techniques is usually required to be incorporated into the core of the cybersecurity curriculum in order to prepare the future cybersecurity professionals. In this paper, we will discuss the design of 16 virtual labs used in our ethical hacking class. We hope that other programs will benefit from the experiences and findings discussed in this paper.

## 2 Background

In recent years, ethical hacking began appearing in the cybersecurity curricula at higher education institutions. In a typical ethical hacking (penetration testing) course, the students are exposed to a plethora of open-source and commercial security tools that are used by both security professionals and the “black hat” hackers. In order to help students to understand the concepts learned from the class and help them to practice hacking skills, labs are usually included in the course. Since many of the hacking lab exercises are dangerous to perform on a production network and could pose a security risk to the institutional or home networks, they are usually conducted in a virtual environment. Currently, there are several solutions to provide the virtual hacking lab environment.

### Virtual labs Developed by Third Party Companies

Recently, a few companies such as Infosec Learning, etc. provide online, cloud based remote accessible laboratory environment. Internet access and a browser are all you need to access these labs. The pro for such solution is that the institution is not responsible to purchase, install, update or upgrade the hardware and software associated with the virtual labs. As a result, there is no investment from the institution side. In addition, the institution does not need to provide technical supports to the virtual environment. The con is that although the access is typically free for instructors, students do need to pay a fee to access the labs which is not cheap. In addition, the access is usually limited to one semester. After semester ends, the access expires and students have to pay the fee again in order to access the labs in case they’d like to continue to

practice their hacking skills. It also worth mentioning that those labs are pre-canned which are extremely difficult to be tailored to meet individual course's need. In addition, the lab materials are often not aligned well with topics to be covered in an ethical hacking class.

A few professional training companies such as SANS and EC-Council, etc. also developed curriculum and associated labs for ethical hacking (penetration testing). For example, SANS offers a comprehensive penetration testing curriculum that consists of 10 courses and associated certificate exams. These curricula have excellent contents with most recent technologies and lab exercises. Furthermore, the SANS's labs are closely aligned with the current industry practice in penetration testing. However, they were created for short-term intensive training for the field professionals, usually in a one-week boot camp format. In addition, the curriculum and labs created by SANS are proprietary which cannot be duplicated for classroom use. EC-Council has a popular certificate exam and training curriculum for Certified Ethical Hacker (CEH). Although EC-Council has a version of its ethical hacking curriculum that could be adopted by academia, its virtual lab environment - iLabs requires a fee to access.

## **Virtual labs Developed by Textbook Authors and Other Researchers**

Over the past several years, researchers developed various security labs for education purpose. For example, SEED labs developed by Du [2] include over 30 labs for various security topics. However, those labs are very general which do not target the ethical hacking concepts. In addition, the lab environment is purely Linux based which does not include the Windows operating system. ITSEED is a similar project by Wang [4]. Unlike the SEED labs, the ITSEED project is not actively developed and maintained by its developer. [1, 3] designed virtual labs used for teaching offensive security in a higher education setting. The tools used in those labs are somewhat limited. Instead of designing comprehensive security labs, some researchers [6, 5] focused on designing specialized labs for a particular topic. Since those labs are very limited in topics, it is difficult to meet the whole course need. Since offering offensive security course is fairly recent in higher education, there are not many suitable ethical hacking textbooks available for a higher learning setting. The author's ethical hacking class adopts Georgia Weidman's book titled *Penetration Testing A Hands-On Introduction to Hacking*. The textbook author did provide a virtual lab environment for the book. However, since the book was published in 2014, some of the lab contents and tools are outdated.

### 3 Design of the Virtual Hacking Labs

Based on the discussions from the previous section, it is difficult to find one fit for all solution from the current available options. As a result, we decided to build a customized virtual hacking lab environment to closely follow the industry trend in penetration testing while minimizing the costs for the institution. Those labs train students with practical hands-on skills while helping to prepare them for industry penetration testing certificates. Students have free access to the labs and they can continue to access those labs after semester ends as long as they are still a major in the department. The designed labs follow the work flow of a penetration tester as defined in the Penetration Testing Execution Standard (PTES). We divide our 16 labs into four phases of the penetration testing: 1. Reconnaissance, 2. Scanning, 3. Exploitation, and 4. Post-Exploitation. Those labs align well with a 15-week semester which provide students with one lab per week on average.

Lab manuals with detailed step-by-step instructions with screenshots which highlight important steps and commands are provided for the students to complete each lab. Students are required to write a report of each lab by completing the required tasks and answering questions related to the topics covered in each lab.

#### 3.1 Lab Settings

A pod of six virtual machines (VMs) are used in our lab environment. They are: 1. Kali Linux by Offensive Security, 2. Ubuntu Linux by the textbook author, 3. Metasploitable 2 Linux by Rapid7, 4. Customized Windows XP, 5. Customized Windows 7, and 6. Customized Windows 10. Kali Linux serves as the attacker machine which the other five VMs serve as victim machines.

The virtualization product used to host the pod of VMs in our labs are VMware Workstation Pro. Although license is required for using this product, VMware provides significant discounts to academic institutions through the VMware Academic Software Licensing Program. It also worth noting that our VMs can be run from other free virtualization products such as VirtualBox, etc. if such subscription is not available. The three Linux VMs free software while the three Windows VMs are licensed. Once again, academic institutions receive significant discounts throughout Microsoft academic license subscription.

Most of our labs employ open source tools and systems developed and distributed by the security communities. We use additional commercial tools such as Nessus, etc. provided by security vendors free-of-charge for educational purposes.

## 3.2 Lab Overviews

### Module 1: Reconnaissance

Lab 1: NetCat Relay

Objectives: 1. Use iptables to configure firewall rules on the Linux machine, 2. Use Netcat to relay network traffic, 3. Pivot through a system to get access to a service listening on another system that is firewalled and unavailable to the attacker.

Lab 2: Recon-ng for Reconnaissance

Objectives: 1. Perform DNS recon using commands such as host, dig and nslookup, 2. Use Recon-ng to perform DNS reverse lookup, 3. Use Recon-ng to perform DNS cache snooping to identify antivirus tools used by the target organizations, 4. Use Recon-ng to perform other reconnaissance activities.

### Module 2: Scanning

Lab 3: Scanning using Nmap

Objectives: 1. Use Nmap to perform various scans such as TCP, UDP, version and OS fingerprinting, 2. Use Nmap Scripting Engine (NSE), 3. Compare how Nmap behaves when NSE scripts are run with and without version scanning.

Lab 4: Scapy for Pentest

Objectives: 1. Use Tcpcat to sniff and analyze packets, 2. Use Scapy to craft packets in scenarios that are highly useful in penetration testing, 3. Use the Scapy module to write Python scripts to perform various pentest activities.

Lab 5: Ettercap for the Man-in-the middle Attack

Objective: 1. Use Wireshark to sniff and analyze packets, 2. Use Ettercap to perform ARP cache poisoning and DNS cache poisoning, 3. Use the Scapy module to write Python scripts to perform TCP RESET attack on FTP and SSH.

Lab 6: Nessus and Metasploit Database

Objectives: 1. To run vulnerability scan using Nessus, 2. To use Metasploit database and analyze its contents for future penetration testing needs, 3. To run a Metasploit module from a script.

### Module 3: Exploitation

Lab 7: Metasploit Pivoting

Objectives: 1. Use Metasploit psexec module, 2. Pivot using Metasploit route, 3. Use Metasploit auxiliary modules for port scan and proxy.

Lab 8: Service Side Exploitation, Meterpreter and Port Forwarding

Objectives: 1. To use Metasploit to perform a service side attack, 2. To use Meterpreter to perform various penetration testing tasks, 3. To use Meterpreter port forward feature to pivot through a compromised system to get access to a listening service on another system.

Lab 9: Client Side Exploitation and Antivirus Evasion

Objectives: 1. Use msfvenom to create a stand-alone payload, 2. Use the exploit/multi/handler module in Metasploit, 3. Use Veil Evasion to create payload to evade antivirus tools, 4. Use different ways to deliver payload to the target systems.

#### **Module 4: Post Exploitation**

Lab 10: PowerShell Empire Framework for Post Exploitation

Objectives: 1. Use the PowerShell Empire framework for various post exploitation activities, 2. Search and use various Empire modules, 3. Perform local privilege escalation.

Lab 11: Windows Command Line for Post Exploitation

Objectives: Use Windows command lines to perform tasks which are common in penetration testing

Lab 12: Running Commands on Remote Machines

Objectives: 1. Use Microsoft Sysinternals psexec to run commands on a remote machine, 2. Use different approaches to setup monitor to monitor port activities, 3. Use sc to run commands on a remote machine, 4. Use wmic to run commands on a remote machine and manipulate processes.

Lab 13: PowerShell in Post Exploitation

Objectives: Use PowerShell to perform tasks that are highly useful in penetration testing such as file searching, ping sweep, port scanning, DNS reverse lookup, running commands remotely and downloading files, etc.

Lab 14: Hydra Password Guessing

Objectives: Use hydra to conduct password guessing attack on SSH and SMB.

Lab 15: Password Cracking Using John the Ripper and Hashcat

Objectives: 1. Learn how to benchmark each hash algorithm supported by John and Hashcat, 2. Use John and Hashcat to crack LANMAN and NT hashes from Windows, 3. Use John and Hashcat to crack Linux hashes.

Lab 16: NTLM Sniffing, Cracking and Pass the Hash Attack

Objectives: 1. Use Tcpdump to sniff the NTLM packets, 2. Use Netcat to transfer files between two systems, 3. Use Cain to crack passwords, 4. Perform the pass the hash attack, 5. Use mimikatz Kiwi to dump hashes and cleartext password.

## **4 Preliminary Assessment Results**

The above mentioned labs were first used in our Ethical Hacking class in Spring 2019. 16 students were enrolled in the class. At the end of the semester, an anonymous assessment survey with six questions was administered during the class to seek students' feedback on the labs. Students were required to rate each question with a scale 1 to 5 with 1 indicating strongly disagree while 5



indicating strongly agree. 16 surveys were received. Table 1 is the result for the survey.

Table 1: Result for the Assessment Survey

Question	Score
1. The virtual testing environment is easy to build and use	4.25
2. Each lab's learning objectives are well defined and closely match the topics covered in the class	4.69
3. The lab instructions (including screenshots) are clear and easy to follow	4.44
4. Each lab's length is appropriate and I can finish it within a reasonable time frame	4.75
5. The labs help me to better understand Ethical Hacking topics covered in the class	4.81
6. The questions in each lab help me to reinforce the knowledge learned in the class	4.88

This preliminary result shows very positive feedback from the students. The author is teaching the same class in Fall 2019. Based on the results from Spring 2019, the author revised some of the lab components. A survey will be administered at the end of Fall 2019 semester. Results will be compared to analyzed to verify the effectiveness of those changes.

## 5 Conclusion

Hands-on lab exercises are a critical component of cybersecurity education. In this paper, we discussed the hacking labs developed at our institution which are designed to fit in a 15-week academic semester. These labs provide students with a free environment to practice hands-on skills at any time, any place with or without Internet access. The preliminary results from the students' feedback are very positive. Amid with the rapid evolvement of penetration testing technologies and tools, we will continue to refine our labs to align with the industry trend.

## References

- [1] Yu Cai. Using case studies to teach cybersecurity courses. *Journal of Cybersecurity Education, Research and Practice*, 2018(2).
- [2] Wenliang Du. SEED: Hands-on lab exercises for computer security education. *IEEE Security & Privacy*, 9(05):70–73, September 2011.

- [3] Chengcheng Li. Penetration testing curriculum development in practice. *Journal of Information Technology Education: Innovations in Practice*, 14:85–99, 2015.
- [4] Xinli Wang. ITSEED: Development of instructional laboratories for IT security education. Washington, D.C., November 2013. USENIX Association.
- [5] Yien Wang and Jianhua Yang. Ethical hacking and network defense: Choose your best network vulnerability scanning tool. In *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 110–113, March 2017.
- [6] Jianhua Yang, Yien Wang, and Thomas Reddington. Integrate hacking technique into information assurance education. In *2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 381–387, March 2016.

# Discrete Math: To Blend or Not Blend\*

*Charles Hoot*

*School of Computer Science and Information Systems*

*Northwest Missouri State University*

*Maryville, MO 64468*

*hoot@numissouri.edu*

## Abstract

This paper describes the experience of simultaneously teaching a blended section and a traditional face-to-face section of a Discrete Math course. Significant portions of the course materials were identical between the two versions, including the major assessments of student work, leading to an ability to more directly compare and discuss the outcomes in the two sections. In this comparison, the blended section typically lagged their counterparts in the traditional section. While the blended students were able to overcome their deficits in a majority of topics by the final exam, they still lagged on proofs.

## 1 Introduction

As the internet has become a pervasive presence in our lives, it was inevitable that it would be used to provide materials for traditional and non-traditional students. Non-traditional distance learners may take a course and rarely, if ever visit the campus that they are taking courses from. Introduced in 2006, Massive Open Online Courses (MOOC) have been an additional spur to traditional colleges to offer more content online [1]. Unfortunately, trying to integrate MOOCs into traditional courses has led to student dissatisfaction with the online portion of the content [5]. Students desire face-to-face interactions. To address issues of both accessibility and interactivity, courses that are a blend of online material paired with reduced hours in a face-to-face environment have been developed [3].

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

In the Fall of 2019, I took my traditional Discrete Math class and used it as a base to develop a blended (1 hour online 2 hours in class) version of the course. The intent was to determine the feasibility of offering a blended Discrete Math class and look for any pitfalls along the way. The following is a description of what was found.

## 2 The Course

The Discrete Math course is 200 level undergraduate class. Students in the course are typically either Computer Science or Math majors and are usually in their freshman or sophomore year. Topics include basic functions, sequences and summations, intuitive big-O, logic, sets, countability, recursive definitions, basic proofs, proofs by contradiction and induction, asymptotic analysis, and counting. Where possible, topics that are covered earlier in the course are revisited later to help reinforce the material. Thematically, the big topics are recursion and asymptotic analysis. As with any course, choices had to be made and some topics were left out.

1. While modular arithmetic is covered, there is not full coverage of number theory.
2. The notion of probability in terms of counting equal probability events is introduced, but not covered in depth.
3. Basic graph terminology, Euler circuits and graph colorings are introduced in the last week of class, but are not part of the material that is covered by the homework or exams.
4. Trees, relations, grammars, and automaton are not covered at all.

In most of these cases, the decision to leave out material relates to the fact that the Discrete Math class is being taught out of the Math department and there are Math courses that cover the material. As an example, there are full courses in the Math department on Probability and Statistics, Number Theory, and Graph Theory. Additionally, the Computer Science course on Algorithms covers graphs and trees in some detail.

## 3 Course Layout

In this section, the sequencing of the material and assessments in the traditional version of the course is laid out. This is followed by a discussion of the differences for the blended course.

### 3.1 Traditional

The traditional version of the course meets 3 times a week on Monday, Wednesday and Friday for 50 minutes and follows a regular pattern. Nearly every week there is a homework assignment that is due at the beginning of class. The homework is scanned and then returned immediately to the students. After class is over, a solution is published which leads into a quiz at the beginning of the next class period. This gives students a chance to review their work and refine their understanding of the material without having to wait for their homework to be graded. The quiz prompts them to take it seriously. The quiz starts at the beginning of the class period and lasts for 10 minutes. Once the quizzes have been collected, a solution is written and discussed. The intention is that the quiz question is comparable to one of the questions on the upcoming midterm exams. After 4 weeks of homework and quizzes, there is a 50 minute midterm exam with ten questions addressing those 4 weeks of material. There are 13 homework assignments over the course of the semester, so there are 3 midterms with an extra bit that ends up on the final. At the end of the course there is a comprehensive final exam.

The goal of this layout is to provide the student with prompts to take the material in small chunks and build on previous learning. Students first see the material in class and then have a homework that structures their reading in order to solve the problems. Help sessions are provided to help students over major hurdles in understanding. Once their homework has been submitted, they have an opportunity to study and compare their solution with the model solution provided by the instructor. Hopefully, any remaining large misunderstandings are addressed and students are refining their understanding in preparation for the quiz. At this point, students should have seen the material three times. The quiz provides another targeted feedback loop. The midterm then is over a digestible group of topics. Hopefully, the students are not learning the material at this point, but are reviewing it (leading to reinforcement and better retention over time.) Of course the final exam provides one more opportunity to revisit the material.

### 3.2 Blended differences

The blended class met every Wednesday and Friday with Monday considered as their flexible out of class study time. On Saturday, a reading assignment from Rosen's book on Discrete Math [6] would be posted along with the following study materials.

- **Worksheet:** The material was broken up into manageable sections with targeted questions that the student would complete as they read the material.

- **Solution:** All questions from the worksheet were answered.
- **Selected Videos:** Appropriate pre-existing videos over tougher topics were provided as a reference.
- **Online Quiz:** Students took an online quiz with approximately 10 multiple choice questions. There was no limit to the number of times that a student could take the quiz up until Tuesday evening. If they hit 80% on the quiz, the score would be upgraded to 100%. The goal was that students could take the quiz multiple times to help understand the material, but need not feel like they had to get a perfect score. After a student took the quiz, they could review their answers along with commentaries.

### 3.3 Assessments

Aside from the online quiz provided to the students of the blended section, the students in both courses had the same assessments. The homework sets and the associated paper based quizzes were shared. The midterm exams and the final were shared. As the instructor and grading policies were also the same, the ability to compare results between the sections was enhanced.

## 4 Results For the Final (Where we ended up.)

In all cases, to determine if there was a statistically significant difference in averages an unpaired t-test was performed. Since the number of students was small (21 in the blended section and 23 in the traditional section) the use of a t-test was appropriate [2]. While the scores were associated with individuals, the analysis only looks at group assessments, so pairing was not appropriate. Finally, as is often the case, a p-value of 0.05 or less was considered to be statistically significant. [2][4]

Table 1 shows the average score (out of 10) for each of the question on the final exam in the order that they were covered in the course. As can be seen, both sections performed comparably on the early material, with the blended section doing better on two questions. This indicates that for those topics the students in the blended section were able to overcome any setbacks due to the nature of the blended course. (See next section.)

It is the material on proofs where we see a difference <sup>1</sup> that rose to the level of statistical significance with the students in the blended section performing poorly. Both groups performed equally poorly on structural induction, which is a subject that prior students have also struggled over. The remaining topics

---

<sup>1</sup>Proof by contradiction has a p-value of 0.0407 and proof by induction has a p-value of 0.0411.

again show comparable results, but with a slight edge for the students in the traditional section.

I offer the following explanation for the difference in performance on the proofs material. The proof materials require significant use of non-intuitive logic and reasoning. While there are some patterns that students can use, it is harder to push that material off onto the students to learn by themselves. An interactive presentation of a proof beats reading about the proof and answering questions. On top of those issues, it is also more difficult to come up with good online multiple choice quizzes that illustrate proofs. The students in the blended section got behind on this material and never caught up. If I were to do a blended version of Discrete Math again, I would consider reorganizing the material so that proof by induction was in class only.

Table 1: Final Exam Results by Question Topic

Topic	Blended		Traditional	
	average	standard deviation	average	standard deviation
Summation	8.95	1.28	8.48	2.13
Predicate Logic	8.38	1.02	8.09	1.88
Logical Equivalence	9.19	0.98	9.48	0.95
Sets	8.81	1.75	9.30	1.11
Proof by Contradiction	7.71	2.74	9.04	1.22
Proof by Induction	6.48	3.23	8.39	2.79
Big-O Proof	6.62	2.76	7.43	2.98
Structural Induction	6.60	2.48	6.30	3.05
Counting and Recurrences	9.40	0.74	9.61	0.81
Counting	7.76	1.26	8.39	1.76

## 5 Results on Course Aggregates (How we got there.)

Table 2 shows the performance as a percentage on each of the assessments along with a p-value. Overall, the aggregate results show only one statistically significant difference between the two sections. The collection of twelve quizzes has a p-value of 0.0063. It is hard to make strong conclusions as only three of the individual quizzes (Logical Equivalence, Predicate Logic, and Sets) exhibited a statistically significant difference. On the other hand, for every quiz but one, the average over the blended section was lower than that of the traditional section. It is not unreasonable to propose that the online materials did not prepare the students in the blended section as well as the face-to-face presentation. This led to a 10% deficit on the homework (which was not quite significant) that carried over into the quizzes. The students in the blended

section were able to reduce the deficit to around 5% for the midterms, but did not fully catch up. While close to parity was achieved for most of the topics on the final, there were still areas where the deficit remained.

Table 2: Course Results by Assessment Type in Percent

Assessment type	Blended average	standard deviation	Traditional average	standard deviation	P-value
Homework (13)	68.94	13.27	78.29	17.11	0.0507
Quizzes (12)	66.63	10.61	76.02	11.01	<b>0.0063</b>
Midterm 1	79.73	8.67	83.28	13.65	0.3139
Midterm 2	81.71	9.92	86.85	11.02	0.1128
Midterm 3	78.93	10.38	82.98	11.70	0.2331
Final Exam	79.91	9.87	84.52	10.28	0.1373

## 6 Caveats

As with any work that is based on small numbers, it is hard to get statistically significant results. In this particular case, the differences in scores between the two sections could merely be due to chance in the distribution of students across the two sections. Another possibility that could not be accounted for was that there was an element of self selection when students signed up for the sections. Regrettably, it was not possible to do random or matched assignment of students to the two sections.

Another issue that could have affected the results is that the exams were administered sequentially for the two sections. The traditional section took the quizzes and midterm exams at 10am and the blended section followed at 11am. There was potentially information about the exam conveyed from one section to the next. The affect of this is mitigated since the exams are about application of knowledge/techniques rather than memorization. Knowing that there was going to be a certain kind of problem would not be as helpful. Especially, as students can make well informed guesses as to the kinds of problems that will be on the exam. If this did have an effect on the results, the expectation is that it would improve the scores on the blended section (which in general were lower.)

For the final exam, the order of administration was switched with the blended section taking the final two days before the traditional section. Aside from two of the proof questions, the results on the rest of the exam were comparable and did not indicate a large effect due to order.



## 7 Student Reactions

Some of the students in the blended section did not realize that it was blended until after classes started or did not understand what that entailed until later in the course. A few students expressed dissatisfaction with not getting the teaching time they paid for. Getting the right kind of students and managing their expectations is an important part of creating a blended course. One thing that the students really liked were the online quizzes with unlimited attempts and full credit at 80%.

## 8 Conclusions

The results of this experiment are mixed. The data suggest that the students in the blended section did not grasp the material as well as their counterparts in the traditional section. In particular, some topics did not fit as well in a “learn on your own” model as compared to an interactive approach. Proofs are complicated and benefit from the additional time spent in a classroom.

On the other hand, most of the results did not reach the level of statistical significance, though some were close. The results on the final exam could be considered encouraging (aside from the two proofs questions). There were topics where the students in the blended section caught up with their peers in the traditional section. With careful organization and monitoring, a blended Discrete Math course could provide similar outcomes as a traditional course.

## References

- [1] Josh Bersin. Use of moocs and online education is exploding: Here’s why. *Forbes*, July 6, 2016.
- [2] R.C. Blair and J. J. Higgins. A comparison of the power of wilcoxon’s rank-sum statistic to that of student’s t statistic under various non normal distributions. *Journal of Educational Statistics*, (5):309—334, 1980.
- [3] Clement C. Chen and Keith T. Jones. Blended learning vs. traditional classroom settings: Assessing effectiveness and student perceptions in an mba accounting course. *Journal of Educators Online*, 4(1), Jan 2007.
- [4] Michael P. Fay and Michael A. Proschan. Wilcoxon-mann-whitney or t-test? on assumptions for hypothesis tests and multiple interpretations of decision rules. *Statist. Surv.*, (4):1–39, 2010.
- [5] MJ Israel. Effectiveness of integrating moocs in traditional classrooms for undergraduate students. *International Review of Research in Open and Distributed Learning*, 16(5), 2015.
- [6] Kenneth Rosen. *Discrete Mathematics and its Applications*. McGraw Hill, 7th edition, 2012.

# DIVAS at Three: Image Processing Outreach\*

*Mark M. Meysenburg<sup>1</sup>, Tessa Durham Brooks<sup>2</sup>,*

*Erin Doyle<sup>2</sup>, Raychelle Burks<sup>3</sup>*

*<sup>1</sup> Department of Computer Science*

*<sup>2</sup> Department of Biology*

*Doane University*

*Crete, NE 68333*

*{mark.meyenburg,tessa.durhambrooks,erin.doyle}@doane.edu*

*<sup>3</sup> Department of Chemistry*

*St. Edward's University*

*Austin, TX 78704*

*rburks@stedwards.edu*

## Abstract

The DIVAS (Digital Imaging and Vision Applications in Science) project addresses workforce challenges in science, technology, engineering, and mathematics by creating a pedagogical and programmatic “on-ramp” that empowers natural science majors to engage in authentic computational problems as members of skilled, professional teams. We summarize our findings of the program’s effectiveness after three years of implementation.

## 1 Introduction

Computation is an increasingly necessary skill in the workforce for students studying biology and chemistry, and it has been shown that early introduction to computational applications with frequent practice makes graduates better

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

able to take advantage of training resources and collaborate with computer scientists and mathematicians; see, for example Christensen, et al. [2]

Motivated by these realities, in the Doane DIVAS (Digital Imaging and Vision Applications in Science) project [6], we are developing, using, and testing instructional practices and curricular innovations to engage and train science, technology, engineering, and mathematics (STEM) undergraduate students at Doane University and St. Edward's University in computing, related to image processing, to benefit the STEM workforce of tomorrow. In the project, we address STEM workforce challenges with our pedagogical and programmatic "on-ramp" that empowers natural science majors to engage in computational problems as members of skilled, professional teams. We have developed and are testing institutional practices and curricular innovations that are engaging and training STEM undergraduate students in Python programming, through computer vision and image processing, to address relevant scientific questions.

The objectives of the DIVAS project are 1) explore the effectiveness of image capture and analysis as a computational application, utilized in introductory biology and chemistry courses, to increase self-efficacy in utilizing computers to solve a problem; 2) explore the effectiveness of coding workshops on DIVAS Scholar attitudes toward computation, career path, and their ability to demonstrate effective computational thinking; 3) measure impacts of pair programming projects and code reviews, and professional development seminars on students' self-efficacy and ability to apply computational skills; and 4) investigate the impact of curricular and co-curricular interventions in computation on student preferred and actual career path.

We recruit primarily first-year students by visiting introductory natural science courses in the fall semester. Once accepted, DIVAS scholars enter the program with a one-credit seminar (DIVAS Seminar I) in which they learn coding basics, conduct an image exploration, and gain professional exposure. Then, in the early summer, the scholars participate in a week-long coding workshop. The workshop includes two days of elementary bash, git, and Python programming instruction, and three days of image processing instruction. Scholars then complete two pair programming projects over four weeks and conduct an optional three weeks of summer research. Scholars mentor the next cohort and learn code repository maintenance as well as the basics of grid computing in the DIVAS Seminar II course in the spring of the next year. Students enrolled in biology and chemistry courses may also be exposed to one or two class session image processing modules.

## 2 Interventions

The various interventions of the DIVAS project are described in more detail below.

Beginning DIVAS scholars participate in a professional development seminar (DIVAS Seminar I) during the spring of their first year in the program. Taught by DIVAS faculty, the class meets once each week for fifty minutes and serves as an introduction to the DIVAS project. This seminar includes an introduction to images and images as data, professional explorations designed to showcase careers that use computation and computational thinking, and a brief introduction to coding in Python, through an online platform for learning bioinformatics and programming through problem solving, Rosalind [8].

Two course modules, of one to two class sessions each, were designed for introductory chemistry (CHM 126) and an upper-level human physiology course (BIO 356). Both modules asked students to collect and analyze image data relevant to the course learning objectives. These interventions may or may not be experienced by the DIVAS scholars cohort, as they take place in general audience courses.

In the introductory chemistry module, students placed droplets of water on to various surfaces. They took a picture of a droplet on each surface and analyzed the angle between the surface and the water droplet to measure the surface's hydrophilicity.

In the human physiology course, students took pictures of a subject's eye before and after a flash of light (delivered by the camera). The change in pupil diameter was used as a measure of autonomic nervous system function.

Following the end of the spring semester, DIVAS faculty teach an intensive, five-day "bootcamp" workshop to introduce the DIVAS scholars and other interested undergraduate students, graduate students, faculty, and other professionals to the basic tools required to use simple image processing / computer vision in scientific research. To date, we have taught the workshop twice at Doane University and once at St. Edward's University in conjunction with the DIVAS project, and once at the University of Nebraska-Lincoln as an alpha Data Carpentry workshop.

The first two days of the workshop focus on basic computing and programming skills: an introduction to the bash shell, an introduction to git, and an introduction to Python programming. The last three days of the workshop cover how to use the OpenCV library to perform image processing tasks. This portion of the workshop currently covers image basics, image representation in OpenCV, drawing on images and bitwise operations, creating histograms, blurring images, thresholding, edge detection, and finding and manipulating contours.

The first two days of the workshop are modeled after the existing bash

/ git / Python workshops authored by Software Carpentry [4], and is freely available online [3]. The Image Processing portion is also written as a Carpentries lesson, and is currently available in beta format worldwide via the Data Carpentry site [1]. The Image Processing workshop is undergoing a revision to use the easier-to-install Scikit-Learn computer vision libraries rather than OpenCV; after these revisions are complete, the workshop will be taught by Data Carpentry instructors as an official Data Carpentry beta workshop for the first time at the University of Arizona, in January 2020.

At the end of the coding workshops, scholars were presented with two projects, one with a colorimetric focus and one with a morphometric focus. For each project, the group was divided into pairs. After discussing the questions that the images presented could address, pairs split out to develop code to extract relevant data from the images to address those questions. At the end of the first week of the project, teams converged for a code review. Each team presented the code they had developed while the rest of the group read and annotated the code so they could ask questions and help work through issues. Teams identified their goals for the next week and then split off again to further develop their code. A code review at the end of the second week was utilized to share progress and determine additional steps to improve or finish the project. This process was repeated with a different project in the following two weeks.

Following the pair programming project, the DIVAS scholars had the option of completing three or more weeks of independent research in years two and three; the research component was required in year one. Research projects were either selected from those available in faculty laboratories or were designed by the scholar.

As of this writing, the DIVAS Seminar II course has been taught two times, once each for the first and second cohorts of DIVAS scholars. In the first Seminar II course, the scholars cleaned up and organized their summer research project code, developed online portfolio entries regarding their DIVAS experiences, prepared and presented local conference presentations on the program, and helped improve the coding workshop for its second use. Based primarily on student feedback, the second Seminar II course was modified to include more challenging academic content. In this course, additional material included an introduction to parallel programming using Python and OpenMPI, including a final project to create a “Burning Ship” fractal image using the Doane University supercomputer, Onyx.

### 3 Measurements

During the DIVAS project, we measured the effectiveness of the various interventions with several established and newly developed instruments, described

in more detail below.

We measured participating students’ perceived self-efficacy in computing and their intention to pursue a career path involving computing via on-line Qualtrics surveys using questions developed by Shell, et al. [9] and Peteranetz, et al. [7] Questions in the survey asked how much students know about careers involving computer science applications, programming, or computational thinking; where to find information about such careers; their general knowledge of computational thinking; their perceived ability to use computational algorithms to solve problems in their scientific domain; and more. The questions related to career paths required answers on a four- or five-point Likert scale. The questions related to self-efficacy required answers on a 100-point scale, with higher values representing more confidence in self-efficacy for a particular item. Participants took the survey before and after the major interventions in the project.

We measured participating students’ computational thinking skills using a rubric of our own design, based on definitions from the ISTE and CSTA [5], among others. To attempt to evaluate their thinking in a larger context than simply in programming, we organized our rubric around the first four phases of the RADIS (Recognize / Analyze / Design / Implement / Support) framework.

## 4 Results

Based on data gathered from the instruments described above, our three year results show that multiple interventions have had significant positive impact on students self-efficacy and interest in using computing in their future careers, and a positive impact on certain computational thinking skills. These results are presented below, organized by the intervention type.

Self-efficacy and career path surveys were delivered pre- and post-intervention for each of the three years. Since survey completion was voluntary, not all students who completed the course modules completed surveys. The chemistry module results are shown in Table 1. Self-efficacy gains were strong for both modules in year 1, but were not seen in years 2 and 3. Career path gains were apparent in the chemistry module for both years 1 and 2, but not year 3.

Table 1: Self-efficacy and career path gains the CHM 126 module. Effect sizes (Cohen’s-D) for each year shown; \*,  $p < 0.05$ , \*\*,  $p < 0.01$ .

	Year 1	Year 2	Year 3
Self-efficacy	1.27**	ns	ns
Career path	1.09**	0.52*	ns

Results for the BIO 356 module (Table 2) show that there were significant gains for self-efficacy in year one and significant gains for career path in year two, while the other gains were not significant.

Table 2: Self-efficacy and career path gains the BIO 356 module. Effect sizes (Cohen’s-D) for each year shown; \*,  $p < 0.05$ , \*\*,  $p < 0.01$ .

	Year 1	Year 2	Year 3
Self-efficacy	1.91**	ns	ns
Career path	ns	0.33*	ns

Self-efficacy and career path data were gathered pre- and post-seminar. Computational thinking skills were assessed using our CT rubric to score responses to problems in which scholars were asked to ‘program’ a cup-stacking robot to build a particular cup arrangement.

We saw significant gains in self-efficacy or career path each year of the program (Table 3). Combining the three years, we found significant positive effects of the intervention on both self-efficacy and career path. We did not find significant gains in computational thinking in any year of the program, nor did we find effects in the aggregate data.

Table 3: Self-efficacy and career path gains in DIVAS Seminar I. Effect sizes (Cohen’s-D) for each year of the program and the three years combined (Cum.) shown; \*,  $p < 0.05$ .

	Year 1	Year 2	Year 3	Cum.
Self-efficacy	2.38*	ns	0.74*	0.35*
Career path	ns	1.38*	ns	0.63*

Another measure of the effectiveness of the DIVAS Seminar I course can be found in the results of the Doane course evaluations, which are administered using the IDEA survey form. Scholar perceptions of learning gains as reflected in IDEA survey data were analyzed for the three years of the program. The objective “Acquiring skills in working with others as a member of a team” was among the highest rated, with an average score of  $4.45 \pm 0.68$  out of a possible high of 5. The other learning objective with this same high score was “Learning appropriate methods for collecting, analyzing, and interpreting numerical information.”

Overall, DIVAS Seminar I was found to be effective in improving the self-efficacy of scholars toward computing as well as positively influencing their intended career path. We also found that it impacted the way scholars viewed

how computational work is done as shown in IDEA survey responses.

Self-efficacy and career path data were gathered pre- and post-workshop, and computational thinking skills were measured at the end of the workshop using our CT rubric, as applied to the code students wrote for the final workshop challenge problems. At the end of each workshop day, we surveyed participants, asking them to rate the degree of the day’s materials they felt they mastered. We saw a significant improvement in self-efficacy in years one through three combined (Cohen’s-D = 0.57,  $p < 0.01$ ). There were no significant changes in intended career path. In year 1, we saw a significant improvement in the ‘Implementation’ criteria for CT skills (Cohen’s-D = 0.96,  $p < 0.05$ ). As of this writing CT results for years two and three are still being calculated.

In the end-of-day surveys, we found a high average degree of perceived mastery for the Python / Bash / git portion of the workshop, and then a drop for the first two days of the computer vision portion (Table 4). We believe this is due to the increased complexity in the subject matter. By the third day, the perceived degree of mastery grew, as participants were able to use their newfound skills to successfully complete the challenge questions.

Table 4: Participant responses to the question “What percentage of the day’s material do you feel you have mastered?” for each day of the Coding Workshops.

	Day 1	Day 2	Day 3
Python Intro	75.9%	76.3%	-
Image Processing	63.3%	63.0%	72.2%

Self-efficacy and career path data were collected at the end of the summer (after pair programming and summer research, if the scholar participated). The code generated in the pair programming projects and summer research were also scored using our CT rubric.

We did not see any significant gains in self efficacy or career path at the end of the summer. We are not surprised by this result. Coming in to the summer, student self-efficacy was already high compared to where they were at the beginning of the program. At the beginning of the program, the average baseline self-efficacy score across the three years was  $655 \pm 217$  out of a total possible 1000 points. By the end of the coding workshop (before pair programming), average self-efficacy had grown to 837 and the standard deviation had narrowed to 178. At the end of the pair programming projects and summer research, self-efficacy had increased on average to  $881 \pm 176$ . Overall, self-efficacy moved in a positive direction over these interventions, though not significantly. This may be expected since students were already close to the



ceiling of the instrument at the end of the coding workshop. In other words, it is expected that we will see diminishing returns in self-efficacy with each subsequent intervention tested.

We saw significant gains in self-efficacy in both years of DIVAS II, and a significant gain in career path scores in year two (Table 5). Responses on the IDEA course surveys showed, similar to DIVAS Seminar I, that students perceived the largest gains in “Learning appropriate methods for collecting, analyzing, and interpreting numerical information” ( $4.33 \pm 0.52$ ). Interestingly, scholars responded equally positively to the statement “My background prepared me well for this course’s requirements” ( $4.5 \pm 0.55$ ), reflecting the gains in self-efficacy we saw in the previous survey data.

Table 5: Self-efficacy and career path gains in DIVAS Seminar II. Effect sizes (Cohen’s-D) for years one and two shown; \*,  $p < 0.05$ ; \*\*,  $p < 0.005$ .

	Year 1	Year 2
Self-efficacy	1.2*	
Career path	1.5*	3.93**

Thirteen of the 17 DIVAS scholars from the three years of the project (76%) were female, a significantly higher percentage than the total percentage of females in the majors represented in the project or in STEM overall. A strong testament to the success of the program can be seen in how DIVAS scholars have persisted in coding in various ways, incorporating skills gained in the DIVAS project into their academic careers, extracurricular activities, and career planning. One Scholar majoring in Biology declared a Software Development Minor, a second Biology major switched to a Bioinformatics major, and two scholars have taken non-required electives that emphasize computational skills. One Scholar participated in an external REU program in Computational and Systems Biology, and several DIVAS scholars have continued research projects that incorporate coding or computational thinking. Three DIVAS scholars have gone on to work as peer tutors for Doane’s Center for Computing in the Liberal Arts. One additional student who participated in both the coding workshop and paired programming is pursuing a Ph.D. in Complex Biosystems.

## 5 Discussion

Overall, DIVAS scholars showed significant increases in self-efficacy towards computing when comparing self-efficacy scores for scholars across all three cohorts at the end of the summer research experience with initial pre-test scores (Cohen’s D-effect size= 1.05,  $p = 0.001$ ). Gains occurred primarily through

Seminar I (pre-post Cohen’s D-effect size = 0.35,  $p < 0.05$ ) and the coding workshop (pre-post Cohen’s D-effect size = 0.57,  $p < 0.01$ ). An additional increase in self-efficacy of 44 points was observed when comparing scores at the end of the coding workshop to scores at the end of the summer research experience, although not significant ( $p = 0.167$ ). Average self-efficacy scores prior to beginning the DIVAS program and at the end of each intervention are shown in Figure 1.

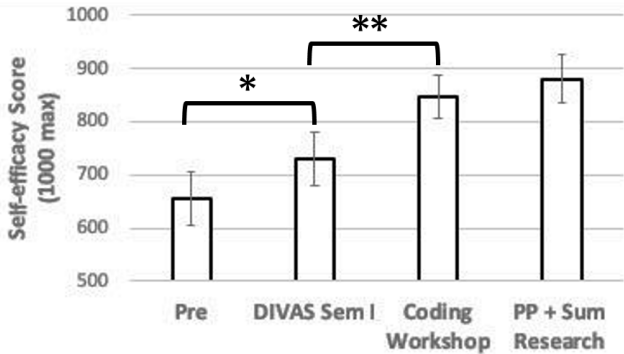


Figure 1: Average self-efficacy scores over three years of DIVAS scholars.

The impact of the DIVAS program on scholars’ intended career paths was more difficult to quantify. Although the career path survey did not show significant gains when comparing scores from the end of summer research to the initial pre-test ( $p = 0.120$ ), significant gains were observed between the pre-test and the end of Seminar I (Cohen’s D effect size = 0.63,  $p = 0.020$ ). This result is not surprising, as Seminar I directly addresses potential careers that use images and coding, whereas the other interventions do not. Additionally, scholars were observed to become ‘warmer’ or ‘colder’ to a career utilizing computing. This effect is apparent in the increased standard deviation in career interest scores post intervention which start at  $\pm 2.3$  after Seminar I and grow to  $\pm 3.02$ , then  $\pm 3.46$  after coding workshops and pair programming/summer research. We see this as a healthy progression, especially because scholar self-efficacy grows throughout the program.

## 6 Conclusion

The use of images in the project seems to be a very effective “hook” to encourage natural science students to become interested in using computation in their research. Our results support our intention to continue to use all of our inter-

ventions in the next phase of the project. In this next phase, we intend to test the impact of our interventions on students at a wider selection of universities.

## 7 Acknowledgments

The work is supported by the National Science Foundation IUSE Exploration & Design: Engaged Student Learning program, under Grant No.: 1608754.

## References

- [1] Data Carpentry. Image processing with Python, 2019. <https://datacarpentry.org/image-processing/>.
- [2] Ken Christensen, Dewey Rundus, Hiroshi Fujinoki, and Darrel Davis. A crash course for preparing students for a first course in computing: Did it work? *Journal of Engineering Education*, 91(4):409–413, 2002.
- [3] Erin Doyle. Python Intro, 2019. <https://eldoyle.github.io/PythonIntro/>.
- [4] The Software Carpentry Foundation. Our lessons, 2019. <https://software-carpentry.org/lessons/>.
- [5] CSTA / ISTE. Operational definition of computational thinking, August 2011. <http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf>.
- [6] Mark Meysenburg, Tessa Durham Brooks, Raychelle Burks, Erin Doyle, and Timothy Frey. DIVAS: Outreach to the natural sciences through image processing. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education, SIGCSE '18*, pages 777–782, New York, NY, USA, 2018. ACM.
- [7] Markeya S. Peteranetz, Abraham E. Flanigan, Duane F. Shell, and Leen-Kiat Soh. Perceived instrumentality and career aspirations in CS1 courses: Change and relationships with achievement. In *Proceedings of the 2016 ACM Conference on International Computing Education Research, ICER '16*, pages 13–21, New York, NY, USA, 2016. ACM.
- [8] Rosalind. Rosalind | problems, 2019. <http://rosalind.info/problems/list-view/?location=python-village>.
- [9] D. F. Shell, M. P. Hazley, L. K. Soh, L. Dee Miller, V. Chiriacescu, and E. Ingraham. Improving learning of computational thinking using computational creativity exercises in a college CSI computer science course for engineers. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, pages 1–7, Oct 2014.

# A Course-Based Undergraduate Research Experience (CURE) in Computer Science: An Experience Report\*

*Fahmida Hamid*  
*Computer Science*  
*Grinnell College*  
*Grinnell, IA 50112*  
*fahmida.hamid@gmail.com*

## Abstract

This article demonstrates a pedagogy of a research-focused Computer Science course for undergraduates in a liberal arts environment. The long term benefits of Course-based Undergraduate Research Experiences (CUREs) in different STEM fields are the driving forces for modeling such a course. The course balances a research-focused, semi-supervised, active learning experience and a traditional lecture-focused, supervised classroom-environment. This article summarizes the current semester's student experiences and suggests possible scope for improvements.

## 1 Introduction

Course-based undergraduate research experiences (CUREs) offer an effective way of integrating research into an undergraduate science curriculum and extending research experiences to a large, diverse group of early-career students [2, 5]. Students who participate in CUREs develop content knowledge and technical skills specific to the area of research [8]. An increasing number of well-designed and well-controlled studies show that CUREs can influence students learning, development, and educational and career trajectory[3, 4, 7]. Several studies [2, 5] discuss about research-focused courses in different STEM

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

fields: Biology, Chemistry, Medicine, etc. In the Computer Science (CS) discipline, though there have been some practice already in different organizations, a model for such course has not yet been reported to the academic community. This article details a model of a CS course following the principles of course-based undergraduate research experience.

In this paper, section 2 describes the course structure and activities, section 3 states student experiences and outcomes, section 4 makes suggestions about extending the goals and involvements beyond the classroom, and section 5 draws conclusion. Due to the page-limit constraints, detail definition and logical structure of CURE, short and long term benefits of such courses are omitted. Interested readers might find relevant information in article [2].

## 2 CURE in Practice for CS Students

In Computer Science, research areas such as Natural Language Processing, Information Retrieval, Machine Learning, Artificial Intelligence, Data Science, etc. do not always need specialized equipments/hardwares other than a modern computer-equipped lab space (teaching-lab) for deploying a full research-focused course. The experience report is based on such course in Information Retrieval(IR) track. The next couple of sections outline the structure and activities of the course.

### 2.1 Course Layout

The course objective is to provide hands-on experiences with several existing software tools through the programming assignments and to train them to develop critical thinking and analytical skills through written assignments. In order to offer a balanced way of learning new techniques and applying them for conducting research, the course is divided into two main phases:

- Supervised Learning Phase
- Semi-supervised Learning (Research) Phase

The next two sections introduce the major activities during these phases.

### 2.2 Supervised Learning Phase (Week 1 ~ Week 7)

The activities of this phase are mostly **lectures** highlighting and demonstrating important concepts from textbook readings. When appropriate, external readings can be used. Students are asked to complete written assignments (individual works), in-class tasks and programming assignments (group works).

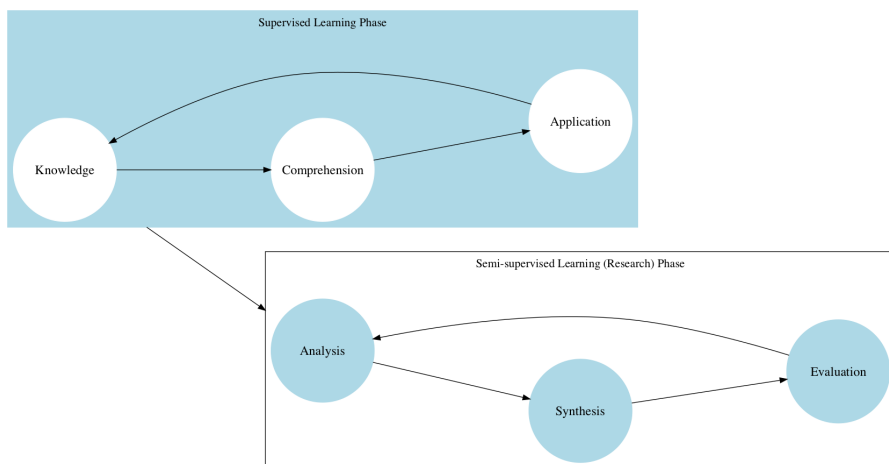


Figure 1: A revised Bloom's Taxonomy [1] for a research-focused course.

### 2.2.1 In-Class Tasks

Providing students with models and worked examples can help them learn to solve problems faster [6]. During the second hour of every class, students either run/test small experiments (with code provided by the instructor) or implement/modify a partially solved model to get a better understanding.

### 2.2.2 Programming Assignment

The programming assignments are designed with the following goals in mind:

- Students should have a lot of freedom in choosing the layout and the techniques while implementing the solution.
- Students should know about the standard evaluation techniques and be able to design new scales of measures.
- Students should write detailed reports for the experiments.
- Students should be comfortable using different standard datasets so that they can test their work with larger collections.

### 2.2.3 Written Assignment

Students are asked to submit written reports (summarize the concepts and answer few related questions) about published articles (chosen by the instructor) during the first and the fourth week. The goal is to improve their analytical and writing skills on technical matters.

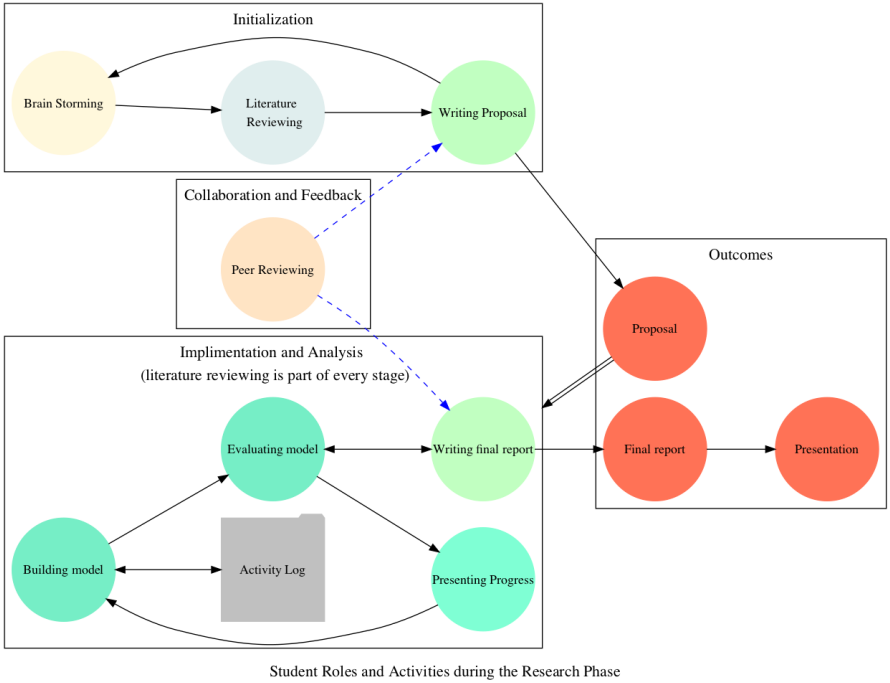


Figure 2: Major Student Activities during the Research Phase

Programming assignments and the template for written assignments can be found in appendix A.

### 2.2.4 Term Exam

The supervised learning phase ends with an open-book, two-hour long exam where students are asked to solve analytical questions on covered topics.

The covered topics on this phase can be found in appendix B.

## 2.3 Research Phase (Week 8 ~ Week 14)

The first day of this phase is a full lecture led by the instructor introducing the students to different research sub-areas and possible research problems in each of those areas. At the end of the session, students form groups on their own and start talking about possible research problems that sound interesting to them.

The instructor takes a passive role (monitor and guide) from now on. During the eighth week, students spend their time on writing research proposals which are approved after two trials of presentation (one informal and one formal) in front of the class.

As **literature reviewing** is an important part of research, each student-group presents two articles (relevant to their research idea) as a basis of their work. They also submit a weekly plan: approximate amount of spent time per week and tentative weekly goal. Asking them to budget time is important as this helps them find a balance between their ambitions and achievable goals within time and other constraints. Based on the demand of the research project, students may need to find relevant datasets or state a plan for creating datasets.

### Template/Guideline for Writing A Research Proposal

**Title:** Find an interesting and representative title.

**Abstract:** Highlight the basic idea and its importance.

**Project Description:** With few literature reviews, explain your idea, the scope, the broader impact of your project.

**Datasets:** Cite proper existing datasets (if required) that you plan to use or explain how you plan to construct one for your work.

**Preparation:** Report the skills and knowledge that you think will be useful.

**Outcomes & Timeline:** Highlight some final expected outcomes (a paper, poster, software, API, etc.)

**References:** Cite relevant research works (at least 5 to begin with).

To make the students accountable for maintaining the quality of their work, every student also work as **anonymous reviewer** for other students' projects. The final work is submitted as a paper following standard formats commonly found in most technical IR articles: title, abstract, introduction, related work, dataset, methodology, result analysis, and conclusion/future directions.

### Review Template

1. Summarize the project-idea in your own words.
2. Rate at the scale of 1 to 10 (1 being the worst and 10 being the best):
  - Is the research question clearly stated?
  - Have they explained clearly why their idea is interesting and impactful to some parties?
3. State some strengths and some weaknesses of the project idea.
4. Have they clearly talked about the evaluation mechanism and datasets (if needed) for their project?
5. (Optional) Provide some suggestions to the team.



Careful attention is paid on their **writing quality**: explaining the methodology and contribution, citing relevant works, acknowledging peers for their comments and suggestions, etc. The 14-week long activity ends with formal 20-min presentation by each group in a seminar hall where students from the school will be invited to attend.

## 2.4 Time Commitment

The students meet twice a week for 2 hours in a teaching lab. It is expected from the students that they would complete assigned readings (approximately 1~2 hours) for the class and complete the assignments outside of class-meetings (approximately 4~6 hours). Overall, students are expected to spend approximately 7~10 hours per week for the course.

## 2.5 Grading/Assessment

50% of the course grade is allocated for the research phase. Clarity of thinking, efficiency of implementation, depth of analysis, structure and readability of the project report, and final presentations are the determining factors for achieving that 50% score. The rest of the course-grade is distributed on regular programming assignments, written assignments, class participation, and one term exam. A complete course syllabus and related materials can be found at authors github repository: <https://github.com/FahmidaHamid/CCSC2020CP>

# 3 Students Experiences

Given the research sub-areas as {Keyphrase extraction and Summarization, Learning to rank, Question-Answering, Recommender Systems}, the running student projects are the followings:

### Student Projects

**Information Desk:** to create a powerful Question-Answering system, leveraging the power of Generative Adversarial Networks (GAN) and the World Wide Web (WWW).

**Teaching Machines:** to Reason on texts, to build a Question-Answering model with complex questions (how or why) and eventually understand the sequential facts embedded in the documents to produce the answer.

**Keyphrase Extraction of YouTube Video Transcripts:** to outline the contents and to create tags to improve search results.

**A Study of Keyphrase Extraction on Celebrity Tweets:** to create a celebrity-tweet dataset and apply unsupervised keyphrase extraction techniques on it.

The first two groups are studying several Machine Learning techniques for implementing automatic answer generation in different situations. The third group is using YouTube’s API for building their own “transcript dataset” and applying meta-information to build a Naive-Bayes Classifier for extracting meaningful phrases. The fourth group has manually created a dataset of “Monthly Celebrity-tweets” and will be using an unsupervised technique to automatically extract keywords. The first and the third group will use human annotators to help create gold-standards for evaluating their system’s performances. The second group uses a dataset published by Facebook that comes with standard answers. The group plans to extend the model from producing short phrased answers to a complete sentence.

The major outcomes of the course from the student perspectives are:

- (a) To be able to write a research proposal and a technical paper.
- (b) To be able to think independently and discover a problem (area of concern)
- (c) To be able to write constructive criticisms of other’s work.
- (d) To plan for publishing the work as a student paper/poster to a proper venue.

## 4 Extending the Scope and Possibilities

If planned with enough time and resources, several other possibilities can be included to add more value and learning experience:

**Invited talks:** Scholars in similar research tracks can occasionally give invited-talks. In some cases, students can remotely join at different events (e.g. ACM TechTalk) and thus flip a lecture on similar/same topic.

**Student talks:** Offering bonus-points for actively participating in some student clubs and presenting the research will help students build confidence and acquire more knowledge. This can motivate the students in audience seats for conducting research as well.

**Workshop/Poster presentations:** Students should aim for presenting their work to external events. Regional or national undergraduate research symposiums are possible suits for their work. “What will be the impact of my research?”, “Why is this problem still unsolved?” – these questions lead students into determining the scale of their work and not limit their achievements only to a good letter grade.

## 5 Conclusion

The goal of the course is to lead the students to the process of scientific discovery in a small scale. The first round of experiment at Grinnell College with eight senior students has been successful in several ways. Given more time and scope, in next trial, several modifications can be done: having an overlapping research and supervised learning phase, offering student-stipends for teams that can participate at poster/workshop events outside of the college, etc. In short, research-focused courses create a trend of involving undergraduates into research in an effective way. Faculties with different research interests should offer research-focused courses by rotation to engage larger and diverse set of students.

## References

- [1] Nancy E Adams. Bloom's taxonomy of cognitive learning objectives. *Journal of the Medical Library Association: JMLA*, 103(3):152, 2015.
- [2] Lisa Corwin Auchincloss, Sandra L. Laursen, Janet L. Branchaw, Kevin Eagan, Mark Graham, David I. Hanauer, Gwendolyn Lawrie, Colleen M. McLinn, Nancy Pelaez, Susan Rowland, Marcy Towns, Nancy M. Trautmann, Pratibha Varma-Nelson, Timothy J. Weston, and Erin L. Dolan. Assessment of course-based undergraduate research experiences: A meeting report. *CBE: Life Sciences Education*, 13(1):29–40, 2014.
- [3] M Kevin Eagan Jr, Sylvia Hurtado, Mitchell J Chang, Gina A Garcia, Felisha A Herrera, and Juan C Garibay. Making a difference in science education: the impact of undergraduate research programs. *American educational research journal*, 50(4):683–713, 2013.
- [4] Sylvia Hurtado, Nolan L Cabrera, Monica H Lin, Lucy Arellano, and Lorelle L Espinosa. Diversifying science: Underrepresented student experiences in structured research programs. *Research in Higher Education*, 50(2):189–214, 2009.
- [5] Jane L. Indorf, Joanna Weremijewicz, David P. Janos, and Michael S. Gaines. Adding authenticity to inquiry in a first-year, research-based, biology laboratory course. *CBE: Life Sciences Education*, 18(3), 2019. PMID: 31418655.
- [6] Barak Rosenshine. Principles of instruction: Research-based strategies that all teachers should know. *American educator*, 36(1):12, 2012.

- [7] P Wesley Schultz, Paul R Hernandez, Anna Woodcock, Mica Estrada, Randie C Chance, Maria Aguilar, and Richard T Serpe. Patching the pipeline: Reducing educational disparities in the sciences through minority training programs. *Educational evaluation and policy analysis*, 33(1):95–114, 2011.
- [8] Jack T. H. Wang. Course-based undergraduate research experiences in molecular biosciences-patterns, trends, and faculty support. *FEMS Microbiology Letters*, 364(15), 07 2017.

## A Sample Assignments

Programming Assignment 01

Allocated Time: 2 weeks

Goal: To implement one of the most common applications in IR

Problem Statement:

Build a search engine using your preferred programming language. Your code should be well documented. While implementing your engine, follow the basic building blocks of a standard search engine. Report the strengths and weaknesses of your model.

Server Side should do the followings: crawl at least 100 unique pages, parse them, create index, build web-graph by analyzing the links, and rank them (you may want to implement the page-rank algorithm). It is a good idea to design a topical search engine, like, crawl pages whose title contains a particular topic (e.g. sports). That way, your web-graph will be well-connected. Client Side should have a very simple interface to take input (text query) from the user, search in the index, and print the related web-links in some order. Bonus points will be allocated for designing interactive user-interfaces.

Programming Assignment 02

Allocated Time: 2 weeks

Goal: To find the latent features from the so-called unstructured and abnormally large collection of data.

Problem Statement:

The spam filtering problem is one kind of text categorization problem with the categories being spam and ham. The structure of email is richer than that of flat text, with meta-level features such as the fields found in MIME compliant messages. Researchers have recently acknowledged this, setting the problem in a semi-structured document classification framework. Several solutions have been proposed to overcome the spam problem. Among the proposed methods, much interest has focused on the machine learning techniques in spam filtering. They include rule learning, Naive Bayes, decision trees, support vector machines, or combinations of different learners. The basic and common concept of these inductive approaches is that using a classifier to filter out spam and the classifier is learned from training data rather than constructed by hand.

Your job is to choose two possible machine learning approaches to solve the problem and analyze their performances. Then you will report possible mechanisms to improve the current models you implemented.

Written Assignment

Allocated Time: 1 week

Study a given article "ABC" and answer the following questions:

- Summarize the article. (State the overall contribution of the article within 500 words.)
- Explain their hypothesis using the following structure: (assume a relevant structure M is given.)
- The authors used mechanism P and Q for evaluation. Suggest another relevant evaluation mechanism.
- State at least two cases where the stated hypothesis will fail (or, State some weaknesses).
- Find another problem where you think the hypothesis (solution) can be applied equally effectively.

B Covered Topics

Topics Covered
Boolean Retrieval Model
Search Engine Architecture
Web Crawler & Basic Text Processing Technique
Inverted Index & Query Processing
Search Result Interface & Link Analysis
Vector Space model
Probabilistic Information Retrieval
Language Models
Evaluation in IR
Discussion on possible Research Tracks
- Question-Answering
- Recommender Systems
- Keyphrase Extraction
- Learning to Rank

# LAGradebook: A Tool for Course-Level Comparative Learning Analytics\*

*Christopher Phillips and Jesse Eickholt*

*Department of Computer Science*

*Central Michigan University*

*Mt. Pleasant, MI 48859*

*{phill1cp,eickh1jl}@cmich.edu*

## Abstract

Learning analytics aims to collect and analyze data to improve learning outcomes. Several tools for learning analytics exist and have been successfully used in the context of higher education but their cost and related institutional challenges can be limiting factors. To complement larger, institutional efforts for learning analytics adoption, we have developed a tool to support smaller scale applications. The tool ingests raw, student assessment data and produces a rich, learning analytics enabled spreadsheet. The spreadsheet calculates descriptive statistics and relative performance and contains pair-wise correlation metrics of assessment items and individual student reports. The reports and comparative performance measures help students monitor their relative performance. The low-cost and course-specific nature of the tool supports increased access to learning analytics in more classrooms.

## 1 Introduction

Learning analytics (LA) centers on the idea that it is possible to use data, a focal point of our modern society, to help inform the decision-making process as it relates to the field of academia. More specifically, learning analytics makes “use of analytic techniques to help target instructional, curricular, and support resources to support the achievement of specific learning goals” [14]. The particular techniques used include summarizing, modeling and pattern recognition

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

and the applications of learning analytics can be descriptive, predictive, or prescriptive in nature. These efforts can take place at an institutional level (e.g., using assessment data from several courses, advising reports, entrance exams) or on a smaller scale (e.g., using data from a specific course of an instructor). A number of tools have been developed and evaluated to support learning analytics at the institutional level. Such tools provide a way for an interested party to engage with data and extract useful data regarding student success, academic efficiency, and student engagement with course materials. Some of the more prominent examples of LA tools include Check My Activity [6], Course Signals [2], ECoach [11] and Degree Compass [4]. The Check My Activity tool helps students keep track of how engaged they are with their learning management system (LMS) [6]. Students are then able to compare their use of the LMS to their peers' use of the LMS. This tool allows a student to recognize where they are fully engaged and where their engagement might be lacking. Purdue's Course Signals software is designed to help with student engagement and success and will alert instructors of at-risk students and allows the instructor to assign a green, yellow, or red light in the LMS [2]. At the University of Michigan, a tool called ECoach has been used to recognize when students perform better than expected in a course and then interviews them for tips, tricks, and advice [11]. The system will then provide this tailored advice to any future student that is struggling in the course. Degree Compass has been used by Austin Peay State School [4]. Rather than focusing on any specific course, Degree Compass evaluates and analyzes the whole degree program. By looking at statistics like course registration efficiency and time-to-degree completion this software aims to create the perfect course schedule for each student that is optimized for success. In addition to these research oriented tools, commercial products are also available and directly tie into learning management platforms such as Blackboard [3].

While many successful applications of LA systems have been documented in higher education, their adoption has been challenged by institutional factors and faculty reception. The size of institutional systems, the breadth of involvement of personnel across the institution and privacy concerns are often noted as challenges to institutional adoption of learning analytics [5, 9, 13]. Additionally, cost is arguably a concern when considering commercial products. Smaller scale institutional efforts to support learning analytics have focused on faculty. Common professional development paradigms such as workshops and faculty learning communities have been used to educate faculty on learning analytics [12, 10]. Such efforts may provide faculty with knowledge of learning analytics but not provide the needed tools.

It has been posited that the intention of learning analytics is transferability and that "analytic and predictive models need to be reliable and valid at the

scale beyond the individual course or cohort" [5]. Arguably, there are advantages to larger, more generalizable learning analytics tools (e.g., ECoach or Degree Compass) but cost or institutional adoption can be limiting factors. As a concurrent pathway for wider adoption of learning analytics, we propose techniques and tools that can be adopted and used at a smaller scale (e.g., instructor level). In line with this call, we have developed and present LA-gradebook. It is a tool that can be adopted at the instructor level to support descriptive and comparative learning analytics for instructors and students.

## 2 Design Considerations

In designing a tool to support instructor-based learning analytics, it was decided that the tool should be open source, community licensed and decoupled from any particular learning management system. An additional constraint imposed was that it should take as input raw assessment data from a text file. These choices eliminated cost concerns and the need for institutional IT support. Again, the aim of this work was to create an alternative to institutional adoption of learning analytics and so it was important that the adoption requirements and costs of our approach were as low as possible to support individual instructor adoption.

Of initial interest and inspiration was the Gradebook Calculator. This is a web-based tool provided by the DataLab at Carnegie Mellon University. It ingests raw gradebook data and identifies struggling students, finds unreliable assessment items and discovers challenging concepts for students [1]. We chose to avoid a web-based tool due to privacy concerns that instructors might have in uploading student data. We also wanted to help instructors relay the insights gleaned by learning analytics back to the students which was not directly supported by the Gradebook Calculator.

Generally speaking, what was desired was a means to ingest raw assessment data, perform learning analytics and generate student reports. Processing of the data needed to be performed locally and any required tools needed to be community licensed or commonly available in an academic setting. This led to the selection of Python and Microsoft Excel. A Python program would ingest raw assessment data from a text file and create a rich, learning analytics enabled Excel spreadsheet. This would allow the data to remain local and leverage tools that most have at their disposal (i.e., Python and Microsoft Office suite). The Excel spreadsheet would support the generation of reports and produce summary data that could be imported into a LMS, and as a result help instructors share insights with students.

The aforementioned design requirements (e.g., instructor level, local processing, platform independence) restricted the type of learning analytics that



could be performed. When data collection is restricted to one course or one instructor, the options for predictive analytics are limited and as a result a number of descriptive analytics were selected to support student comparisons. As an example of what is possible with descriptive analytics, consider the Check My Activity tool that allowed students check their LMS activity with those that performed above, below or at their performance category [6]. This type of information can be used to "nudge" students into action [7].

### 3 Implementation

A LAGradebook is constructed using an application written in Python and making use of the openpyxl module [8]. The application reads in a tsv or xls file containing assessment data and produces a Microsoft Excel compatible spreadsheet. The primary purpose of the Python application is to produce an interactive LAGradebook (i.e., the Python program builds and embeds into the spreadsheet the logic needed to perform calculations rather than perform the calculations and produce a static document).

Using the data provided, the Python application will create several sheets in the newly created workbook. The first sheet is an copy of the initial data used for creation of the workbook. The second sheet is the control sheet. This sheet allows for some customization, such as which assignments to include in future calculations. This sheet also provides baseline descriptive statistics, average and standard deviation. The average and standard deviation are calculated with Excel functions that have been custom configured in the Python source. The control sheet allows the user to choose whether a column (i.e., an assessment item) should be used in calculations for top performance and relative performance through the use of a drop-down selector in the sheet. The sheet initially checks "No" unless the column has previously been selected. This can be modified by the user at any time. The "Top Performance" sheet is where the user will see the top performers on the chosen columns. The user may alter the z-score used in this calculation to have a broad or narrow criteria. This sheet also provides information on how often each individual student has been identified as a top performer on all assignments to-date. The "Relative Performance" sheet provides information about each student's relative performance on the selected assignments. Instructors will be able to see whether a student is performing at, above, or below average, compared to their peers. The "Correlations" sheet calculates and displays the correlation coefficient for each pairing of assessments scored numerically. The final sheet is a "Student Report" sheet that contains summary information for each student. Page breaks are automatically inserted between students to facilitate distribution when printed. Again, it is important to note that these sheets are dynamic. As the user

makes changes on the control sheet (e.g., selecting or deselecting assessment items for top performance and/or relative performance), the respective sheets in the LAGradebook will automatically update. Figures 1 and 2 illustrate the Control Sheet and Relative Performance Sheet, respectively.

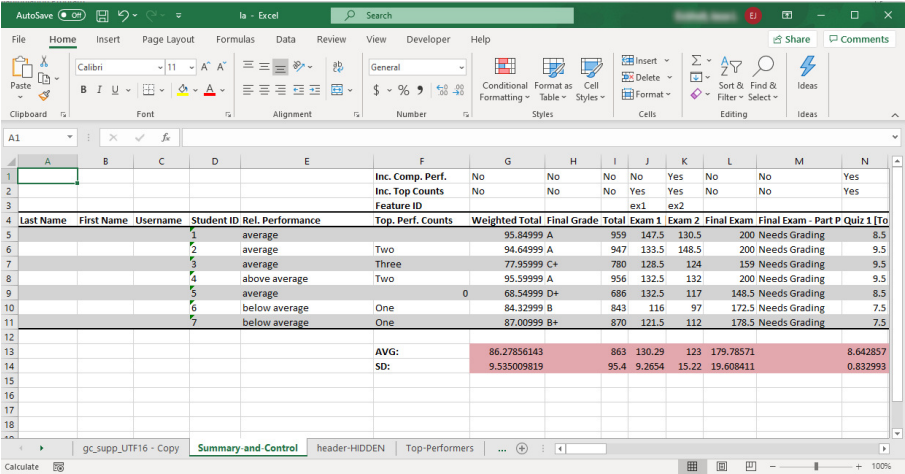


Figure 1: Sample Control Sheet generated by LAGradebook.

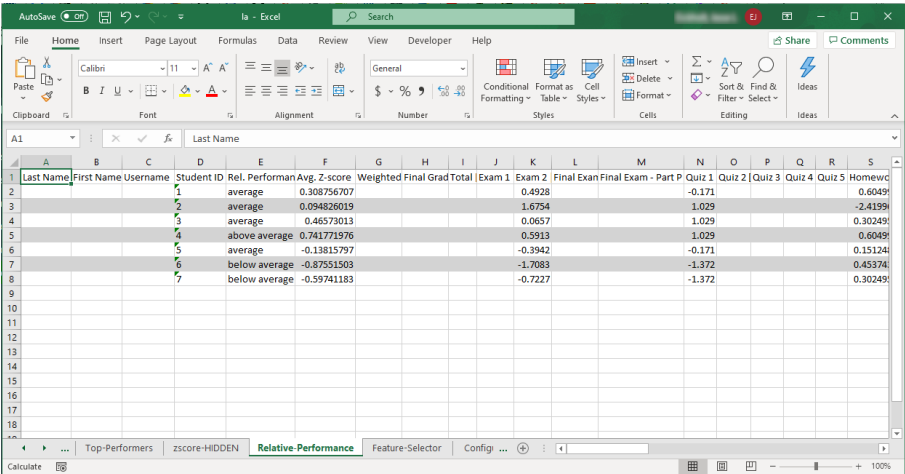


Figure 2: Sample Relative Performance sheet generated by LAGradebook.

Given that the Python program embeds needed logic and calculations inside the spreadsheet, new assessment data cannot be added directly to the LAGradebook. A typical use case would involve regenerating the LAGradebook periodically throughout a course. After each use, the original sheet could be imported into a LMS or other gradebook. Configuration data from the control page (e.g., selections for top and relative performance) is persisted across regenerations by embedding some metadata in the title of each column.

## 4 Discussion

The LAGradebook can be utilized in many ways by instructors to intervene on their behalf and to provide students with timely feedback. Presented here are a few sample use cases.

### 4.1 Relative Performance

LAGradebook allows instructors to select assessment items and determine a student's relative performance (e.g., above average, average or below average). This is done by determining a z-score for each assessment item. An overall measure of relative performance is calculated as the average z-score of selected assessment items for relative performance. Similarly, assessment items can be selected and top performers are identified on a per assessment item basis. Overall counts for top performance are calculated on a per student basis. Relative performance and top performance counts are reported on an individual student report that can be printed and given to the student. It is also possible to export the relative performance measures to a learning management system. At present, the only system supported is Blackboard.

### 4.2 Reporting

As the LAGradebook was designed to be agnostic with regards to the learning management system, it does not rely on the learning management system to relay information back to the student. To facilitate reporting, the LAGradebook generates a worksheet of student reports. Each report will print on a separate page and contains overall relative performance, relative performance by selected assessment items, a graph of selected assessment items, top performance count, performance on a selected assessment item and a customized message. Preceding the reports is a control panel that determines which fields should be populated. At certain checkpoints during the course, assessment data could be processed into a LAGradebook. The reports could be printed and distributed to students.

### 4.3 Identification of At-Risk Students

At the end of a course offering, an instructor can use a LAGradebook to calculate pairwise correlation metrics between the final grade and assessment items collected throughout the delivery of the course. These correlation values appear on a separate worksheet and color coded to indicate the strength of the relationship. Assessment items that are highly correlated with the final grade can be used as prognosticators of a future student's final performance provided that a similar assessment item is administered in a later offering of the same course. To be of value, the assessment item used should be one that occurs early in the course to allow time for corrective action or remediation.

## 5 Conclusion

There are many ways in which learning analytics helps students and instructors in promoting student success. In spite of the benefits, institutional efforts to implement learning analytics have been met with some resistance due to the size of institutional systems, the breadth of involvement of personnel across the institution and privacy concerns. To complement larger, institutional efforts for learning analytics adoption, we have developed a tool to support smaller scale applications. The tool ingests raw, student assessment data and produces a rich, learning analytics enabled spreadsheet. The spreadsheet calculates descriptive statistics and relative performance and contains pair-wise correlation metrics of assessment items and individual student reports. The reports and comparative performance measures help students monitor their relative performance. By using similar assessments and historical course data, instructors can identify students at-risk of failing a course. The low-cost and course-specific nature of the tool supports an increase in access to learning analytics, making it available to more classrooms.

## References

- [1] Datalab tools + resources. <https://www.cmu.edu/datalab/tools/>.
- [2] Kimberly E Arnold and Matthew D Pistilli. Course signals at purdue: Using learning analytics to increase student success. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 267–270. ACM, 2012.
- [3] Blackboard. Blackboard analytics. <https://www.blackboard.com/education-analytics/index.html>.
- [4] Tristan Denley. Austin peay state university: degree compass, 2012. EDU-CAUSE Review Online. <https://er.educause.edu/articles/2012/9/austin-peay-state-university-degree-compass>.

- [5] Rebecca Ferguson, Leah P. Macfadyen, Doug Clow, Belinda Tynan, Shirley Alexander, and Shane Dawson. Setting learning analytics in context: Overcoming the barriers to large-scale adoption. *Journal of Learning Analytics*, 1(3):120–144, 2014.
- [6] John Fritz. Classroom walls that talk: Using online course activity data of successful students to raise self-awareness of underperforming peers. *The Internet and Higher Education*, 14(2):89–97, 2011.
- [7] John Fritz. Using analytics to nudge student responsibility for learning. *New Directions for Higher Education*, 2017(179):65–75, September 2017.
- [8] Eric Gazoni and Charlie Clark. <https://openpyxl.readthedocs.io/en/stable/index.html>.
- [9] Steven Lonn, Timothy A. McKay, and Stephanie D. Teasley. Cultivating institutional capacities for learning analytics. *New Directions for Higher Education*, 2017(179):53–63, 2017.
- [10] Leah P. Macfadyen, Dennis Groth, George Rehrey, Linda Shepard, Jim Greer, Douglas Ward, Caroline Bennett, Jake Kaupp, Marco Molinaro, and Matt Steinwachs. Developing institutional learning analytics ‘communities of transformation’ to support student success. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference, LAK ’17*, pages 498–499, New York, NY, USA, 2017. ACM. Event-place: Vancouver, British Columbia, Canada.
- [11] Tim McKay, Kate Miller, and Jared Tritz. What to do with actionable intelligence: E<sup>2</sup>Coach as an intervention engine. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 88–91. ACM, 2012.
- [12] George Rehrey, Linda Shepard, Carol Hostetter, Amberly Maurine Reynolds, and Dennis Groth. Engaging faculty in learning analytics: Agents of institutional culture change. *Journal of Learning Analytics*, 6(2):86–94, July 2019.
- [13] Kaiwen Sun, Abraham H. Mhaidli, Sonakshi Watel, Christopher A. Brooks, and Florian Schaub. It’s my data! Tensions among stakeholders of a learning analytics dashboard. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI ’19, pages 594:1–594:14, New York, NY, USA, 2019. ACM.
- [14] Angela Van Barneveld, Kimberly E Arnold, and John P Campbell. Analytics in higher education: Establishing a common language. *EDUCAUSE Learning Initiative*, pages 1–11, January 2012.

# Engaging Early Programming Students with Modern Assignments Using BRIDGES\*

*Allie Beckman<sup>1</sup>, Matthew Mcquaigue<sup>1</sup>, Alec Goncharow<sup>1</sup>*

*David Burlinson<sup>1</sup>, Kalpathi Subramanian<sup>1</sup>*

*Erik Saule<sup>1</sup>, Jamie Payton<sup>2</sup>*

*<sup>1</sup>Computer Science, UNC Charlotte*

*{abeckma2, mmcquaig, agoncha1, dburlins, krs, esaule}@uncc.edu*

*<sup>2</sup>Computer and Information Sciences, Temple University*

*payton@temple.edu*

## Abstract

Early programming courses, such as CS1, are an important time to capture the interest of the students while imparting important technical knowledge. Yet many CS1 sections use contrived assignments and activities that tend to make students uninterested and doubt the usefulness of the content. We demonstrate that one can make an interesting CS1 experience for students by coupling interesting datasets with visual representations and interactive applications. Our approach enables teaching an engaging early programming course without changing the content of that course. This approach relies on the BRIDGES system that has been under development for the past 5 years; BRIDGES provides easy access to datasets and interactive applications. The assignments we present are all scaffolded to be directly integrated into most early programming courses to make routine topics more compelling and exciting.

## 1 Introduction

Computational literacy and problem-solving skills are crucial facets of an increasingly tech-driven economy and world. While enrollments in computing

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

majors have grown in recent years, particularly high attrition rates in these degree programs hamper the rate at which colleges and universities contribute to the modern workforce. Students in introductory and second-year courses are most susceptible, and much work has been done to investigate and address the factors contributing to the erosion of this student population [2].

One of the primary factors in maintaining interest among computing majors is their level of engagement with the course material. The perceived relevance of the material to the students' own lives and careers is crucial for kindling a desire to learn how to solve more complex problems down the road. Unfortunately, this is an area of weakness in many computing programs: the introductory courses are packed full of students, and it is enticing for universities to prioritize scalability over quality and rigor by relying on graduate students or more automated tools and paradigms to teach and evaluate the basic content.

Programming assignments in introductory courses have traditionally been contrived to teach basic structures like logical branching and loops using toy datasets, with basic command line input and output comprising students' interaction with the program. More engaging, modern approaches generally involve socially or culturally relevant data, simple graphical libraries, and more gamification of the material. All these features are prioritized in our work.

We present in this paper a lean educational framework that makes student engagement central to the content of introductory programming courses, without changing or compromising the rigor, content, and learning goals of these courses. Our BRIDGES framework uses a mix of real-world datasets from different domains, simple games and interactive applications as engagement tools to emphasize and reinforce core concepts in introductory programming. We present a set of assignments and their relationships to programming concepts, and we show how they were deployed in introductory CS courses and perceived by students of these classes. The framework and assignments are available online (<https://bridgesuncc.github.io/>).

## 2 Related Works

**What CS1 typically look like.** The typical first course in computer science, or CS1, introduces students to programming using a high level programming language such as Java or Python. The course goal is to introduce the basic constructs of the programming language, such as variables and expressions, control structures, functions and simple data structures like lists and strings. These courses train students to solve nontrivial problems using these constructs (e.g., see classic CS1 exemplar [11]).

There are wide variations in how the basic concepts are taught, in terms of the learning environments, tools, pedagogical approaches, and the student pop-

ulation (majors, non-majors) and demographics. Many introductory courses have begun to incorporate graphics, GUIs, and visualizations, as a creative output produced in projects [8, 4] or to illustrate key aspects of the underlying objects or algorithms [7, 3]. Furthermore, a rising awareness of the multi-disciplinary value of computing literacy has encouraged some institutions to experiment with different flavors of introductory programming courses based on game development [1], robotics [6], and data science [5].

**What Makes Students Engaged.** Ultimately, the goal of courses and curriculum is the overall education and academic success of the learners. To that end, materials that capture the imagination of incoming students and reinforce their interest and motivation in computing are particularly valuable. Popular assignment repositories (Nifty Assignments [14], EngageCSEdu [13], etc.) tend to include the ‘fun’ factor, as do game-themed assignments [16]. Usage of real-world and large datasets in course projects have also proven successful [12, 4], in contrast to using tiny, contrived or toy examples that fail to engage students.

In recent years, active learning techniques have been implemented in classrooms to promote student engagement, and include any combination of lab-based instruction, flipped classroom settings, gamification, peer-learning, and use of multimedia content [15, 9, 10].

### 3 The BRIDGES System

The BRIDGES system [4] is relevant to the goals of introductory CS courses such as CS1 and CS2. It has the capability to provide easy access to external datasets that can be readily used in course assignments. Secondly, it provides a 2D abstraction of a grid that can be used for games and image processing. The system provides bindings for the commonly used languages in early CS courses (Java, C++, and Python). Finally, results from assignments are highly visual and can be shared with friends and family, thanks to web-based rendering.

**Dataset Access.** BRIDGES provides simple APIs to access external data sources such as USGS Earthquake data, Wikidata, IMDB actor/movies, Genius’ Song Lyrics, and OpenStreet Maps. A single function call returns data from a specific source, typically a list of objects, that can then be directly used. By using interesting datasets from different domains, assignments can be made more real and relevant to students.

**Visualizing Bitmap Images.** BRIDGES supports bitmap images through a 2D grid abstraction which can be the basis for assignments on images and



image processing. This also directly relates to 2D arrays and array addressing, which are central to CS1 and CS2.

**Game API.** BRIDGES supports a simple Game API that forms the basis for a number of 2D games that are readily usable and aligned with the goals of early CS courses. The game API has 4 core functions, (1) Reading Inputs, (2) Updating the game state using customized game mechanics, (3) Rendering to screen, and, (4) Maintaining a frame rate of 30 frames per second. In order to maintain simplicity of the API for new programmers and ensure smooth rendering, 2D games are implemented as 2D grids with a maximum of 1024 cells where each can be assigned a color and one of 256 predefined sprites, and 10 input keys for interaction. These constraints still enable the construction of many games and applications, and the *(student) programmer can focus on implementing the game logic and updates to the game state, while the display output and graphics are the responsibility of the system.* This lets the student focus on the course-level goals and not on the tool-level details. The user is responsible for implementing two functions: (1) `initialize()`, which is run once at the beginning of the game, and (2) `gameLoop()`, which contains all of the game logic and is called for each frame of the game.

This API is simple yet expressive enough to enable the implementation of a number of 2D games such as Bugstomp, Snake, 2048, Infinite Runner, Minesweeper, and Racing Car; as well as many of the Nifty [14] assignments such as Falling Sand, Spreading of Fire, and Hurricane Tracker. Each of these can be scaffolded to align with the learning outcomes of an early CS courses.

Students will typically run the code in their IDE and interact with the game through a web browser. However, our system also supports exporting games as standalone Android applications.

## 4 A Set of Engaging CS1 Assignments

We now describe a sequence of engaging assignments using BRIDGES that can be the basis for a CS1 curriculum. Table 1 illustrates the assignments and topics these assignments are aligned with. Assignments are scaffolded (available on our website) so that the students will see the specific functions that are to be implemented, in line with the objectives of the assignment. Instructors may request solutions for planning their course.

**Etch a Smile.** The student is given the task of drawing a smiley face. There are two versions of this assignment, Students write single lines of code which fill specific cells on a 2D grid using  $x$  and  $y$  values with a color of their choice. Students can also draw symbols on a cell. Alternately, they can use loops to

Assignment	Topics	Engagement
Etch A Smile	Functions	creativity, fun, visual output
BugStomp	Conditionals	Game Interaction
Tic Tac Toe	Conditionals, Std. I/O	Game Interaction
Song Lyrics	2D arrays, loops, conditionals, strings	real data, explore personal choices
Image Proc.	2D arrays, loops, File I/O	real data, visual output
Mountain Paths	2D arrays, loops, greedy algorithms	real data, visual output

Table 1: Example CS1 Projects, Topic Mappings and engagement factors

fill areas of the 2D grid with specific colors to create a face.

**Where does it fit?** Students will gain experience with generating a visually pleasing output using code that is more interesting than vanilla “hello world” function. They will also gain an understanding of coordinate grids from a programmers perspective. Familiarity with loops can be used to augment this assignment. Fig. 1 illustrates some examples.



Figure 1: Etch A Smile Face Example

**Bugstomp.** The game involves moving a sprite around a 2D grid to step on randomly generated bug sprites. The assignment involves moving around a 2D grid, generate random positions to place bugs in the grid and ensure that the player and bug are always within the grid.

**Where does it fit?** Moving a sprite within a 2D grid requires using input keys, updating the game board, and using conditional statements to ensure all positions stay within the grid, and checking if the character stepped on a bug.

**Tic Tac Toe.** An old assignment with a new look: students create a 3 x 3 board where they take turns placing their symbol and trying to get three in a row, column, or diagonal. Students can play against friends or develop a human vs computer experience. The board displays user chosen symbols for the players, and arrow keys are used to select a move.

**Where does it fit?** Tic-Tac-Toe is an excellent assignment for learning conditional statements, loops, and data management. Students can compare strings or values in an array to update the board. The game grid object of BRIDGES also provides options for students who have not seen arrays yet.

**Analyzing Repetition in Song Lyrics.** The student selects a song from an external data source (supported by BRIDGES) and parses each word. A 2D matrix is created using the words of the song that records whether the  $i$ -th word of a song is the same as the  $j$ -th word and creates unique patterns on a grid depending on the words repetitions of the selected song. See a highly repetitive song in Fig. 2c and one with little repetition in Fig. 2b.

**Where does it fit?** Students will have gained experience parsing and comparing strings using tokens. They will also have an improved understanding of loops, 2D array processing, conditional statements.

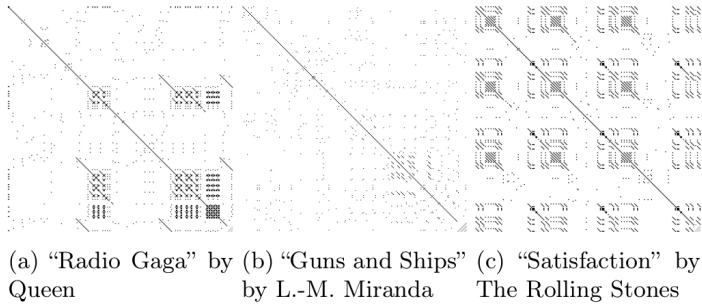


Figure 2: Song Lyrics: Repetition Pattern for different songs

**Image Processing.** The student reads images given in some text format (e.g., PPM RGB images) into a 2D array and performs simple image processing operations such as image flipping, color flipping, removing a color primary. Students learn to work with images and how to manipulate colors.

**Where does it fit?** This is an excellent assignment for 2D array processing, understanding image structures: storage of the array data in a linear sequence, relationship between 1D and 2D addresses, and exercising loops and conditions.

**Mountain Paths.** This is an adaptation of a Nifty assignment [14] but it uses BRIDGES’s visualization capabilities for displaying outputs. Students are provided with an elevation image of a mountain as a text file. Starting from a random pixel on the left edge of the image, the goal is to find a path that takes the least amount of effort to get to the right edge of the image. A simple greedy algorithm is used to make a local decision to move from one pixel to the next, such that the pixel with the least difference in elevation between the current position and the next position is chosen. The selected set of pixels are drawn in color to illustrate the path, as can be seen in Fig. 3.

**Where does it fit?** This assignment is a very nice introduction to greedy

algorithms using a simple and highly engaging real world dataset. Students learn file I/O, implement a greedy algorithm, practice using conditionals, loops and 2D array processing.

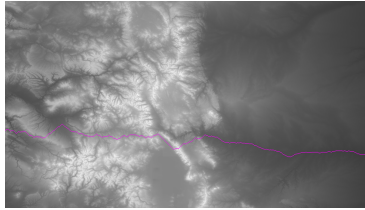


Figure 3: Path of Least Elevation Example

## 5 Student Feedback

The projects detailed in Section 4 have been deployed in sophomore and junior level CS courses. These projects have been used often to introduce a new programming language (C++), or more frequently, as early 'warm-up' projects early in the course, or to illustrate a specific concept, such as object design. Table 2 illustrates the deployment of several of these projects over the past 3 years. We conducted project surveys and student reflections on the assignment that included the following questions:

1. What are the essential concepts that were learned by completing the assignment? [Short Answer]
2. Why is this important? [Short Answer]
3. How does the concept contribute to understanding how to program? [Short Answer]
4. The assignment was relevant to my career goals [5 point Likert scale].
5. The assignment increased my interest in computing [5 point Likert scale]

The reflection survey asked students the following questions:

1. Rate the difficulty of the module (5 point scale)
2. Roughly what percent of the module did you complete [25, 50, 75, 100]
3. Did you find the tasks engaging and meaningful? (4 point scale)
4. What did you like and not like about the assignment? [Short Answer]

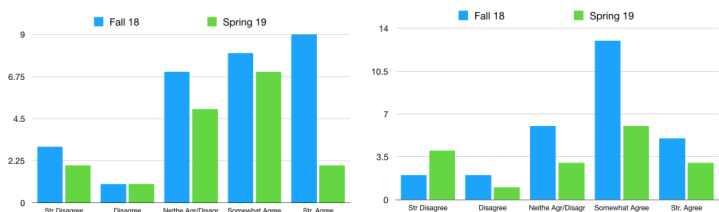
The class enrollment varied, ranging from 30-50 students across the different semesters. Participation was high (a small amount of credit was assigned for completing the surveys); however, only students who completed a substantial portion of the project were counted towards the survey participation.

Assignment	Semester/Year	Data Collection
Etch A Smile	F18,Spr.19	Student Reflection
Tic Tac Toe	F17,F18	Surveys, Student Reflection
Snakes & Ladders	Spr. 19	Student Reflection
Mountain Paths	F17,F18,Spr.19	Surveys, Student Reflection
Image Proc	F17,F18,Spr.19	Surveys, Student Reflection

Table 2: Student Feedback - Data Collection

**Etch a Smile.** Nearly all students found the project easy and completed it. The themes that emerged from the student reflections included the freedom to create a shape of their own choosing and play around with the color grid in creating a face. Examples of student quotes included ‘...allows creativity in a new way!’, ‘experiment and mess around with the coding’, ‘enjoyed the module and ... eager to learn more’, ‘enjoyed the assignment, it was engaging’.

**Tic Tac Toe.** This familiar game was also well received with over 80% completing the assignment and found it engaging. Student reflections were overall very positive, appreciated the moderate challenge, though several found it difficult as they were new to C++. Example free form responses included ‘enjoyed making tic tac toe ... later plan to modify it to run on an arduino’, ‘game is a good way to demonstrate understanding of programming’, ‘liked the assignment, enjoyed creating the game’.



(a) *The assignment increased my interest in Computing.* (b) *The assignment was relevant to my career goals.*

Figure 4: Project Surveys (Fall 18 and Spring 19): Mountain Paths Assignment

**Mountain Paths.** This was a harder assignment, with 65-80% of the class completing 75% or more of the assignment. Almost all found the assignment engaging. The major themes from the reflection surveys were on the challenge of the assignment (‘very challenging and I learned quite a bit’, ‘feel challenged, but also feel satisfied’, ‘was not prepared’), appreciation of the flexibility in the assignment (‘loved the assignment allowed for different inputs’, ‘make a

color object and amend it on each loop - it worked’), choice of the assignment (‘excellent practical example of greedy algorithm’) and the visualizations (‘liked seeing the visualizations’, ‘enjoyable to finally see the best path line’).

Fig. 4 shows the results of a survey of student’s attitude towards computing. The results from two semesters show that the student were engaged by an assignment they perceived as relevant.

**Image Processing** This assignment continued from the Mountain Paths assignment. Students overall found the assignment challenging (roughly half in Fall 2018, 70% in Spring 19 found it difficult), and about 60-70% completed 75% or more of the assignment. Over 90% found the assignment engaging. Students appreciated the similarity of this assignment to the previous assignment (‘last assignment was necessary to prepare me’), appreciation for the assignment (‘good assignment with fairly clear instructions’), enjoyment (‘liked the image processing portion’, ‘really liked seeing how easily implement some simple image processing with this assignment’, ‘liked manipulation of the image’), assessing success/failure (‘my procrastination bit me...’)

## 6 Conclusions

We have presented a set of highly engaging assignments that meet the principal goals of a typical introductory programming course like CS1. The assignments contain important elements of engagement, such as the use of real-world data, visualizations to see the final outputs, and interactive games and applications. We deployed and tested these assignments and collected student feedback. Overall, the student responses have been very positive: They find the assignments interesting, fun, and yet challenging.

Although these assignments have not yet been deployed in CS1, the topics and the assignments are at the level of a CS1. We have begun working with CS1/CS2 instructors using BRIDGES as part of their course. It would be interesting to do a comparison of the student responses with those deployed in a dedicated CS1 course using these engagement principles.

## Acknowledgment

This material is based upon work supported by the National Science Foundation under grant no. DUE-1726809 and CCF-1652442.

## References

- [1] Jessica D Bayliss and Sean Strout. Games as a flavor of CS1. In *ACM SIGCSE Bulletin*, volume 38, pages 500–504, 2006.

- [2] Theresa Beaubouef and John Mason. Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin*, 37(2):103–106, 2005.
- [3] Sarah Buchanan, Brandon Ochs, and Joseph J LaViola Jr. CSTutor: a pen-based tutor for data structure visualization. In *Proc. ACM SIGCSE*, pages 565–570, 2012.
- [4] David Burlinson, Mihai Mehedint, Chris Grafer, Kalpathi Subramanian, Jamie Payton, Paula Goolkasian, Michael Youngblood, and Robert Kosara. Bridges: A system to enable creation of engaging data structures assignments with real-world data and visualizations. In *Proc. ACM SIGCSE*, pages 18–23, 2016.
- [5] Sarah Dahlby Albright, Titus H Klinge, and Samuel A Rebelsky. A functional approach to data science in CS1. In *Proc. ACM SIGCSE*, pages 1035–1040, 2018.
- [6] Amy Delman, Adiba Ishak, Lawrence Goetz, Mikhail Kunin, Yedidiah Langsam, and Theodore Raphan. Development of a system for teaching CS1 in C/C++ with lego NXT robots. In *FECS*, pages 396–400, 2010.
- [7] Prasun Dewan. How a language-based GUI generator can influence the teaching of object-oriented programming. In *Proc. ACM SIGCSE*, pages 69–74, 2012.
- [8] Ira Greenberg, Deepak Kumar, and Dianna Xu. Creative coding and visual portfolios for CS1. In *Proc. ACM SIGCSE*, pages 247–252, 2012.
- [9] Mark Guzdial. A media computation course for non-majors. In *Proc. ITICSE*, pages 104–108, 2003.
- [10] Diane Horton, Michelle Craig, Jennifer Campbell, Paul Gries, and Daniel Zingaró. Comparing outcomes in inverted and traditional CS1. In *Proc. ITICSE*, pages 261–266, 2014.
- [11] Joint Taskforce on ACM Curricula. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM/IEEE Computer Society, 2013.
- [12] Lucas Layman, Laurie Williams, and Kelli Slaten. Note to self: Make assignments meaningful. In *Proc. ACM SIGCSE*, SIGCSE ’07, pages 459–463, 2007.
- [13] Alvaro Monge, Beth A. Quinn, and Cameron L. Fadjo. EngageCSEdu: CS1 and CS2 materials for engaging and retaining undergraduate CS students. In *Proc. ACM SIGCSE*, pages 271–271, 2015.
- [14] Nick Parlante. Nifty assignments, 2018.
- [15] Johanna Pirker, Maria Riffnaller-Schiefer, and Christian Gütl. Motivational active learning: Engaging university students in computer science education. In *Proc. ITICSE*, pages 297–302, 2014.
- [16] Kelvin Sung, Rebecca Rosenberg, Michael Panitz, and Ruth Anderson. Assessing game-themed programming assignments for CS1/2 courses. In *Proc. of GDCSE*, GDCSE ’08, pages 51–55, 2008.

# Using JShell in CS1\*

*Joseph Kendall-Morwick<sup>1</sup>*

*<sup>1</sup>Department of Computer Science, Mathematics, and Physics  
Missouri Western State University*

*St. Joseph, MO 64507*

*[jkendallmorwick@missouriwestern.edu](mailto:jkendallmorwick@missouriwestern.edu)*

## Abstract

A recent update to the Java Language and Java Development Kit (JDK), version 9, has added an official REPL (JShell). JShell allows instructors to introduce Java programming concepts more selectively and to develop exercises free of distractions from verbose syntax relating to programming concepts beyond the scope of the lesson. This paper will relate experience integrating JShell in to a CS1 course and show how it provides for an increased focus on crucial topics, rather than boilerplate, in the early weeks of a course.

## 1 Introduction

Introductory Java programming (CS1) students are frequently asked to ignore a considerable amount of code when they read their first “Hello World” program in Java. Consider this implementation:

```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

The Java programming concepts inherent in the code in this example include: access modifiers, class definition, naming conventions, method definitions (particularly, a static method and also a void return type), statically typed parameters, array types,

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.



command-line arguments, the *System* class, static references (to a field, specifically), the *PrintWriter* class, statements, method calls, and literal strings. The student is intended to soon understand the final three concepts from the “Hello World” example and expected to ignore the code pertaining to the other concepts as boilerplate required to execute the statement. At best, the student will simply be briefly distracted by the presence of this code, and at worst, may form false assumptions about what the code is intended for.

Recent updates to the Java language and Java Development Kit (JDK), specifically versions 9 and 10, have added an official REPL and local variable type inference. These updates allow instructors to introduce many important Java programming concepts at the time that students are expected to learn them, particularly those within the crucial early weeks of the class. For example, consider the contrast in a “Hello World” implementation using JShell:

```
print("Hello World!");
```

This paper will argue for the adoption of JShell in CS1 courses using Java and relate experience of its use through three semesters of CS1 at Missouri Western State University. Further, it will make recommendations for topic ordering within the initial weeks of CS1 focusing on the most simple Java concepts first, in a logical order, and without broaching more advanced concepts before they’re needed or before they can be properly introduced.

## 2 Justification and Related Work

Although “Hello World” is a rather extreme example, Java presents many other similar challenges to novice programmers. Such concerns have been recognized by other instructors and have led some to adopt or develop software tools in teaching Java in CS1 (all predating JShell). Kölling developed BlueJ, an educational IDE with a built in REPL (called the “code pad”) designed to focus early on object-oriented concepts [10]. DrJava is another educational IDE that includes a REPL-like component (the interactions pane) [1]. Rather than a REPL or IDE, Nasrawt et al. developed LessJava, a concise and simple language that is very similar to Java but focuses on the basics of procedural programming [16]. Many web-based applications that provide exercises where students develop methods or other small snippets of Java have been produced, such as codingbat, CodeLab, Problets, and others [3]. Although these tools are not all REPL’s, they all allow students and instructors to work with smaller snippets of Java programs and provide a more interactive programming environment.

Many papers have been published discussing topic order for teaching Java. The primary debate is over “Objects First” or “Objects Late” [4]. While using JShell can help address objects early, it does not provide visualizations popular with many who advocate this route [6]. To a lesser extent, focusing on either functional or imperative topics first is also a relevant debate [15, 2]. Both approaches can be better supported by JShell. In particular, we share the same concerns about topic order with Java

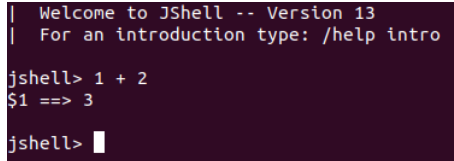
A screenshot of the JShell REPL interface. The background is dark purple. The text is white. It shows the welcome message "Welcome to JShell -- Version 13" and a hint "For an introduction type: /help intro". Below that, the user enters the expression "1 + 2" at the "jshell>" prompt. The system responds with "\$1 ==> 3". The prompt "jshell>" is shown again with a white cursor.

Figure 1: Evaluation of a simple expression in JShell

that led Barland to adopt a “Functions First” approach, but we instead use JShell to mitigate these concerns while introducing imperative concepts first.

Concerns about Java’s learning curve have also been a motivation for many universities to choose a different language, such as Python, for CS1 [12]. Many other languages include official support for REPL environments and the debate over language choice for a CS1 class continues to be a popular topic for publishing in CS education, yet Java remains as a very popular choice for CS1 [4]. There are many considerations for choosing a language for CS1, pedagogical and otherwise, and language choice (beyond Java) will be considered beyond the scope of this paper.

### 3 JShell

In Fall of 2017, Java 9 was released, along with a new official REPL (Read Evaluate Print Loop) called “JShell”. JShell provides a text-based user interface (TUI) that allows programmers of all skill levels to quickly evaluate small “snippets” of Java code (isolated expressions, statements, and declarations) [7]. The programmer simply enters the snippet at the prompt and the snippet is immediately parsed and evaluated. JShell then prints out the result, which may be a syntax error message, run-time error, or in the case of an expression, the value resulting from evaluation. Standard I/O is also handled through the TUI.

JShell eliminates several cumbersome requirements on executing arbitrary Java snippets. There is no requirement of a public class or a main method. Import statements can be issued at any time. In fact, any type of snippet can be evaluated in any order. This can be helpful for expert programmers who wish to quickly test segments of their code or experiment with API’s. However, it is also of significant utility to novice programmers who aim to experiment with fundamental Java language features and attain quick feedback in a controlled environment. Small snippets of Java code relating only to the concepts an instructor desires to teach can be experimented with in a meaningful way without the need for the typical Java boilerplate.

Although similar tools were referenced in the related work section, there are compelling reasons to choose JShell. JShell is an official part of the Java JDK and consists of two components: the JShell tool and the JShell API. The tool (the application the user interacts with directly) is robust and intended to be used by professional programmers. Thus it has desirable features such as tab auto-completion and a complete collection of error messages. The API provides some opportunity for extension

of the JShell system and inclusion of JShell in to other tools. Because the JShell API is maintained by Oracle on the same schedule as the rest of the JDK, JShell can be relied upon to be current with any updates to the Java language or API in the future.

This assurance paid immediate dividends with the release of Java 10 and JEP 286: Local-Variable Type Inference and its simultaneous support in JShell [8]. One reason some universities may choose Python over Java for CS1 is Java’s static types for variable definitions. JShell variable definitions are treated similarly to local variable definitions and can take advantage of type inference. Thus, students learning variable definition and assignment during the early weeks of CS1 can use the **var** keyword to define variables rather than being forced to supply a type, mitigating two of the three principle concerns that motivated Nasrawt et al. [16].

JShell also has its limitations for classroom use. The API allows JShell to be used as a component in other software, but it does not easily afford changes to the functionality of the JShell tool. However, this hasn’t stopped some from extending the capabilities of the JShell tool [19]. Furthermore, JEP 222 explicitly states that “a new interactive language is not the goal” [7]. Executing a JShell “script” without the interactive TUI can lead to unexpected behavior, particularly with standard I/O. Some libraries, such as JavaFX, may also not function as expected within JShell. However, there are workarounds for both of these example limitations [11, 5]).

## 4 Redesigning a CS1 Course with JShell

At Missouri Western State University, Java has been the primary language used for the introductory sequence in computer science comprised of CS1 (CSC 184 - Computing Concepts I), CS2 (CSC 254 - Computing Concepts II), and Data Structures (CSC 285). For many years, the popular “Introduction to Java Programming” text has been used in all three courses [14]. CS1 courses were taught following the initial sequence of topics in this book. Early examples and exercises utilized a common IDE (typically IntelliJ) and later also utilized codingbat – an online tool for unit-tested method development exercises [18].

In the past two years, CSC 184 was redesigned around the use of JShell. New course materials (a set of notes that has become a quasi-textbook and a library of demonstration videos) utilizing JShell were developed. These materials followed a new topic ordering that fully explains concepts as they are introduced syntactically. The most significant changes to the topic ordering were in the early weeks. These, specifically, are outlined in the next subsection. The section following that one describes how JShell was incorporated in to new tools to facilitate the new topic ordering.

### 4.1 Topic Ordering

The first concepts covered, in order, were: numeric literals, arithmetic operators and expressions, function calls, the *int* and *double* types, and casting. Students were tasked with developing simple and complex arithmetic expressions parameterized by variables defined earlier (for example, the Gaussian PDF formula) in an attempt

to bridge their knowledge of algebra to computer programming (a goal shared in approaches taken by Laengrich et al. [13] and Barland [2]). Static imports were used to avoid the need for static references to methods in the *java.lang.Math* class.

Method definitions could be introduced next, but a more imperative approach was chosen to leverage JShell’s interactive interface towards teaching the algorithm concept. JShell allows hassle-free variable definitions and assignment and allows programmers to use these statements interactively (essentially using JShell as a calculator with a sophisticated memory function), making the interactive execution of a pseudo-code algorithm (such as the Euclidean algorithm) possible before conditional and loop structures are introduced.

At this point new control abstractions could be introduced, but we opted to expand the available use-cases for JShell beyond calculation to help motivate the students, introducing them instead to objects (but not class definitions), basic graphics, strings, I/O, and the concept of a script (even though JShell can’t be officially considered a scripting language). Graphics and I/O are used to reinforce basic procedural programming concepts and help students understand more complex data types and basic collections.

After this, students were introduced to method definitions. The methods students were tasked to define in JShell were not instance methods but essentially “functions” with some number of parameters and a return type. This builds on the early work with expressions and opens the door to more of the ideas in the “Functions First” approach [2].

## 4.2 Tools

JShell itself was the primary tool for most coding exercises. JShell tracks each snippet executed during an interactive session and allows the saving of these snippets to a file. In most exercises, students were asked to complete a number of tasks using JShell (trace an algorithm, develop an expression, develop a method, etc) and save their snippet history to a file. JShell can also load these files, re-evaluating each snippet.

Introductory students were not expected to be well acquainted with command-line interfaces and thus it was desirable to use startup-scripts with JShell. One script used simply starts JShell with the printing functions available (avoiding the need for *System.out.println*), and another incorporates a modification that allows independent execution of JShell “scripts” [11]. This approach could also be extended to include other useful functions, libraries, or imports and help the instructor hide other details of working with and executing Java programs that make it difficult to focus on just a few programming topics. For example, a small startup script utilizing some static imports was used to postpone coverage of packages, classes, and import statements while covering static method calls from the *Math* class in the first week:

```
import static java.lang.Math.*;
```

Codingbat was again used for some unit-tested method development exercises, but a new tool currently under development, “Code Cafe”, that utilizes JShell as an execution kernel has also been used [9]. Code Cafe exercise definitions allow inclusion

of startup scripts (such as the example above), code transformations, and arbitrary goal definitions (generalizations of unit tests) that provide more flexibility in exercise structure and specificity in feedback. For example, in the previously mentioned Gaussian PDF exercise, a transformation from a Java expression to a Java method definition with a single return statement was used so that the student’s expression could be easily unit-tested. As another example, all method development exercises in Code Cafe use two custom goals that indicate whether the student correctly specified the return type or the parameter types, respectively, in their method definition.

## 5 Evaluation

In two sections of CS1 in Fall 2017, the Liang textbook and the classic “Hello World” application approach was used. In subsequent semesters, the same instructor (the author) used the new JShell based curriculum in three sections of CSC 184. It should be noted that the JShell cohort includes students taking CS1 in both Spring and Fall, whereas the “Hello World” cohort only includes students taking CS1 in Fall. These cohorts consisted of 38 and 36 students respectively. The proportion of all of these students who went on to attempt CS2 was determined. These figures include students enrolled in CS2 during the Fall 2019 semester who had not yet completed the course at the time data was collected.

Next, the mean change in grade between CS1 and CS2 was determined for each cohort. For each student who completed one of these CS1 sections and a section of CS2, their CS1 grade was compared to their grade in CS2. All grades were issued on an A (4.0), B (3.0), C (2.0), D (1.0), and F (0.0) scale, and only students who completed both sections were included in the mean (students who officially withdrew or who received an F due to non-attendance were excluded). These figures also do not include students enrolled in CS2 during Fall 2019 since they had not completed the course at the time data was collected.

## 6 Results

	attempted CS1	attempted CS2
Hello World	38	19 (50%)
JShell	36	21 (58%)

Table 1: Students attempting CS2 after taking CS1

The results are available in Table 1 and Table 2. Overall, students received lower grades in CS2 than in CS1, but when comparing the drop in GPA among students who took a “Hello World” section of CS1 to those who took a JShell section, it should be noted that the GPA of the JShell students dropped less. Students who took a JShell section also attempted CS2 with greater frequency. However, the differences in the results in both cases are not statistically significant.

	completed CS2	GPA change
Hello World	15	-0.47
JShell	10	-0.20

Table 2: GPA change for students after completing CS2

## 7 Conclusions and Future Work

Although there was not enough data to provide statistical significance in the results, the favorable result from the JShell cohort coupled with the instructor’s perception of this cohort’s improved understanding through the first four weeks is encouraging and motivates further study.

We advise consideration of JShell in several contexts for improving student learning in CS1 courses using Java and the consideration of topic orderings that JShell helps facilitate. REPL’s are popular teaching tools for CS1 and JShell has shown promise as a means of making it easier to teach Java to novice programmers. Extension of JShell provides opportunity for enhancing this benefit through development of start-up scripts or loading libraries that could provide automated grading and other features of tutoring software. Additionally, it is warranted that teaching with JShell be further studied in the context of teaching Java in CS1, but also in comparison to teaching other REPL-based languages in CS1.

Lastly, the JShell API, being an official JDK component, also offers a means of keeping other pedagogical tools current with Oracle’s more rapid update cycle for Java. We hope to further explore the use of JShell as a component in the “Code Cafe” tool to develop original exercise types with new types of goals [9], and hope that others developing pedagogical software will take notice of JShell and consider its use to enhance their own product. We also see value in using new types of tools utilizing the JShell API (such as Jupyter notebooks [17]) and further studying the impact of these tools in the CS1 classroom.

## References

- [1] Eric Allen, Robert Cartwright, and Brian Stoler. DrJava: A lightweight pedagogic environment for Java. In *SIGCSE*, volume 2, pages 137–141. Citeseer, 2002.
- [2] Ian Barland. Some methods for teaching functions first using Java. In *Proceedings of the 46th Annual Southeast Regional Conference on XX*, ACM-SE 46, pages 256–259, New York, NY, USA, 2008. ACM.
- [3] Valerie Barr and Deborah Trytten. Using Turing’s Craft Codelab to support CS1 students as they learn to program. *ACM Inroads*, 7(2):67–75, May 2016.
- [4] Brett A. Becker and Keith Quille. 50 years of CS1 at SIGCSE: A review of the evolution of introductory programming education research. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE ’19, pages 338–344, New York, NY, USA, 2019. ACM.

- [5] bitterfox. JavaFX Supports For JShell. <https://github.com/bitterfox/JavaFXSupportsForJShell>. Accessed 2019-11-09.
- [6] Stephen Cooper, Wanda Dann, Randy Pausch, and Randy Pausch. Teaching objects-first in introductory computer science. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '03, pages 191–195, New York, NY, USA, 2003. ACM.
- [7] Robert Field. JEP 222: jshell: The Java Shell (Read-Eval-Print Loop). [/received/addedtojournal](https://openjdk.java.net/jeps/222). Accessed 2019-11-09.
- [8] Brian Goetz. JEP 286: Local-Variable Type Inference. <http://openjdk.java.net/jeps/222>. Accessed 2019-11-09.
- [9] Joseph Kendall-Morwick. Code Cafe: A Web Application and Framework for Teaching Programming Fundamentals. <https://github.com/jmorwick/codecafe>. Accessed 2019-11-09.
- [10] Michael Kölling. In Jens Bennedsen, Michael E. Caspersen, and Michael Kölling, editors, *Reflections on the Teaching of Programming*, chapter Using BlueJ to Introduce Programming, pages 98–115. Springer-Verlag, Berlin, Heidelberg, 2008.
- [11] kotari4u. JShell Script Executor. [https://github.com/kotari4u/jshell\\_script\\_executor](https://github.com/kotari4u/jshell_script_executor). Accessed 2019-11-09.
- [12] Theodora Koulouri, Stanislao Lauria, and Robert D. Macredie. Teaching introductory programming: A quantitative evaluation of different approaches. *Trans. Comput. Educ.*, 14(4):26:1–26:28, December 2014.
- [13] Matthias Laengrich, Joerg Schulze, and Amruth N Kumar. Expression tasks for novice programmers: Turning the attention to objectivity, reliability and validity. In *2015 IEEE Frontiers in Education Conference (FIE)*, pages 1–8. IEEE, 2015.
- [14] Y. Daniel Liang. *Introduction to Java Programming, Brief Version, Student Value Edition (11th Edition)*. Pearson, 11th edition, 2017.
- [15] Andrew Luxton-Reilly, Simon, Ibrahim Albluwi, Brett A. Becker, Michail Giannakos, Amruth N. Kumar, Linda Ott, James Paterson, Michael James Scott, Judy Sheard, and Claudia Szabo. Introductory programming: A systematic literature review. In *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE 2018 Companion, pages 55–106, New York, NY, USA, 2018. ACM.
- [16] Zamua O. Nasrawt and Michael O. Lam. Less-Java, More Learning: Language design for introductory programming. *J. Comput. Sci. Coll.*, 34(3):64–72, January 2019.
- [17] Spencer Park. IJava: A Jupyter kernel for executing Java code. <https://github.com/SpencerPark/IJava>. Accessed 2019-11-09.
- [18] Nick Parlante. CodingBat: Code Practice. <http://codingbat.com>. Accessed 2019-11-09.
- [19] John Poth. JShell Maven Plugin. <https://github.com/johnpoth/jshell-maven-plugin>. Accessed 2019-11-09.

# Is It Getting Foggy in Here? Cloud Computing in the Classroom\*

*Denise M. Case and Michael P. Rogers*  
*School of Computer Science and Information Systems*  
*Northwest Missouri State University*  
*Maryville, MO 64468*  
*dcase@numissouri.edu, mprogers@mac.com*

## Abstract

Cloud computing is now an integral, essential business tool. The same advantages it brings to business apply to academia, and in computer science education in particular it offers the ability to transform how and what we teach in many applied courses. This experience paper examines the history of cloud computing to provide context, offers concrete examples of how cloud computing can be used in the classroom, and provides tips for faculty wishing to explore this technology.

## 1 Introduction and Motivation

In just a few years, cloud computing has evolved from marketing-speak to a compelling business tool for two main reasons. First, it provides a dizzying array of computing services, relieving businesses of the need to install, manage, and secure software. Secondly, it entirely eliminates the need to purchase and maintain hardware while delivering *scalability*, the ability to adjust the number of deployed servers to meet fluctuating demand.

Those reasons are also, unsurprisingly, precisely why cloud computing is becoming more prevalent in computer science education. In a wide variety of applied courses that require significant computing resources — including, but not limited to, mobile computing, web services, databases, internet of things, big data, and machine learning — cloud computing spares woefully overworked

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.



and underpaid faculty [18, 10, 21] the heartache of having to set up and run these services. As a bonus, use of cloud computing in these classes prepares students for a workplace in which cloud computing plays an increasingly large role [9].

In this paper, we briefly discuss the history of cloud computing, the characteristics of modern cloud computing systems, evaluate the major cloud service providers (CSPs) vis-à-vis computer science education, identify courses where cloud computing can be beneficial, and conclude with tips for integrating cloud computing into the classroom.

## 2 History

Cloud computing — utilizing computing services and resources from a remote location — is a relatively old idea. As early as 1940, at a computing conference at Dartmouth College, George Stibitz used a teletypewriter to control a “Complex Number Computer” housed at Bell Labs, in New York City [16]. However, the Complex Number Computer was not programmable, only capable of performing a set of fixed complex number calculations. The first recognizable cloud computing systems appeared in the 1960s, with the rise of time-sharing operating systems that service bureaus ran on mainframes, and made available to businesses via modem. The term “cloud” was used in the early 1990s by telecommunications companies to indicate where their responsibilities began. The earliest use of the phrase “cloud computing”, and a prescient description of its potential, can be found in a 1996 presentation by executives at Compaq [5, 20]. Attention to cloud computing grew after a 2006 conference in which Eric Schmidt distinguished between the established paradigm – client/server, where proprietary protocols and software were used to connect the two – and an emergent paradigm, cloud computing, where any computer or mobile phone could gain access to data services and hardware, provided that the device came with a standard web browser [22]. The major cloud platform providers (Amazon, Google and Microsoft) began cloud computing operations in 2006, 2008 and 2010), respectively.

Around that time, academics began looking at adding cloud computing to the curriculum [3, 17]. By 2014, demand was high enough and the subject broad and complex enough that a comprehensive course with multiple knowledge areas was proposed [2]. By 2015, a comprehensive literature review summarized the results of various empirical studies related to the application of cloud computing in higher education [14]. Specific cloud computing competencies, called knowledge units, began to be specified for a variety of curricula, and the National Information Assurance Education and Training Program (NIETP) suggested an optional Cloud Computing Knowledge Unit when awarding

the NSA/DHS National Centers of Academic Excellence in Cyber Defense Education (CAE-CDE) [8].

### 3 Characteristics of Modern Cloud Computing Systems

To understand how cloud computing can be used in the classroom, it is imperative to understand how modern cloud computing systems are structured and what they offer. Cloud computing can be broadly classified into these categories:

- **Infrastructure-as-a-Service (IaaS)** offers hardware: servers, storage and networking. Everything else, from operating system up — must be installed by users. Fortunately, CSPs make OS installation straightforward. Using a web-based interface, users specify an operating system (from multiple flavors of Linux and Windows), machine type (with the option to specify CPU, GPU, and RAM), firewall options (http, https or none), and location. Then, with a click of a button, the user has a virtual machine (VM) at their disposal. Users interact with the VM either through the command line over ssh, or graphically, using VNC or Remote Desktop client (Desktop-as-a-Service).
- **Platform-as-a-Service (PaaS)** provides a complete working environment, with a pre-configured operating system, database access and assorted runtime environments ready to use. While the focus is on the platform, paradoxically developers do not in fact spend any time on it: they develop web apps on their local machines, using familiar tools. Command-line utilities, provided as part of the cloud provider's software developer kit, are used to upload code to be deployed on the cloud. A signature feature of cloud computing is the ability to scale the number of servers on which the applications run, to meet demand. Our students need to be aware of this capability, although the applications that they create in class will likely not require it.
- In **Software-as-a-Service (SaaS)**, a web browser provides access to applications, such as mail (Gmail, Outlook 365); productivity applications (Office 365, G Suite), etc. In some courses, such as web apps and services, our students may develop applications that fall in the SaaS category, using PaaS or IaaS as a mechanism for doing so.
- In **Functions-as-a-Service (FaaS)**, the developer deploys a function to be executed when triggered by some event, in a stateless fashion (although the function can communicate with databases to preserve state [11]). Serverless functions can act as a gateway to a cloud computing system, taking in data delivered over HTTP, and act as pipes, connecting one part of a cloud computing system to another. So, for instance, data

from a mobile phone could be sent to a serverless function, filtered and processed, then sent to storage, then on to a query service and then on to a visualization tool, all automatically. The CSPs permit serverless functions to be written on a local machine and then deployed using a command-line utility, but unlike with PaaS, they also offer the ability to write the entire function in a web browser window. As such serverless functions are by far the easiest of the service categories to get started with, and lend themselves well to quick but impressive demos in almost any CS class.

- **Backend-as-a-Service (BaaS)** encompasses a suite of services that mobile and web apps often need: data storage, data analytics, and push notifications. CSPs provide APIs to interact with SQL and NoSQL databases as well as query and data-visualization tools specifically engineered for the large amounts of data that CSP customers might generate. To demonstrate the power of their systems, and give customers something to practice on, CSPs provide very large, interesting data sets to explore. For educators, these collections are a real boon, and could find application in any number of courses.

Services offered by CSPs can be used individually, but in many cases, they are connected via a pipeline, based on publish-subscribe paradigm. For example, an iOS app might publish a topic (which includes data) to a pub-sub service which would trigger a serverless function subscribed to that topic to write data to a database and/or trigger a subsequent query.

## 4 Evaluating Cloud Service Providers from a Pedagogic Perspective

The three largest CSPs are Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure (AZR). Which to choose? In a class focused on cloud computing it would be appropriate and educational to examine all three. However, familiarizing students with even one of these is no trivial undertaking, and managing student accounts is cumbersome, so it is best to choose one to get started. Academic research offers comparisons of CSPs in terms of performance [1], advice on how to select CSPs [15], and how to integrate CSPs into the classroom [4]; and discussions have carried over into industry [12, 6, 7, 13], but there has been little discussion from a computer science education perspective.

In terms of market share, AWS predominates, so AWS experience might make graduates more attractive to more employers. However, with the dearth of cloud computing courses in CS, most employers would be satisfied with experience with any CSP.

Apart from marketshare, the strongest argument in favor of AWS is its education program, AWS Educate ([aws.amazon.com /education/awseducate](https://aws.amazon.com/education/awseducate)). It allows students to sign up for one of two accounts - one that requires a credit card, the other that does not (but caps access to some services). AWS's training resources are all housed here: this includes open source and AWS-developed courses, as well as certification opportunities. Students can also be grouped into virtual classrooms: the instructor must specify how many credits they wish to allocate each student, and can choose a template (Cloud Basics, Big Data, Machine Learning and AI, Building Scalable Websites, Serverless Computing, etc.), that restricts the services on offer, to avoid overwhelming students with choices. AWSEducate also integrates with popular learning management systems.

Google has a somewhat similar program, Google For Education ([edu.google.com/programs](https://edu.google.com/programs)). It too provides a mechanism to apply for credits and houses numerous teaching resources, but Amazon's product, based on the third-party site Vocareum, appears more comprehensive and is better integrated. Whereas setting up courses and a cloud-sandbox within AWS Educate just requires one application, Google For Education, based on the third-party site Qwiklabs, only allow "trainers" to set up classrooms, and becoming a trainer is an onerous process.

Google's greatest strength, from an academic perspective, comes from its ease-of-use and supporting documentation. Having compared platforms side-by-side, we find that GCP's user interface is particularly easy to navigate, uncluttered, and designed for efficiency. For instance, when displaying services, GCP's hamburger menu neatly organizes services hierarchically; AWS shows all of its services at once, daunting and not as easy to navigate. AZR is somewhere in between, with a smaller number of services displayed by default, but no discernable order to them.

When creating a virtual machine, AWS's EC2 (Elastic Computing 2, a name which does not reflect its purpose), presents the user with some arcane choices (perhaps relevant to a professional, not necessary for a student). GCP, whose Virtual Machine service is hard to mistake for anything else, provides just three buttons - Create, Import, or Take the QuickStart - and the ensuing screen when the user clicks Create is a paragon of good design. [[AZR VM]]

In many cases, AWS forces the user to make a choice, whereas GCP provides reasonable default values. As a concrete example, AWS requires the user choose a name and pick a permissions role (that may take several minutes to generate) when creating a serverless function. GCP provides a default name (function-1) and permissions are predetermined, making for a smoother, more efficient workflow. Lastly, GCP's documentation is particularly clearly written, and, as befitting a search engine giant, well organized.

Beyond materials provided on site, Google Engineers star in numerous YouTube videos that walk users through GCP's features; and its Qwiklabs, apart from the difficulty in setting up classrooms, are an excellent hands-on starting point.

A final issue involves billing. Both AWS and GCP offer students a maximum of 50 credits, capped but credit-card-free, an absolute necessity in an academic setting.

## 5 Applications in the Classroom

There is a fair degree of complexity associated with integrating cloud computing exercises into a course. In general, early programming courses and foundational math and theory courses are not typically target classes. Instead, the most suitable areas of the curriculum for engaging cloud computing are likely to be mid-to-upper level elective or application-focused courses. Theses may include higher-level undergraduate courses as well as core and elective courses in a graduate program. We have identified specific lesson topics for a variety of courses.

### 5.1 Internet of Things

Internet of Things (IoT) applications have the potential to generate massive amounts of sensor data, known as telemetry. Cloud computing, with services that can ingest, store, and query petabytes of data, is a perfect fit. Typically, IoT access to the cloud involves a CSP service, called an IoT hub/bridge/core (the name is CSP-dependent). Once an IoT device is registered, it can send telemetry to the hub/bridge/core, which uses a pub-sub strategy to shunt it to where it is needed. Some of the CSPs allow users to construct virtual pipelines: the IoT hub/bridge/core publishes telemetry; the pipeline, as a subscriber, is alerted that data is available; it then delivers it to data storage or data analytics services for further processing. For finer control (e.g., to monitor telemetry and discard outliers), it is possible to publish to a serverless function, which then delivers it to other parts of the cloud system.

In our classes, we use an IoT prototyping system made by Particle [19], the Photon. It abstracts away many messy and potentially insecure connection details. Users connect to the Particle Cloud over HTTPS, can retrieve the values of firmware variables from the device using GET requests, and invoke firmware functions using POST requests. Device management can be done using a web-based console. The Particle Cloud offers no storage / data analytics capabilities. Fortunately, it is integrated with AZR and GCP, offering webhooks to connect to those services. The user publishes an event to the Particle

```
// Photon firmware
String telemetry = String::format( "{\"temp\":%.1f, \"humidity\":%.1f}", temperature, humidity);
Particle.publish("weather", telemetry); // invokes integration
```

Particle cloud  
integration

GCP Pub/Sub  
Topic

Topics			<a href="#">CREATE TOPIC</a>	<a href="#">DELETE</a>
Filter table				
<input type="checkbox"/>	Topic ID ↑	Encryption	Topic name	
<input type="checkbox"/>	incoming-weather	Google-managed	projects/weather-1852/topics/incoming-weather	

Cloud Function

<input type="checkbox"/>	Name ^	Region	Trigger	Runtime	Executed function	
<input checked="" type="checkbox"/>	handleWeather	us-central1	Topic: incoming-weather	Node.js 8	handleWeather	⋮

Datastore

Datastore		Entities	<a href="#">CREATE ENTITY</a>	<a href="#">COPY</a>
Entities		<a href="#">QUERY BY KIND</a> <a href="#">QUERY BY GQL</a>		
		Kind weather		
<input type="checkbox"/>	Name/ID ↑	temperature		
<input type="checkbox"/>	id=5629499534213120	25		

Figure 1: Adding cloud computing to an IoT project.

Cloud, embedding a JSON object, triggering a webhook, called an integration in Particle's parlance.

For AZR, telemetry enters "through the front door", triggering devices in

Microsoft's IoT hub. With GCP, Particle bypasses Google's IoT Core service, and instead uses pub-sub. In both cases, IoT management is still primarily done on the Particle cloud, thus melding the simplicity of a vendor-supplied IoT management system with the computing power of a CSP.

Figure 1 shows the steps in the process: firmware triggers a Particle integration; the Particle integration triggers a GCP Pub/Sub topic; the Google Pub/Sub topic triggers a Cloud Function; and from that function it is possible to send data to any GCP service (in our case, Google Datastore). The path is admittedly serpentine, and it takes an entire class period (80 minutes) to get everyone through the initial setup. But once that has been accomplished, the students can generally repeat the process on their own without difficulty.

Students find IoT challenging because they must deal with both hardware and software, and without specialized hardware must rely on print statements to debug firmware. However by the time that we have introduced cloud computing into the IoT course, just over halfway through the semester, students have already more or less mastered the basics, and have already worked with some IoT-specific cloud systems (e.g., Ubidots and ThingSpeak). These offer nowhere near the power of AWS, GCP, or AZR, but are significantly less complex. Once we turn to a CSP, students again must resort to print statements to debug serverless functions and web apps by scanning through logs. This is done twice, on the Particle Cloud to ensure that the data is being sent, and on the CSP to see if the incoming telemetry is being properly parsed and sent on. We walk through several worked examples in class, and trace the path of the telemetry from start to finish, before we make assignments, which typically involve the same data flow, but different sensor data and different analytical requirements.

## 5.2 Big Data

The cloud offers a perfect place for students to explore big data processing, architectures, and technologies. Massive amounts of information are being ingested from IoT devices (as noted above), as well as on-line information systems, point-of-sale applications, social media, public data, historical data, government and research sources and more. These massive amounts of highly valuable and varied information are referred to as 'big data', and present a unique set of challenges for data engineers that need a place to hold and process this information. In general, data can be characterized as:

- **data at rest.** This includes *data lakes*, which hold raw information in native format using a flat architecture, and *data warehouses*, which hold cleaned, processed, filtered information using a hierarchical or organized format, in a centralized, scalable location.

- **data in motion.** This refers to data moving across a network, often processed and piped in real-time as the information is generated.
- **data in use** refers to data held in memory. This enables faster processing and minimizes the amount of time spent reading and writing data.

Standard big data technologies, familiar to practitioners in the field, include Apache Hadoop (with the Hadoop Distributed File System, HDFS, and MapReduce and its resource manager YARN), for batch processing; Apache Flink, for processing streaming data; Apache Spark, for processing both data in motion and data at rest, using just one code base.; and finally Apache Kafka, for piping and processing data.

Each CSP offers its own unique solutions and technologies to host these tools and services, in combination with their own proprietary offerings.

For illustration (all the CSPs have similar offerings), we will consider GCP. GCP offers DataProc clusters, a custom Hadoop-like service. DataProc offers big data students a chance to design solutions that separate redundant, fault-tolerant storage with distributed computing processes running in parallel. Students can fire up a DataProc cluster, configuring the mode with master and worker nodes. The master node includes YARN, the HDFS NameNode, and job drivers. Each worker node includes a YARN NodeManager and HDFS DataNode. Students can configure replication factors, number of cores per machine, disk size, and more. Students can then use their clusters to submit Spark jobs. Spark jobs can be connected to Google's BigQuery or used with traditional Hadoop tools such as Hive and Pig for extract, transform, and load (ETL) processing. BigQuery offers an extremely fast processing engine with usage costs based on storage, rather than CPU cycles.

Huge public datasets, some over a billion records, are available in the GCP Public Dataset Program. Students can explore the data with the web UI (as shown in Figure 2), via REST APIs, the command line, or via Python, Java, and .NET client libraries. The first terabyte of data is free each month, offering a chance to explore exciting data sets that include genomics, weather, cryptocurrency, demographics, and more.

### 5.3 Machine Learning (ML)

Processing cloud-scale big data requires advanced techniques to gain timely, useful insights. Cloud providers offer a variety of tools to explore machine learning on small or massive datasets. Machine learning (ML), a subset of the field of artificial intelligence (AI), is increasingly being covered in business courses, information systems courses, advanced Excel classes, programming classes, and more. ML has application in chemistry, physics, biology, finance, economics, healthcare, agriculture, energy production and distribution, risk



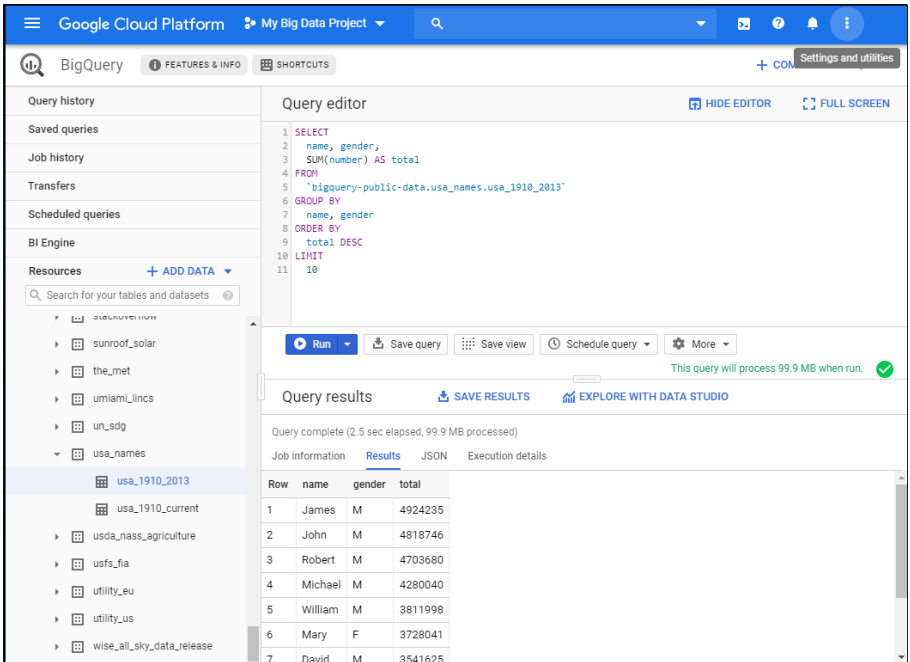


Figure 2: Using cloud computing to query big data.

management and more. The availability of open-source libraries for Python, R, Excel, Java, .NET, and many other languages makes the introduction of this key technique a powerful addition to many courses.

Machine learning applies to a wide variety of approaches for gaining insights from information when processing instructions are not provided explicitly, and we instead provide patterns, models, rules, or other general approaches which are then applied using a variety of methods. There are a variety of approaches to ML including supervised, unsupervised, reinforcement, self-learning and more. Modeling techniques include statistically-based regression analysis, rule-based decision trees, supervised support vector machines for classification and regression based on training examples, artificial neural networks, Bayesian networks or influence diagrams based on directed acyclic graphs, genetic algorithms employing mutation and crossover, and more.

GCP provides tools for ML developers and data scientists to implement ML techniques and explore the results. For example, GCP offers Cloud ML Engine as part of its AI Platform. Pre-trained models are available as building blocks, accessible through APIs from most popular programming languages,

making it easy for students to get started using ML. GCP's AutoML enables training even without ML expertise and Deep Learning VM Images are already set up with the most popular tools, including PyTorch for computer vision and natural language processing applications, TensorFlow, the open source library that uses data flow graphs to create deep neural networks, and scikit-learn for Python machine learning projects for classification, regression, and clustering.

## 5.4 Mobile Computing

BaaS is also known as Mobile Backend-as-a-Service (MBaaS), underlining the importance of cloud computing to mobile applications: most mobile apps have at least some cloud requirement, if only to backup data. Compared to the challenges of IoT, MBaaS is straightforward: all the CSPs provide direct access to their major services via platform-specific APIs. This contrasts favorably with the circuitous route that we must use in IoT (cf. 5.1). Rather than trigger a pub-sub event that in turn triggers a pipeline or serverless function, which in turn interacts with a service, we can make a native API call in Swift (iOS), or Java or Kotlin (Android) to deliver data directly to a service. As expected, the APIs differ among the CSPs. In a comparison of the CSPs various "Getting Started" code samples for writing to a NoSQL database in iOS, GCP requires by far the least amount of code: 4 statements, excluding error handling (as shown in the following code). As is common with NoSQL databases, it is not necessary to create collections ahead of time, nor specify a schema for the documents: it can all be done dynamically when the app first connects.

```
FirebaseApp.configure()
// an in-app representation of Firestore DB
let db = Firestore.firestore()
// a collection of documents, equivalent to SQL records
var ref: DocumentReference? = nil
ref = db.collection("users")
    .addDocument(data:
["first": "Grace","last": "Hopper","born": 1906])
{ err in
    if let err = err {
        print("Error adding document: \(err)")
    } else {
        print("Document added with ID: \(ref!.documentID)")
    }
}
```

## 5.5 Developing Web Apps and Services

All major CSPs offer options for hosting server-side and full-stack applications on the web. GCP for example, offers App Engine, which hosts applications written in Python, Node.js (server-side JavaScript), PHP, Java, C#, Go, Ruby, and more. The AZR equivalent of App Engine is, appropriately, Web Apps; AWS offers the more cryptically named Elastic Beanstalk.

Most full-stack apps also require a central data store that the CSPs can also provide. For example, GCP offers MongoDB Atlas hosting as well as Cloud SQL. MongoDB is a popular NoSQL document data store and the GCP option offers a free 3-node replica set. Cloud SQL options include low-priced MySQL, PostgreSQL, and SQL Server options.

This example assumes students have a cloud repository with a web app project to be deployed. After enabling their billing account as described above, have students google "Installing Google Cloud SDK" and choose an installation option appropriate for their operating system. Download and run the SDK installer. Add an app.yaml file to the root project folder:

```
runtime: nodejs10
env: standard
basic_scaling:
  idle_timeout: 600s
  max_instances: 1
resources:
  cpu: 1
  memory_gb: 0.5
  disk_size_gb: 10
```

The suggested settings help keep costs low. If deploying a Node project, you might need to set the Node version to "^10" so it can run version 10 or higher. We added one dependency so we could create log files in the cloud: "@google-cloud/logging-winston": "^3.0.0".

The three commands below create a deployment site in the cloud by performing three key tasks: (1) creating a cloud project (2) verifying the cloud project (3) initializing an App Engine app in the project. Choose a unique project name using kabob case (lowercase with hyphens) to use in the following commands.

```
gcloud projects create project-name --set-as-default
gcloud projects describe project-name
gcloud app create --project=project-name
```

When initializing the app, you will get asked to choose a region. Choose a nearby region with a free option. The following commands will deploy/push your project code up to the cloud, then open a browser to your running app. During the deploy command, when you get an error about enabling billing, just follow the URL provided to enable billing for the project.

```
gcloud app deploy
gcloud app browse
```

## 6 A Guide to Getting Started

Working with the major CSPs in courses typically involves requesting an educational account with credits that allow students to practice with a set of services that normally must be backed by a credit card. Beyond these, users interested in higher-end services will need to provide a credit card, and can expect to incur some charges. Educational accounts provide enough of a credit buffer to get started. Students must remember to shutdown and/or delete configured resources after use to avoid prematurely draining their accounts.

Each CSP, as noted earlier, has an education portal (Google Cloud For Education, AWSEdulate and Azure For Education), where the instructor must apply for credits by providing such information as the course title, number, description, dates, numbers of students and staff, etc. Responses are typically returned in a couple of business days. Forward the link to students (or, in the case of AWS Educate, upload a spreadsheet containing email addresses). Once students provide information and accept terms, they will get an email confirmation with a link to get started.

All the CSPs provide numerous tutorials to help users get oriented to the user interface and acquainted with various services and APIs. We recommend assigning these tutorials as assignments, with students turning in links to their created projects. If these are due before the CSPs are to be introduced in lecture, it frees up class time for more important and interesting topics.

As mentioned previously, another highly recommended resource is Qwiklabs. Both GCP and AWS (but not AZR) have partnered with this company to produce a series of no-cost labs that lead viewers through services on their respective platforms. Participants are provided with temporary accounts in which to do their work, so it has no impact on usage. The labs are updated regularly, so what the student sees in the lab video matches what they see in the browser.

## 7 Conclusions

From its beginnings in the 20th century, cloud computing has emerged to be a major force in business and academia. This paper has provided a concise guide with specific recommendations for introducing cloud computing elements into several key areas of the curriculum.

## References

- [1] Srikanth Kandula Ang Li, Xiaowei Yang and Ming Zhang. Cloudcmp: comparing public cloud providers. *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement (IMC '10)*, pages 1–14, 2010.
- [2] Hongyu Pei Breivold and Ivica Crnkovic. Cloud computing education strategies. In *2014 IEEE 27th Conference on Software Engineering Education and Training (CSEET)*, pages 29–38. IEEE, 2014.

- [3] Justin Cappos, Ivan Beschastnikh, Arvind Krishnamurthy, and Tom Anderson. Seattle: a platform for educational cloud computing. In *Acm sigcse bulletin*, volume 41, pages 111–115. ACM, 2009.
- [4] Singha Chaveesuk, Phayat Wutthirong, and Wornchanok Chaiyasoonthorn. Cloud computing classroom acceptance model in thailand higher education's institutes: A conceptual framework. In *Proc of 2018 10th Intl Conf on Info Mgmt and Engg*, pages 141–145. ACM, 2018.
- [5] Compaq. *Internet Solutions Division Strategy for Cloud Computing*. Available at [https://s3.amazonaws.com/files.technologyreview.com/p/pub/legacy/compaq-cst\\_1996\\_0.pdf](https://s3.amazonaws.com/files.technologyreview.com/p/pub/legacy/compaq-cst_1996_0.pdf), Retrieved 17-Nov-2019.
- [6] Larry Dignan. *op cloud providers 2019: AWS, Microsoft Azure, Google Cloud; IBM makes hybrid move; Salesforce dominates SaaS*. Available at [zdnet.com/article/top-cloud-providers-2019-aws-microsoft-azure-google-cloud-ibm-makes-hybrid-move-salesforce-dominates-saas/](http://zdnet.com/article/top-cloud-providers-2019-aws-microsoft-azure-google-cloud-ibm-makes-hybrid-move-salesforce-dominates-saas/), Retrieved 10 November 2019.
- [7] Brian Turner Drake, Nate. <https://www.techradar.com/news/best-cloud-computing-service>. Available at <https://www.techradar.com/news/best-cloud-computing-service>, Retrieved 2-Nov-2019.
- [8] National IA Education and Training programs (NIETP). *2019 Knowledge Units*. NSA Cybersecurity, 2019.
- [9] Flexera. *Cloud Computing Trends: 2019 State of the Cloud Survey*. Available at [flexera.com/blog/cloud/2019/02/cloud-computing-trends-2019-state-of-the-cloud-survey/](http://flexera.com/blog/cloud/2019/02/cloud-computing-trends-2019-state-of-the-cloud-survey/), Retrieved 16-Nov-2019.
- [10] Alice Fothergill and Kathryn Feltey. "i've worked very hard and slept very little": Mothers on tenure track in academia. *Journal of the Motherhood Initiative for Research and Community Involvement*, 5(2), 2003.
- [11] Martin Fowler. *Serverless Architectures*. Available at [martinfowler.com/articles/serverless.html](http://martinfowler.com/articles/serverless.html), Retrieved 17-Nov-2019.
- [12] Cynthia Harvey and Andy Patrizio. *Aws vs. azure vs. google: Cloud comparison*. Datamation, 2017.
- [13] Cynthia Harvey and Andy Patrizio. *AWS vs. Azure vs. Google: Cloud Comparison*, 2017.
- [14] Mohamud Sheikh Ibrahim, Norsaremah Salleh, and Sanjay Misra. Empirical studies of cloud computing in education: a systematic literature review. In *International Conference on Computational Science and Its Applications*, pages 725–737. Springer, 2015.
- [15] Farookh Khadeer Hussain Omar Khadeer Hussain Le Sun, Hai Dong and Elizabeth Chang. Cloud service selection: State-of-the-art and future research directions. In *Journal of Network and Computer Applications*, pages 134–150. ACM, 2014.
- [16] Math and Denison University CS Dept. *George Stibitz*. Available at <http://stibitz.denison.edu/>, Retrieved 17-Nov-2019.

- [17] Saju Mathew. Implementation of cloud computing in education-a revolution. *International Journal of Computer Theory and Engineering*, 4(3):473, 2012.
- [18] Davison M Mupinga and George R Maughan. Web-based instruction and community college faculty workload. *College Teaching*, 56(1):17–21, 2008.
- [19] Particle. *particle.io*. Available at <https://www.particle.io/>, Retrieved 17-Nov-2019.
- [20] Antonio Regaldo. *Who Coined 'Cloud Computing'?* Available at <https://www.technologyreview.com/s/425970/who-coined-cloud-computing/>, Retrieved 17-Nov-2019.
- [21] Eric Roberts. Computing education and the information technology workforce. *ACM SIGCSE Bulletin*, 32(2):83–90, 2000.
- [22] Eric Schmidt. *Search Engine Strategies Conference: Conversation with Eric Schmidt hosted by Danny Sullivan*. Available at <https://www.google.com/press/podium/ses2006.html>, Retrieved 17-Nov-2019.

# Effect of User Involvement in Information Systems Capstone Course: A Case Study\*

*Cindy Zhiling Tu and Joni Adkins*  
*School of Computer Science and Information Systems*  
*Northwest Missouri State University*  
*Maryville, MO 64468*  
*{cindytu,jadkins}@numissouri.edu*

## Abstract

This paper conducts a case study in a Master of Information Systems capstone course context to investigate how user involvement affects the performance of the capstone projects. After investigating user involvement in the IS capstone course, we find there are different forms of user involvement and different factors decide the degree of user involvement. Continuous user involvement throughout the capstone project life cycle benefits the capstone projects.

## 1 Introduction

Many information systems (IS) degree programs culminate in functional skills in a capstone course to integrate the concepts [15]. Placed at the end of the curriculum, a capstone course allows students to assess and share their achievement of the program's outcomes [8, 19]. IS capstone course can facilitate a student-executed IS project where teams would work for a real industry client to address a real business problem. Through the capstone projects, IS program goals are reviewed and students communicate their academic accomplishment to professional peers [4].

IS capstone course covers the full spectrum of information systems development from conceptualization, analysis and design, to prototyping and/or implementation. In a capstone course, students usually work in small teams and

---

\*Copyright ©2020 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

finish one independent project within one semester. Students are compelled to become self-directed learners through a structured reflection [22]. Due to the nature of small-group student projects and the short time period, many instructors have incorporated agile methodologies into their capstone courses [14, 20].

Agile method advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages rapid and flexible response to change [3]. Studies revealed that Agile approaches were more appealing to student teams and customer expectations were better met by the final software product in capstone projects [21]. Customer collaboration is one of the core values of the Manifesto for Agile Software Development. Intense communication between different stakeholders and rapid feedback based on regular delivery of products are the main features of Agile processes [3]. Therefore, how to collaborate with the customer is critical for the success of capstone projects using Agile methods.

User involvement is beneficial in information system development. User involvement provides useful information about users' needs and increases the understanding of users' values [12]. Students in IS capstone projects work with real-world clients who are their product users. However, in most cases, IS capstone projects may not cover the full system development processes. Instead, the final product of capstone projects is a prototype without actual implementation. Different from the real system product context, does user involvement provide the same benefits to the Agile capstone project? How will user involvement affect students' satisfaction? Little research has been done on the relationship between user involvement and the IS capstone course. This research focuses on the effect of user involvement on students executed capstone projects by conducting a case study.

The rest of the paper is organized as follows. In the next section, the conceptual background is given. Then, a case study is presented, followed by the results and discussion. Finally, the conclusion section is provided.

## 2 Conceptual Background

User involvement is a widely accepted principle in the development of useful and usable software systems [12]. Traditional IS participation theory states a positive link between user involvement and system success which is defined in terms of system quality, user information satisfaction, user acceptance, and system use [16]. User knowledge and involvement are identified as key elements of product quality [11]. User involvement improves quality through precise requirements, enhancing alignment between developers and users, creating a positive attitude toward the system among users, and enabling effective use of



the system [1]. Depending on the level of task complexity and system complexity, user involvement can result in improved user satisfaction [17]. A lack of user involvement is found to be associated with failed software projects [13].

User involvement may have different forms, which are characterized as informative, through consultative, to participative [12]. Informative users provide information, consultative users comment on a predefined service or range of facilities, while participative users influence decisions relating to the whole system [5]. Thus Users can be involved as information providers, commentators, or objects for observations, depending on whether they influence decisions, or whether they participate in development [12].

Empirical studies find that user involvement is decided by the following factors: communication and relationship, professional and technical competence, users' IT skills and awareness, user organization's technical infrastructure and working environment, expertise in the user problem domain, and contract awarding procedure [2]. Because developers and users have different vocabularies, interests, and values, their communication and interplay are complicated [12]. The developers and users may have conflict perceptions regarding user involvement in system development. Impediments to user involvement in system development are identified as: user's busy schedule, lack of user confidence and motivation, lack of understanding of the task mode and implementation constraints, user's lack of education and knowledge on systems issues, and excessive time spent by the developer in contacting users regarding the problem domain [23]. Besides, two contextual factors, task complexity and system complexity, determine the need for user involvement [17].

Regarding the timing of when users should participate throughout the project life cycle, many studies find that continuous user involvement benefits system development [6, 7, 9, 10]. It is found that user involvement is most efficient and influential in the early stages of system development because more accurate user requirements were gathered and user needs were better reported [12].

In summary, user involvement has different forms and different factors decide the degree of user involvement. Continuous user involvement throughout the development life cycle leads to improved product quality and user satisfaction.

### 3 Case Study

User involvement was investigated in a real capstone course context utilizing a case study to assess the effect of user involvement on the student-executed capstone projects. A regional public state university in the Midwest created a Master of Science degree in Information Systems (MSIS) and included a one-

semester capstone project course. This case study was conducted in spring 2019, from January to May 2019.

This capstone course is designed to help students integrating the knowledge gained during the MSIS program. Ten students were assigned to two project teams. Two on-campus departments served as clients of two real-world projects. One client was the Human Resource (HR) department, considering to replace the current HR management information systems. The other client was the Career Center, trying to rebuild the university's online recruitment system. Both clients had realized the ineffectiveness and inefficiency of current information systems and decided to rebuild or purchase a new system. The student project teams would need to provide necessary decision tools to help their clients determine whether to buy or build. To accomplish this goal, both teams went through the information systems development processes including user requirement collection, analysis and design, documentation, and prototyping.

Students applied Scrum methodology to complete their projects. Scrum is a popular Agile software development process for small teams [18]. Scrum is made up of 4 sprints which are 3 weeks of duration. Teams must complete a set of project tasks during every sprint. At the end of each sprint, the sprint outcomes or potentially deployable prototypes were demonstrated to the clients. The clients were able to give direct feedback to the sprint outcomes thus students could improve in the following sprint. The project teams made a formal presentation that showcased the final products to the clients, along with a final report that included the documents prepared during the development processes.

The capstone projects were managed through on-going consultation with the instructor. Students had daily scrum meetings and regular class meetings. Clients were expected to be committed to communicating with the students promptly. Clients were involved in the capstone projects through interviews, sprint review meetings, and email communications. The two projects were similar in the development processes but were completed differently. For Team 1, the HR department delegated an IT specialist in the IT department to be the contact. The IT specialist initiated the students' capstone project but she did not work in the HR department thus she had limited knowledge about HR management processes. For Team 2, the Director and Associate Director of the Career Center were the main contacts. They knew exactly what they wanted for the new system. We observed the client involvement situations in the whole process and summarized them in Table 1 and Table 2.

At the end of the semester, all students took a survey about the capstone course. There were 3 questions about the client. The two teams had a significant difference in their perceptions of client involvement. The questions used a

Sprint	Planned Tasks	Client Involvement Needed	Actual Client Involvement	Consequences
	Project kick-off	Kick-off meeting	The IT specialist and all HR staff participated. A meeting plan was decided.	Good start.
1	To review the current system and collect user requirements from the client.	Need to interview HR staff to understand the business processes and then figure out the functional requirements for the new system.	The IT specialist tried to schedule interviews but failed.	Students did not get information from the HR department. They reviewed the current system by themselves.
2	To prepare requirement specifications for the new system.	Need HR staff to give insight into the functional requirements.	Only the IT specialist gave some feedback to students' sprint outcomes.	Students prepared requirement specifications according to their understanding of the HR system.
3	To do a gap analysis. To analyze, design and prototype part of the functions.	Need to meet with HR staff to check the gap between the current system and their operation convenience. Also, a vendor system demonstration meeting was scheduled.	Students met with individual HR staff to collect their requirements and opinions. However, the schedule was not efficient. The vendor meeting was canceled.	Gap analysis did not complete due to the tardiness. System analysis and design could not finish because students did not know the real business processes well. The prototype was not done.
4	To finalize gap analysis. To analyze, design and prototype part of the functions.	Need to meet with HR staff to get their feedback about the analysis and design.	HR staff was not available. Students got some feedback from the IT specialist.	Students finished the gap analysis based on their observation to real business processes and their understanding of the new system. System analysis and design continued but prototyping was not done.
	To finalize the project.	Need to present the final product to the client.	HR staff was not available. Only the IT specialist attended the final presentation.	Students only completed requirement analysis and part of the system design. The gap analysis report was done. No prototype.

Table 1: Team 1 – Human Resources Management Information System Project

<b>Sprint</b>	<b>Planned Tasks</b>	<b>Client Involvement Needed</b>	<b>Actual Client Involvement</b>	<b>Consequences</b>
	Project kick-off	Kick-off meeting	Both Directors attended. A meeting plan was decided.	Good start.
1	To review the current system and collect user requirements from the client.	Need to interview the client to understand the business processes and then figure out the functional requirements for the new system.	Several interviews were conducted. The client replied to students' questions via emails promptly.	Students completed requirement specification.
2	To do vendor analysis. To analyze, design and prototype part of the functions.	Need to meet with several vendors.	The client authorized the students to contact 7 vendors who showed their system demos to students.	Students got sufficient information from 3 vendors. A sprint prototype was done.
3	To continue vendor analysis. To analyze, design and prototype part of the functions.	Need to meet with several vendors. Need the client to review their sprint outcomes.	Students met with the other 2 vendors. Both Directors attended the sprint review meeting and gave their feedback to the prototype.	Students got sufficient information from vendors and completed vendor analysis for the client. Students got feedback on their prototype.
4	To finish system analysis, design, and prototyping.	Need the client to review their sprint outcomes.	Both Directors attended the sprint review meeting and gave their feedback to the prototype.	Students got feedback on their prototype.
	To finalize the project.	Need to present the final product to the client.	Both Directors attended the final presentation and evaluated the students' work.	Students completed the functional prototype.

Table 2: Team 2 – Online Recruitment System Project

5-point scale with 5 being strongly agreed and 1 being strongly disagreed. The mean scores of Team 1 for all questions were lower than Team 2. The results are listed in Figure 1.

Questions	Team 1 Mean (n=5)	Team 2 Mean (n=5)
Our team worked well together with our client.	4.6	5
We had an open dialogue with our client during the project.	4	5
The client was effectively involved in our project.	3.6	5

Figure 1: Mean Scores from Survey

## 4 Discussion

After studying the above case, we can see how user involvement affected the performance of capstone projects. First, students in Team 1 felt their client was not effectively involved in their project. Team 1 did not complete their system prototype mainly because they did not have enough time to do it. Their client was not available for most scheduled meetings and thus delayed the whole project’s progress. The team could not get the necessary information for many tasks. The final product of their project was incomplete. Students in Team 2 were very satisfied with their client involvement. Team 2 finished all product backlogs as planned. In addition to the system prototype, this team also provide their client with a vendor analysis which helped their client make the make-or-buy decision. Therefore, user involvement is associated with the performance of the capstone project.

Second, there were different user involvement forms in this case. The client of Team 1 should be HR staff, but only the IT specialist was involved. She did not know the HR business process very well and could not influence the HR department’s decisions. She could provide some information about the system but not enough. She could not provide any consultation on the system. Team 1 had partially informative user involvement. The client of Team 2 includes the two Directors of the Career Center, who knew their business processes very well and could make decisions in the department. They provided the team with all the needed information, commented on their analysis report and prototypes, and gave consultation in different ways to the team. Since the project was not implemented, the team did not need participative user involvement. The client’s informative and consultative user involvement promoted project success.

Third, different factors influenced user involvement. The HR staff always had very busy schedules and it was hard to meet all of them together. The IT specialist had little expertise in the HR domain even though she tried to be involved actively. The HR staff was reluctant to spend excessive time in contacting the project team regarding their HR problems. Furthermore, compared to the online recruitment system, the HR system was much more complicated and the project tasks were more complex. All these factors impeded the client's involvement in Team 1.

Finally, continuous user involvement is necessary for capstone projects. In the Team 1 project, the HR staff was involved in the early stage with plans in the kick-off meeting. However, they did not continue the active involvement in the following stages. The client of Team 2 kept participating in all sprint review meetings and provided the team with prompt support during the whole project duration. Continuous involvement guaranteed the quality of the project products.

## 5 Conclusion

A case study was conducted in a Master Capstone course context to investigate how user involvement affects the performance of the Capstone projects. To conclude, user involvement has the same effect on capstone projects as on information system development. When we investigate user involvement in the IS capstone course, we find there are different forms of involvement and different factors decide the degree of involvement. Continuous user involvement throughout the capstone project life cycle benefits the capstone projects. These findings are consistent with prior literature.

## References

- [1] Ulrike Abelein. *User-Developer Communication in Large-Scale IT Projects (Doctoral dissertation)*. PhD thesis, University of Heidelberg, 2015.
- [2] Boluwaji Akinnuwesi, Faith-Michael Uzoka, Stephen Olabiyisi, Elijah Omidiora, and Paula Fiddi. An empirical analysis of end-user participation in software development projects in a developing country context. *The Electronic Journal of Information Systems in Developing Countries (EJISDC)*, 58:1–25, 07 2013.
- [3] Agile Alliance. Agile 101. <https://www.agilealliance.org/agile101>. Accessed November 2019.
- [4] Joseph B. Cuseo. Objectives and benefits of senior year programs. In John N. Gardner and Gretchen Van der Veer, editors, *The Senior Year Experience: Facilitating Reflection, Integration, Closure and Transition*. Jossey-Bass Inc., San Francisco, CA, 1998.

- [5] Leela Damodaran. User involvement in the systems design process – a practical guide for users. *Behaviour & Information Technology*, 15(6):363–377, 1996.
- [6] Katrien De Moor, Katrien Berte, Lieven De Marez, Wout Joseph, Tom Deryckere, and Luc Martens. User-driven innovation? challenges of user involvement in future technology analysis. *Science and Public Policy*, 37(1):51–61, 2010.
- [7] Joyce Fortune and Diana White. Framing of project critical success factors by a systems model. *International Journal of Project Management*, 24(1):53–65, 2006.
- [8] Eric H. Hobson, Philip E. Johnston, and Alisa J. Spinelli. Staging a reflective capstone course to transition pharmd graduates to professional life. *American Journal of Pharmaceutical Education*, 79(1):1–10, 2015.
- [9] Rashina Hoda, James Noble, and Stuart Marshall. The impact of inadequate customer collaboration on self-organizing agile teams. *Information and Software Technology*, 53(5):521–534, may 2011.
- [10] Jack Shih-Chieh Hsu, Tung-Ching Lin, Guang-Ting Zheng, and Yu-Wen Hung. Users as knowledge co-producers in the information system development project. *International Journal of Project Management*, 30(1):27–36, 2012.
- [11] James J Jiang, Gary Klein, Hong-Gee Chen, and Laura Lin. Reducing user-related risks during and prior to system development. *International Journal of Project Management*, 20(7):507–515, 2002.
- [12] S. Kujala. Effective user involvement in product development by improving the analysis of user needs. *Behaviour & Information Technology*, 27(6):457–473, 2008.
- [13] S. Kujala, M. Kauppinen, L. Lehtola, and T. Kojo. The role of user involvement in requirements quality and project success. In *Proceedings of IEEE international requirements engineering conference*, RE’05, pages 75–84. IEEE, 2005.
- [14] Alejandra Magana, Ying Ying Seah, and Paul Thomas. Fostering cooperative learning with scrum in a semi-capstone systems analysis and design course. *Journal of Information Systems Education*, 29(2):75–91, 2018.
- [15] Michael Maloni, Pamila Dembla, and J. Anthony Swaim. A cross-functional systems project in an is capstone course. *Journal of Information Systems Education*, 23(3):283–296, 2012.
- [16] M. Lynne Markus and Ji-Ye Mao. Participation in development and implementation - updating an old, tired concept for today’s is contexts. *Journal of the Association for Information systems*, 5(11):514–544, 2004.
- [17] James D. McKeen and Tor Guimaraes. Successful strategies for user participation in systems development. *Journal of Management Information Systems*, 14(2):133–150, 1997.
- [18] Linda Rising and Norman S. Janoff. The scrum software development process for small teams. *IEEE Software*, 17(4):26–32, 2000.

- [19] Randolph E. Schwering. Optimizing learning in project-based capstone courses. *Academy of Educational Leadership Journal*, 19(1):90–104, 2015.
- [20] Toni Taipalus, Ville Seppänen, and Maritta Pirhonen. Coping with uncertainty in an agile systems development course. *Journal of Information Systems Education*, 29(2):117–126, 2018.
- [21] David Umphress, T. Dean Hendrix, and James H. Cross. Software process in the classroom: The capstone project experience. *IEEE Software*, 19(5):78–81, 2002.
- [22] Sarah E Wallace. Team-based learning in a capstone course in speech-language pathology: Learning outcomes and student perceptions. *Communication Disorders Quarterly*, 37(1):44–52, 2015.
- [23] Stephanie Wilson, Mathilde Bekker, Peter Johnson, and Hilary Johnson. Helping and hindering user involvement - a tale of everyday design. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, CHI '97, pages 178–185, 1997.



# Introduction to Alexa Programming\*

## Conference Tutorial

*Jay Canty, Edgar Cerna, and Wen-Jung Hsin*  
*Department of Computer Science and Information Systems*  
*Park University*  
*Parkville, MO 64152*  
*{jay.canty, edgar.cerna, wen.hsin}@park.edu*

In this one-hour tutorial, the participants will be introduced to the Alexa programming environment, and learn how to create an Alexa skill. Specifically, this tutorial is designed for the participants who have never programmed any Alexa skills before. The outline of the tutorial is as follows:

- (0) Prior to coming to the tutorial, the participants are encouraged to sign up for an account in Amazon Developer Services [1], and another account in Amazon Web Services [2] so that the tutorial is primarily designated for learning the actual content.
- (1) General introduction to Alexa programming environment.
- (2) General introduction to Alexa related devices.
- (3) Demonstration of Alexa skills.
- (4) Build an Alexa skill.
- (5) Demonstration of additional Alexa skills.
- (6) Discussion on Alexa programming resources, experience sharing, and Q&A.

## Acknowledgements

This project is made possible through the funding provided by the Park University Faculty Development Endowed Funds.

## References

- [1] Amazon Developer Services and Technologies. <https://developer.amazon.com/>. Retrieved January 12, 2020.
- [2] Amazon Web Services. <https://aws.amazon.com/>. Retrieved January 12, 2020.

---

\*Copyright is held by the author/owner.

# Real-World Data, Games and Visualizations in Early CS Courses Using BRIDGES\*

## Conference Tutorial

*Kalpathi Subramanian<sup>1</sup>, Erik Saule<sup>1</sup>, Jamie Payton<sup>2</sup>*

*<sup>1</sup>Computer Science*

*The University of North Carolina at Charlotte*

*{krs, esaule}@uncc.edu*

*<sup>2</sup>Computer and Information Sciences*

*Temple University*

*payton@temple.edu*

Despite the huge explosion in CS enrollments in the past few years, retention of CS majors remains a serious concern. Grounding Computer Science concepts in reality by solving important real-world problems or fun problems are key to increasing students' motivation and engagement in computing, which may provide a path to improving retention in CS degree programs. This workshop provides instructors with a hands-on introduction to BRIDGES (<http://bridgesuncc.github.io>), a software infrastructure for programming assignments in early computer science courses (CS1, CS2, data structures, and algorithms). BRIDGES provides capabilities for creating more engaging programming assignments, including (1) easy access to real-world data, spanning domains such as social networks, science, government, movie, music, and literature, (2) *visualizations* of the data or data structures, (3) an easy to use API for creation of games that leverage real-world data, and (4) algorithm benchmarking. Using BRIDGES in data structures, algorithms, and other courses have shown better student outcomes in follow-on courses, when compared to students from other sections of the same course. BRIDGES has impacted over 1500 students across 10 institutions since its inception 5 years ago. A repository of BRIDGES assignments (<http://bridgesuncc.github.io/newassignments.html>) is now maintained for BRIDGES users. Workshop attendees will engage in hands-on experience with BRIDGES, play with example datasets and will have the opportunity to discuss how BRIDGES can be used in their own courses.

---

\*Copyright is held by the author/owner.

# Short Modules for Introducing Heterogeneous Parallel Programming\*

## Conference Tutorial

*David P. Bunde*  
*Department of Computer Science*  
*Knox College*  
*Galesburg, IL 61401*  
*dbunde@knox.edu*

CS faculty have spent the last several years adding content on parallel computing to their curricula, following the technology since essentially all processors sold today have multiple cores. A typical setting in which to teach parallel computing is on a multicore processor with identical cores and this is currently the main configuration for desktop and laptop systems. As the technology continues to evolve, however, systems have been incorporating several kinds of heterogeneity [2]. Many phone processors include cores of different sizes, with high-performance “fat cores” and lower-performance “thin cores”, allowing them to vary their power and performance profile over time. Other processors incorporate low-power modes or special instructions for specialized computations. Meanwhile, high-end systems make heavy use of accelerators (e.g. GPGPU).

This tutorial presents three modules that introduce heterogeneity in different ways. Each can be done in only a few days of class time and fits within a standard course. Here are the modules:

1. The first module is a lecture that presents the performance benefits of systems with fat and thin cores for workloads containing varying amounts of parallelism; serial computation is more efficient on a fat core while highly-parallel computation benefits from the increased number of cores made possible by including thin cores. This conclusion is first demonstrated theoretically using Pollack’s rule, an empirical observation that the performance of a core is proportional to the square root of its area. It is then supported with benchmark results from smartphone processors that use a heterogeneous core arrangement.

---

\*Copyright is held by the author/owner.

2. The second module is aimed at a course where MIPS assembly language is taught. The module introduces ARM assembly, particularly Thumb mode, which runs with greater power efficiency but requires additional instructions. This allows for an exploration of a tradeoff between power on one hand and performance and code size on the other. It is based on the Raspberry Pi, a low-cost system aimed at hobbyists.
3. The final module again uses the Raspberry Pi, this time as an example of an embedded system with a parallel processor. The camera sensor is used to take an image and processes it using an object recognition algorithm called Local Binary Patterns. This is an embedded system variation on a recent Peachy Parallel Assignment [1].

Materials for these modules are available at <http://faculty.knox.edu/dbunde/teaching/hetero/CCSC-CP20.html>.

## Acknowledgements

This tutorial is partially supported by the National Science Foundation under grant OAC-1829554 and the Paul K. & Evalyn Elizabeth Cook Richter Memorial Fund. The modules being presented are joint work with Apan Qasem, Phillip Schielke, Arsalan Najeeb, Shebaz Chowdhury, and Annie Song. A preliminary version of this tutorial was presented at CCSC-MW 2019.

## Biography

Dr. Bunde is the William & Marilyn Ingersoll Professor of Computer Science at Knox College. He joined Knox College in Fall 2006. His research interests include parallel computing education, High-Performance Computing (HPC), and the propagation of educational innovations.

## References

- [1] O. Ozturk, B. Glick, J. Mache, and D.P. Bunde. Peachy parallel assignments (EduPar 2019). In *Proc. 9th NSF/TCPP workshop on parallel and distributed computing education (EduPar)*, 2019.
- [2] M. Zahran. Heterogeneous computing: Here to stay. *CACM*, 60(3):42–45, 2017.

# Error Detection and Correction Using Hamming Code \*

## Nifty Assignment

*Rad Alrifai*  
*Mathematics and Computer Science*  
*Northeastern State University*  
*Tahlequah, OK 74464*  
*alrifai@nsuok.edu*

Hamming code is a popular algorithm used in several applications including communication networks, DRAM, and external storage to detect up to two-bit errors and correct one-bit-errors. In this assignment, students write a program consisting of several functions needed to implement hamming code. The students work independently to write an original code in any programming language and they have two weeks to complete the assignment. The assignment is given to students enrolled in introduction to computer architecture, which has two prerequisite courses: discrete mathematics and CSII. The submitted code is required to perform the following tasks: accept a binary number as input or convert a non-binary input to a binary number, calculate the position of the parity bits for a given input, calculate the value of the parity bits, evaluate the value of the parity bits to determine if there is any error, perform error correction as needed, output the correct data bits after correcting any input errors. This assignment teaches several skills including: applying memory error detection and correction principles covered in basic computer architecture, using knowledge and skills gained from discrete mathematics, and coding in a high-level programming language.

---

\*Copyright is held by the author/owner.

# Drawing With A Turtle \*

## Nifty Assignment

*Saty Raghavachary*  
*Department of Computer Science*  
*University of Southern California*  
*Los Angeles, CA 90089*  
*{saty}@usc.edu*

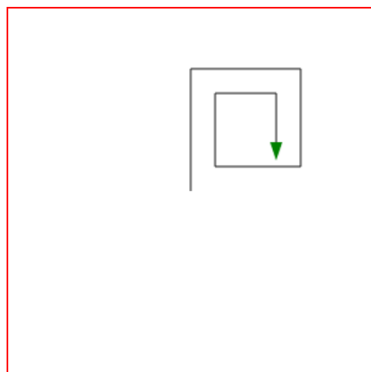
## 1 Description

The goal of the assignment is to get students to try out simple, interactive commands to plot interesting figures (with varying colors and line thicknesses). There is no software to install, nothing to compile - students can immediately start creating shapes. eg. by entering each line below one at a time in the textbox, the figure that follows is interactively and incrementally generated.

```
forward(100)
right(90)
forward(90)
right(90)
forward(80)
right(90)
forward(70)
right(90)
forward(60)
right(90)
forward(50)
right(90)
forward(40)
right(90)
```

Command (type code in the textbox below)

Canvas (graphical output of our commands)



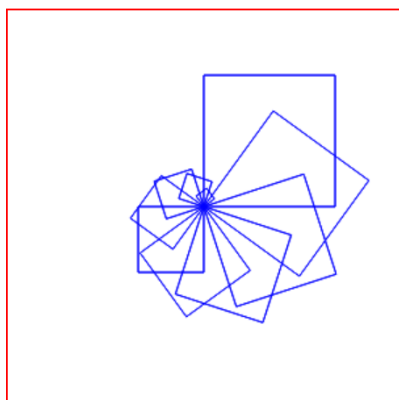
---

\*Copyright is held by the author/owner.

After using step by step constructions as the starting point, motivated students can advance to using functions to encapsulate code - this introduces them to the notion of abstraction, task composition (complex drawings can be constructed using simpler ones, which are themselves abstracted away using function names and parameters), and code reuse. An example is shown below - typing `sqSp()` into the command textbox calls 'sqSp', which in turns calls 'square':

**Command (type code in the textbox below)**

**Canvas (graphical output of our commands)**



**Definitions (type reusable code)**

```
function square(side) {  
  repeat(4, function () {  
    forward(side);  
    right(90);  
  });  
}  
  
function sqSp() {  
  hideTurtle();  
  color(0,0,255);//r,g,b, 0 to 255  
  for(s = 100; s > 0; s -= 10) {  
    square(s);  
    right(36);  
  }  
}
```

Compared to a more conventional turtle implementation based on LOGO or Python, this JavaScript-based one has a huge benefit - it can be run off a standard web page, on any platform, even on a phone or tablet. This lets students 'jump in' right away, without having to install LOGO, or Python or its turtle module (library).

The assignment appears to satisfy the various items in the 'niftiness' checklist that is on the Nifty Assignments site <http://nifty.stanford.edu/> - specifically, the assignment is:

- 'fun', visual
- doable by a variety of students
- extensible by students in multiple ways
- easily adoptable by instructors
- modifiable by instructors

## 2 Metadata

Summary	The assignment asks students to draw colorful figures, using a screen-based turtle that responds to simple typed-in commands - no installation required, easy to get started right away.
Topics	2D graphics, simple 'turtle based' programming, computational thinking.
Audience	CS1/CS2 students. CS1 students would use sequences of turtle commands to plot figures; CS2 students would learn the concept of functions (including nested function calls and recursion), and use this to construct more complex figures including fractals, spirolaterals, etc.
Difficulty	Easy (CS1)/intermediate(CS2). The difficulty level for CS2 would vary, depending on in-depth they get into writing and calling functions.
Strengths	The assignment would provide students, exposure to programming, in an interactive, visual manner; using relatively simple commands (such as forward(10)) they can create extremely complex and pretty patterns, which would let them experience the power, and fun, of coding. After mastering the basics, they can transition to more advanced programming (eg. using functions, including recursive ones).
Weaknesses	No significant weakness; students do need to learn basic turtle syntax, and later, the syntax for creating functions.
Dependencies	None, beyond what the assignment comes with [HTML, CSS and JavaScript code for turtle handling are all part of the assignment] - this makes it very easy to get started.
Variants	After doing the assignment, interested students can use this as a starting point to do much more - create more complex figures using recursive functions, explore JavaScript as a language for graphics (including learning canvas and WebGL-based graphics programming), try their turtle drawings in Python (using the 'turtle' module), try their commands in NetLogo [ <a href="https://ccl.northwestern.edu/netlogo/">https://ccl.northwestern.edu/netlogo/</a> ] which offers a rich simulation environment, etc.



# TwHeatmap: Visualizing Sentiment Analysis of Tweets\*

## Nifty Assignment

*Evelyn Brannock and Robert Lutz*  
*Georgia Gwinnett College*  
*Lawrenceville, GA 30043*  
*{ebrannoc, rlutz}@ggc.edu*

## 1 About

Sentiment Analysis is a popular application of Natural Language Processing (NLP). This exercise offers the capability to perform opinion mining in the political arena by feeding data into a cloud natural language processor, without in-depth proficiency in machine learning (ML) algorithms. It is an engaging mechanism for interesting students in using ML to extract information from voluminous amounts of text found on Twitter to understand the structure and meaning of text.

## 2 Materials

- Educational codes for access to Google Cloud Platform (GCP)
- Credentials to access APIs
- Jupyter Notebook

---

\*Copyright is held by the author/owner.



## 4 Metadata

Summary	<p>Students are asked to provide an app that provides results of a sentiment analysis of tweets on some current “hot” political subject, such as the impeachment report or tweets from President Trump (shown).</p> <ul style="list-style-type: none"><li>• Load required libraries</li><li>• Provide credentials to access APIs</li><li>• Establish calling endpoint, call parameters and make the request</li><li>• Coerce the response into a list of messages.</li><li>• Create a (reusable) function for sentiment analysis using Google’s Natural Language Processing</li><li>• Produce report or visualization</li></ul>
Topics	ReSTful Programming, Machine Learning, Natural Language Processing, Sentiment Analysis, Data Visualization
Audience	Late CS1, early CS2, Web Development, Data Analytics / Data Visualization, Artificial Intelligence
Difficulty	Medium, not because of code required, but because the student must understand the workflow required to ensure end-to-end success of the app
Strengths	<ul style="list-style-type: none"><li>• <b>Global in nature:</b> Supports a variety of languages.</li><li>• <b>Introduces the value of integration to other applications and tools:</b> Encourages learning another API</li><li>• <b>Deep algorithmic and coding knowledge is not required:</b> Students are not required to understand complex AI concepts, statistics, or algorithms</li><li>• <b>Adaptability to multiple audiences:</b> Starter code can be provided to scaffold content areas of other courses</li></ul>
Weaknesses/ Issues	<ul style="list-style-type: none"><li>• Must obtain credits/credentials to access Twitter and GCP APIs</li><li>• Integration issues between the various interacting technologies</li><li>• API Overload – WordCloud, BeautifulSoup, GCP, twitter</li><li>• Cloud service throttling</li></ul>
Dependencies	<ul style="list-style-type: none"><li>• Google Cloud Platform</li><li>• Twitter and Twitter Developer Access</li></ul>
Variants	Any subject can be analyzed, multiple programming languages can be introduced, a user interface which includes date range, gauges, correlation to other data, etc., find sentiment of replies, track trends over time, many visualizations can be explored