The SoTL Grant Proposal:

Integrating Low Stake Formative and Summative Assessments into Teaching of Introduction to Computer Science for Continuous Improvement and Effectiveness Measurement

Mohsen Dorodchi, Ph.D.

Teaching Associate Professor

Department of Computer Science

University of North Caroline, Charlotte

Abstract:

High failure rate in the introductory programming courses of computer science (CS1) is a well-known problem which is also believed to be the major reason in declining retention rates [15, 14]. On the other hand, it is reported that some students change their major from computer science in spite of good grades they earn in the first two programming courses [16]. As a result, the Computer Science Education (CSE) community has identified the need to include various instructional teaching strategies and tools as part of course design to address students' high failure rates in programming courses. In this paper, we propose a new model for teaching introductory programming course for large lecture and closed labs configuration using active learning techniques. Our primary studies confirm that a coherent lecture-lab model is very effective in first programming course in computer science (CS1). To achieve the desired coherency, we have developed our activities based on established theories such as Kolb's learning model and Bloom's Taxonomy (BT). The presentation of this work starts from analyzing few essential problems in students' learning of first programming course. Furthermore, we review our developed interventions or methods addressing the aforementioned problems. And finally, we propose our method which combines planning, evaluation and action together using Logic Model [11].

Budget Request for SOTL Grant

Year ___2017___

Joint Proposal? _____ Yes __X__ No

Title of Project: **Integrating Low Stake Formative and Summative Assessments into Teaching of Introduction to Computer Science for**

Duration of Project: 2017-18

Primary Investigator(s): Mohsen Dorodchi

Email Address(es): Mohsen.Dorodchi@uncc.edu

UNC Charlotte SOTL Grants Previously Received (please names of project, PIs, and dates): ________________

Allocate operating budget to Department of: Computer Science

		Year One
Account #	Award	January 2017 to June 2017
Faculty Stipend	Transferred directly from Academic Affairs to Grantee on May 17	$3850
911250	Graduate Student Salaries	$3150
911300	Special Pay (Faculty on UNCC payroll other than Grantee)	
915000	Student Temporary Wages	
915900	Non-student Temporary Wages	
920000	Honorarium (Individual(s) not with UNCC)	
921150	Participant Stipends	
925000	Travel - Domestic	
926000	Travel - Foreign	
928000	Communication and/or Printing	
930000	Supplies	
942000	Computing Equipment	
944000	Educational Equipment	
951000	Other Current Services	
	GRAND TOTAL	$7000

		Year Two
Account #	Award	July 2017 to June 2018
Faculty Stipend	Transferred directly from Academic Affairs to Grantee on May 18	$3850
911250	Graduate Student Salaries	$3150
911300	Special Pay (Faculty on UNCC payroll other than Grantee)	
915000	Student Temporary Wages	
915900	Non-student Temporary Wages	
920000	Honorarium (Individual(s) not with UNCC)	
921150	Participant Stipends	
925000	Travel - Domestic	
926000	Travel - Foreign	
928000	Communication and/or Printing	
930000	Supplies	
942000	Computing Equipment	
944000	Educational Equipment	
951000	Other Current Services	
	TOTAL	$7000
	GRAND TOTAL	$14000

1. Narrative that explains how the funds requested will be used is provided here.

The project starts in spring 2017, and during that semester the faculty investigator will execute a pilot set of low stake assessments with the proposed active learning activities. The data will be collected in spring (and in the summer). Later in the summer, the labs and surveys will be modified for fall and spring of 2017-18 based on the analysis of the data and study of students and teaching assistant surveys. In case of teaching the course in summer of 2017, more data will be collected. The major research by the faculty will be done during the academic year of 2017-18 when extensive data will be collected from different class sessions. It is intended to present the result of this work at one of the prestigious computer science education symposiums or conferences.

The graduate student who will be working on this project is Nasrin Dehbozorgi who is a new Ph.D. student in the college of computing and informatics working with Mary Lou Maher and Mohsen Dorodchi on Design Patterns for Active Learning. She has been working on design-pattern based introductory computer programming labs. Her CV will be available upon request.

The goal is to publish the result of this work in one of Computer Science Education Symposiums or Conferences such as IEEE Frontiers in Education, Frontiers in Computer Science Education, and/or ACM Symposium on Computer Science Education (SIGCSE).

The faculty stipend will be summers of 2017 and 2018 for working on the overall project findings and preparing the materials for applying to larger grants with collaboration with other faculty and/or institutions. The graduate student stipend is mainly to pilot the surveys and actively participate in data collection and analysis. It is worthwhile to reiterate that NSF particularly is interested in educational research with support of real data.

2. Has funding for the project been requested from other sources? ___ Yes __X__ No. If yes, list sources.

UNC CHARLOTTE
College of Computing and Informatics
Department of Computer Science
9201 University City Blvd, Charlotte, NC 28223-0001
t/ 704.687.8560 f/ 704.687.1651 cs.uncc.edu

November 1, 2016

To Whom It May Concern:

I would like to express my support for the Scholarship of Teaching and Learning (SoTL) grant proposal prepared by Dr. Mohsen Dorodchi, entitled 'Integrating Low Stake Formative and Summative Assessments into Teaching of Introduction to Computer Science for Continuous Improvement and Effectiveness Measurement'. I have read his proposal and endorse it with enthusiasm.

Dr. Dorodchi has been continuously working on developing active learning activities for our introductory class, ITSC 1212. He is proposing continual evaluation of learning activities to be able to improve the effectiveness of active learning delivery of content. The introductory programming course at the College of Computing and Informatics has been a challenge for students. Conceptualizing computer science concepts while learning the syntax and semantics of programming languages does not seem to be the right fit for class objectives. Therefore, design and development of active learning based on design patterns developed by different faculty members requires continuous assessment and adjustment for different types of students. Dr. Dorodchi designed and developed new active learning patterns in Summer of 2016 for recall, factual, and peer learning in ITCS 1212. Therefore, the proposed assessment activities are in alignment with the goals of SoTL program which includes the disciplinary study of teaching and student learning.

Proposed project will impact students beyond introductory programming classes. Dr. Dorodchi will be investigating the effectiveness of the developed activity-based learning model. If successful, the model can be adopted in higher level computer science courses and other STEM disciplines. In other words, this instructional research project will have the potential to increase curricular integration and, benefit the entire UNC Charlotte teaching and learning community. I believe the project will contribute to our understanding of conditions that impact the exchange and improvement of pedagogy.

Last but not least, Dr. Dorodchi is a great candidate for SoTL award. He is ambitious, collegial and enthusiastic. He is committed to improving student learning, retention and outcomes in introductory Computer Science courses.

Sincerely,

B. Cukic

Bojan Cukic, Ph.D.
Professor and Chair

Project Narrative:

A. Specific Aims

Project Goal: In this work, we propose to develop and integrate a series of low-stake assessments into existing group activities to be used during lecture and lab sections of the introductory programming course (ITSC1212) based on a re-orientation of undergraduate computing education. The focus is on evaluation of student learning which is based on team-based active learning and peer-instruction and study the effectiveness of the interventions targeting students to understand the fundamental concepts of computer programming.

1. Project Objectives: This project has three major objectives as listed below.
    a. Creating a series of low-stake assessment techniques in evaluating the active learning activities.
    b. Executing the developed assessments to collect data for formative as well as summative evaluation of the active learning activities.
    c. Study the correlation between the effect of lab and lecture activities on student learning of fundamentals of programming.

Statement of the specific research questions: We are going to collect data from students including grades to test the following research questions and hypotheses. For our baseline study, our control group will be from Spring and Fall of 2016. In addition, Spring 2017and Fall 2017 will be our test group. The baseline study will use the grades as the major indicator. In another study, students are compared to themselves as far as the effect of interventions with the pre/post tests. Here are the research questions:

- Baseline Study Research Questions:
    - Would the integration of low-stake assessments (and corresponding adjustments to the course) enhance students' grades?
        - Test 1: No effect observed on students' grades comparing the test group and the control group for transfer and internal students (Spring Semester Students).
            - H_0: average student grades of (Sp 17) $\leq$ average student grades of (Sp16)
            - H_1: average student grades of (Sp 17) $>$ average student grades of (Sp16)

- Test 2: No effect observed on students' grades comparing the test group and the control group for new freshman students (Fall Semester Students).
  - H_0: average student grades of (Fall 17) ≤ average student grades of (Fall16)
  - H_1: average student grades of (Fall 17) > average student grades of (Fall16)

- Pre-Post Tests
  - Would infusion and integration of low-stake assessments enhance students' motivation?
    - Test 3: No effect observed on students' motivation comparing before and after the interventions.
      - H_0: students' attendance after interventions (post) ≤ students' attendance before (pre)
      - H_1: students' attendance after interventions (post) > students' attendance before (pre)
  - Would infusion and integration of low-stake assessments enhance students' self-satisfaction?
    - Test 4: No effect observed on students' self-assessment comparing before and after the interventions.
      - H_0: students' self-assessment score after interventions (post) ≤ students' self-assessment before (pre)
      - H_1: students' self-assessment after interventions (post) > students' self-assessment before (pre)
  - Would infusion and integration of low-stake assessments enhance students' grades?
      - H_0: students' grades show no improvement throughout the semester.
      - H_1: students' grades show no improvement throughout the semester.

2. <u>Rationale for the project:</u> There are two major problems that we try to address, 1) enhancing the learning of programming by continuing on integrating active learning activities such as preparation before the lectures and labs and working on low-stake teams for peer-learning; and 2) enhancing the quality of students learning by infusing and integrating continuous low-stake assessments into the course activities.
3. <u>The impact of the study on undergraduate teaching and learning:</u> We strongly believe that introducing undergraduate students to working in groups and actively learning in the

classroom impacts the learning of students significantly. Moreover, based on our prior experiments we need to adjust the activities and/or their orders based on students' learning models.

B. Literature Review

B.1. Review of the students' learning issues in computer programming

It is reported and our observations also prove the fact that more than one single practice time is needed to learn programming topics [3]. One solution to this problem is to have students prepared before the class activities. In other words, the student's productivity during the class activities would be much higher with preparation due to the following reasons [1].

1) A prepared student could interact with the instructor, teaching assistant, and peers more effectively.

2) A prepared student applies the knowledge of the pre-lab to lab activities and therefore, learns the materials better.

Our experiments as well as our findings from literature surveys confirm the effectiveness of the students' preparations before the class as one of the most effective solutions to enhance the quality of their learning. When students attend class sessions without prior preparation, the core time of activity will be dedicated to preparing them to understand the details of activities and/or set up the basic parts of the experiments. Therefore, students are not able to efficiently accomplish the goals of the learning activity.

Another common issue, which is proved from study of hands-on activities is that some of the concepts cannot be constructed in a simple process and require more iterations than the planned practice sessions [12].

We have also observed similar problems in our labs and active learning lecture classes. To address this issue, we first started by searching among different students' learning patterns, where we found Kolb's learning model. His experiential learning theory (ELT), determines four different stages of knowledge construction that a learner has to go through: 1) stimulation, 2) experimentation, 3) reflection, and 4) abstraction [12].

B.2. Review of Logic Model for Program Planning and Evaluation

The logic model combines three strategies or steps as Planning, Implementing and Evaluating. In the next sections, we will explain the details of the model as we are applying it to our project [11].

B.3. Enhancing Problem Solving Skills

Problem solving in the broad sense means solving different problems beyond traditional well-formulated mathematical problems. In teaching problem solving skill, it seems that the focus is to educate learners to be able to solve different types of problems by following logical reasoning. Traditional models of learning with their hierarchical structures and instructional design indicated that problem solving is a combination of several building blocks, like rules, principles and concepts that the learners use to solve a problem [2].

Polya in [6] classifies “problem solving” technique into four general categories:

- understanding the nature of the problem
- arranging a plan to solve the problem
- test the plan
- observe the outcome of the plan

In [4] it states that the aforementioned steps are broadly adopted in learning programming concepts. Although these principles are very useful, however many students might not have enough confidence in applying them for solving a problem. Woods et al [7] also mentions that problem solving means relating to real world problem. It reports that most of the students that learn problem solving in university don’t have essential skills to solve real world problem.

Gagne’s in 1985 define problem solving as a higher order thinking skill which synthesizes other concepts and rules into a higher order of rules that can be applied in a specific situation [5]. Although it has not been clearly identified, however it can be inferred that problem solving requires both analysis and synthesis skills and includes using and finding principles and rules to solve problems.

From our perspective, the computer programmers should be able to apply the taught problem solving methods of computer science curriculum to real world problems. Therefore, problem solving means practical (real world) problem solving through synthesizing and analyzing.

What kind of skills should problem solvers have? Jonassen et al [8] mention that problem solving includes two main components, which are: Attitude and cognitive aspects. He says that for solving problems, the learners should first want to solve it, and then they should believe that they can do it. In this regards he believes that important attitudinal factors that should be taken into consideration in the process of problem solving are knowledge about self, confidence, anxiety, effort, and persistence. In the same direction, Woods et al [7] has proposed a more detailed list of problem solver's characteristics categorized also into affective as well as cognitive domain of bloom's taxonomy which is as listed below.

Problem Solver's Characteristics - Affective Domain:

I. Takes time to study the problem by reading, collecting information and defining it.
II. Approaches the problem systematically and organized.
III. Focuses on accurate solution rather than solving the problem quickly
IV. Is willing challenge with ambiguity and taking risk, accepts change and is able to manage the stress.
V. Demonstrates flexible approach to problem solving from different perspectives.

Problem Solver's Characteristics - Cognitive Domain:

I. Is able to use a process, and various heuristics and tactics to tackle problems
II. Observes the process of their problem solving and ponders upon its effectiveness.
III. Applies the relevant subject knowledge and critically and objectively evaluates the accuracy, quality and relevance of that knowledge.
IV. Takes notes of the ideas and draws charts/figures, when solving a problem.

This observation recommends that the student should first practice on developing certain attitudes before they are able to acquire necessary skills to solve open-ended problems.

Pattern is a solution to a problem that reoccurs in a specific context [9]. Alexander in [10] explains that a pattern "describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice".

In this project, we propose a model of teaching fundamentals of programming based on the ELT and Logic Model as explained in the next section.

C. Methods:

Using Logic Model, we start from planning stage.

Stage I: Clarifying program theory (Planning):

As shown in figure 1, the planning includes the following steps.

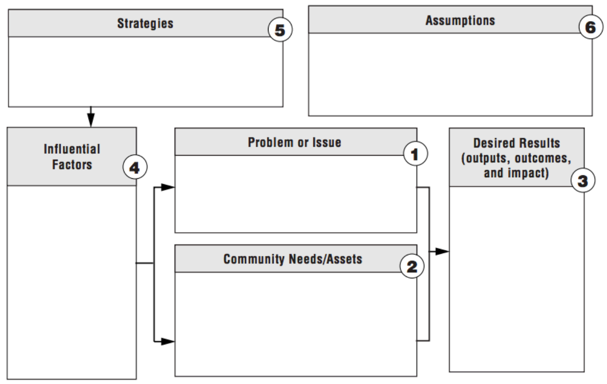

Figure 1. Logic Model Steps in Planning Stage [11]

1. Problem/issue statement: help students learn a wide variety of skills such as:
    - Course content,
    - Critical and analytical thinking including interpretation of data and error analysis
    - Problem solving
    - Self-motivation and self-regulation
    - Ability to work in teams
    - Ability to make connection between class and lab, and lab and real world.
    - Ability to communicate course concepts
    - Curiosity
    - Creativity

2. Community needs/assets:
    - UNCC is promoting active learning as well as computer science education community.
    - Effectiveness study as well as continuous improvement is a need.

3. Desired results (outputs, outcomes and impacts):
    - Academically successful students (The rate of D's, F's, and W's are reduced)
    - Professionally sound students (The student assessment of active and team-based learning confirms the effectiveness of the method)

4. Influential factors:

- Increase student’s interest in science
- Decrease the drop, or failure rates

5. Strategies:
    - Setting clear expectation for students for the course and integrate them into course assignment
    - Outline and model successful ways for students to achieve course goals
    - Provide stepwise milestones and feedback for students to foster mastery experiences through the course
    - Promote student monitoring of progress by requiring reflections on class and lab activities.
    - Motivating the students by connecting the course to student’s interests and goals via use of context-rich problems, real life case studies, or problem based learning
    - Use of team-based learning a structured class approach that involves peer support and individual accountability, to foster student’s self-regulation.

6. Assumptions:
    - Spring semester students in Introduction to Computer Science Course (ITSC1212) are generally different from fall semester students which are normally coming as new freshman group.

Stage II. Demonstrating program’s progress (Implementation):

1. Resources:
    - Closed-lab (or dedicated lab) time and activities
    - Active Learning classroom

2. Activities:
    - Active Learning class activities, prelab and post labs for the lab

3. Outputs:

- Academically successful students (The rate of D's, F's, and W's are reduced)
- Professionally sound students (The student assessment of active and team-based learning confirms the effectiveness of the method)

4. Short & long-term outcomes (SMART: Specific, Measurable, Action-oriented, Realistic, and Timed)
   - Short-term: Enhancement of student learning (as indicated by grades) and motivation level to learn (as indicated by self-assessments)
   - Long-term: T-shaped Professional Graduates able to find jobs within 6 months of graduation

5. Impact:
   - An adjustable active learning model which improves students' learning abilities as well as self-confidence

Resources	Activities	Outputs	Short- & Long-Term Outcomes	Impact
In order to accomplish our set of activities we will need the following:	*In order to address our problem or asset we will conduct the following activities:*	*We expect that once completed or under way these activities will produce the following evidence of service delivery:*	*We expect that if completed or ongoing these activities will lead to the following changes in 1–3 then 4–6 years:*	*We expect that if completed these activities will lead to the following changes in 7–10 years:*

Table 1. Implementation Stage Details [11]

Table 1 can be used during the second stage.

Stage III: Program evaluation questions and indicators (Evaluation):

The Importance of "Prove" and "Improve" Questions

There are two different types of evaluation questions – formative help you to improve your program and summative help you prove whether your program worked the way you planned.

Both kinds of evaluation questions generate information that determines the extent to which your program has had the success you expected and provide groundwork for sharing with others the successes and lessons learned from your program.

Benefits of Formative and Summative Evaluation Questions

Formative Evaluation – Improve

- Provides information that helps you improve your program. Generates periodic reports. Information can be shared quickly.
- Focuses most on program activities, outputs, and short-term outcomes for the purpose of monitoring progress and making mid-course corrections when needed.
- Helpful in bringing suggestions for improvement to the attention of staff.

Summative Evaluation – Prove

- Generates information that can be used to demonstrate the results of your program to funders and your community.
- Focuses most on program's intermediate term outcomes and impact. Although data may be collected throughout the program, the purpose is to determine the value and worth of a program based on results.
- Helpful in describing the quality and effectiveness of your program by documenting its impact on participants and the community.

Evaluation Question Development Process:

1. **Focus area:**
   Our focus in on learning the course content, increasing students' self-motivation and improving their problem solving and teamwork skills.

2. **Audiences:**
   We identified the audiences as teacher and student.

3. **Questions:**
   Here are the examples of the questions that we ask in the low-stake assessments.

Students:

- How do I know if I have learnt a concept?
- How do I identify where I have problems?

Teachers/TA:

- How can I get regular report/feedback from students' performance
- How do we know if the method is improving the student's motivation?
- How do we know if the method is improving the student's learning?

4. **Information use**:
    - The hypothesis tests mentioned earlier in this proposal is going to be based on the collected data.

5. **Indicators:**
    - Many students don't know what they don't know. To improve the learning process, we intend to help students focus on the thought process (i.e. the ability to be meta-cognitive which includes thinking whether they know or understands something)
    - This can be done by asking students questions about their thought process in problem solving by class-wide collaborative interactions and thinking aloud and challenging each other (i.e. having a 'journal club' every other week, it can be an online discussion or at the end of class). This will be a guided discussion by TA/ Instructor raising a topic for students to think and talk about. A well-studied form will be given to students to be filled out at the end of discussion to collect their reflective ideas.
    - Students are also required to fill out reflection forms individually at the end of each class.
    - The data of both reflections (collaborative and individual) are collected and a reflection analysis for mid-course changes/corrections will be applied.

- Lab-test and quizzes and tests (grades in general) are another factor to determine the improvement of students.
- We have a pre-test and post-test at the beginning and end of each semester to evaluate student's knowledge and level of interest/motivation in the field. (I should search for methods to measure the level of interest or motivation, have no clear idea currently)

6. **Technical assistance:**
    - CCI has the learning analytics initiatives and well as other centers in the university that might be interested in contributing to the data analysis of this project.

D. Evaluation

Project evaluation will include collection of data in the use of design patterns in introductory programming courses and to assess the degree to which these tools impact students' grasp of performance. As mentioned before, we try to find out whether preparation (and follow up) activities for the lab and lecture would have any affect on performance of the students in lab tests, lecture tests, and overall grade.
It includes a formative assessment by collecting input and feedback from students enrolled in the ITSC 1212, via qualitative and quantitative means. Student assessment will include a variety of indicators (such as grades, students' surveys, TA surveys) as direct causal links are not possible in quasi-experimental educational interventions. Data collected will include the test grades in lab and lecture as well as the overall all final grade of students. Pre and posttest surveys will be developed to collect additional data.

Summative assessment will include student learning outcomes, as identified in course grades and test outcomes, results from passing grades and retention changes, student survey input, course evaluations, and interview themes. Overall assessment of the project will be determined by evaluating activities and tracking participants throughout the grant lifecycle, presented in summative reports, as shown in the following table.

E. Dissemination: The results of this work will be presented to UNCC community to increase the level of awareness related to open source and its benefits. It is also the intention of the investigator to publish the result of the work in computer science education symposiums and conferences.

F. Human Subject: The students are the human subject in this study and the IRB process will be followed before asking them to participate in the study.

G. Extramural Funding: N/A.

H. TimeLine: We execute the project in the following steps.

	Activity for Spring 2017	**Tentative Date**
1	Preparing the student surveys and IRB approval	Spring 2017
2	Integrating the surveys into the Canvas Assignments and Activities as a part of the course	Summer 2017
3	Execution of the method	Fall 2017, Spring 2018
4	Data Collection	Spring 2017, Fall 2017, Spring 2018
5	Data Analysis	Spring/Summer 2018

References:

[1] Ahmadian, M. 2002. A Prelab-Lab-Postlab Digital Electronics Course as a Prerequisite for the CS Computer Architecture Course. Journal of Computing Sciences in Colleges, ACM, 17(5), 198-203.

[2] Jonassen, David H. Instructional Design Models for Well-Structured and Ill-Structured Problem-Solving Learning Outcomes.

[3] Balid, W., et. al. 2015. The Impact of Different Pre-Lab Preparation Modes on Embedded Systems Hands-on Lab. The 9th Annual American Society for Engineering Education (ASEE) Global Colloquium on Engineering Education.

[4] Lai, Yiu-chi,Wong, Tak-wah. Infusing problem solving skills into computer lessons, ACM SIGCSE Bulletin, 2007.

[5] Kirkley, J. Principles for Teaching Problem Solving: Technical Paper #4.

[6] Polya, G. (1971). How to solve it: A new aspect of mathematical method (2nd ed.). Princeton: Princeton University Press.

[7] Woods D. R., Hrymak, A. N., Marshall, R. R., Wood, P. E., Crowe, C. M., Hoffman, T. W., Wright, J. D., Taylor, P. A., Woodhouse, K. A., Bouchard, C. G. K., Developing problem solving skills: The McMaster problem solving program. ASEE J of Engng Educ., 86, 2, 75-91 (1997).

[8] Jonassen, D., & Tessmer, M. (1996). An outcomes-based taxonomy for the design, evaluation, and research on instructional systems. Training Research Journal

[9] Goodyear, Peter. Patterns , pattern languages and educational design, 2004.

[10] McAndrew, P., Goodyear, P., Dalziel, J., Patterns, designs and activities: unifying descriptions of learning structures, 2006.

[11] W. K. Kellogg Foundation. Logic Model Development Guide, W.K. Kellogg Foundation. www.kkf.org, 2004.

[12] Kolb, D. A. 1984. Experiential learning: experience as the source of learning and development, Prentice-Hall.

[13] Kolb, D., Kolb, A. 2005. The Kolb Learning Style Inventory—Version 3.1 2005 Technical Specifications. http://learningfromexperience.com/media/2010/08/Tech_spec_LSI.pdf

[14] Watson C., Li F. W. B. 2014. Failure Rates in Introductory Programming Revisited, ACM ITiSCE 2014.

[15] Bennedsen, J., Caspersen, E., M. 2007. Failure rates in introductory programming. ACM SIGCSE Bulletin, Volume 39 (2), 32-36.

[16] Biggers, M., et. al. 2008. Student Perceptions of Computer Science: A Retention Study Comparing Graduating Seniors with CS Leavers, ACM SIGCSE Bulletin, 402-406.