Exploring Hypotheses about Computational Thinking with Data Science

AUSTIN CORY BART, Virginia Tech

Lorem ipsum dolor sit amet consectetur adipiscing elit, ante sodales morbi torquent vehicula lobortis convallis erat, fusce viverra diam condimentum vivamus cras. Eget euismod purus aptent netus porta dictum risus dui ullamcorper magnis, arcu integer per natoque litora vulputate hendrerit dictumst rhoncus. Fringilla neque penatibus vivamus ridiculus nec blandit congue lobortis, mattis eros tellus tortor facilisi inceptos aenean, etiam feugiat velit dictumst est ad sociis. Mollis natoque montes mauris etiam fames suscipit condimentum augue luctus ut, orci torquent fermentum euismod curae ridiculus ornare massa aptent. Aliquet maecenas himenaeos consequat est posuere parturient eu et commodo facilisis phasellus, ac sollicitudin quam fusce felis tellus turpis risus aenean. Blandit penatibus turpis ridiculus platea libero sociis risus vitae inceptos dui, pulvinar cras class suscipit dignissim phasellus et quisque nostra convallis, non est dapibus laoreet ante vivamus parturient a mollis.

CCS Concepts: • **Social and professional topics** → **Computational thinking**; *Student assessment*; • **Applied computing** → Interactive learning environments;

Additional Key Words and Phrases: Computational Thinking, Computing Education, Data Science

ACM Reference format:
Austin Cory Bart. 2017. Exploring Hypotheses about Computational Thinking with Data Science. *ACM Trans. Comput. Educ.* 1, 1, Article 1 (May 2017), 27 pages.
https://doi.org/0000001.0000001

1 DESIGNING A COMPUTATIONAL THINKING COURSE WITH CORGIS

The initial versions of the RealTimeWeb project were used in traditional introductory computing courses, and some initial data was collected there. However, the vast majority of the research data used in this dissertation was collected in a Computational Thinking course designed using CORGIS from the ground up. I was a primary developer of the course material, along with Dr. Dennis Kafura and Bushra Chowdhury. Although there are a number of existing Computational Thinking curricula, I defined a particular set of educational objectives and course topics that I believed were valuable for our students. Large portions of our course were created from scratch to achieve our pedagogical goals. In this section, I describe the courses' audience, content, context, structure, technology, and assessments.

1.1 Audience

The Computational Thinking curriculum was designed to fulfill a new general education requirement at Virginia Tech, to provide "Computational Thinking" to all undergraduates. Therefore, students come from the arts, social sciences, humanities, agriculture, and more. Although some

1946-6226/2017/5-ART1 $15.00
https://doi.org/0000001.0000001

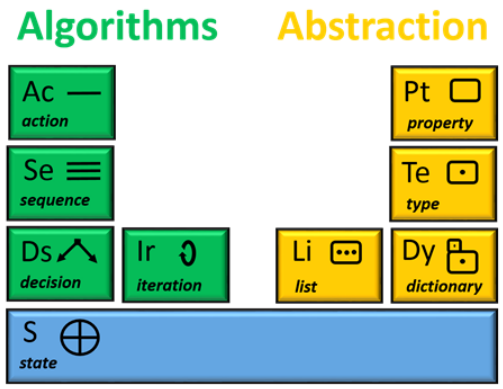


Fig. 1. Fundamental Elements of Algorithms and Abstraction

science and engineering majors enroll, it is expected that most students will not be STEM majors. The students are not expected to have prior programming experience, and the course itself has no prerequisites. Students could be at any level, from Freshman to Senior.

1.2 Course Content

The three major topics in the course are:

- **Abstraction:** The idea that real-world entities (e.g, people, objects, events, actions, processes, etc.) can be simplified and concretely represented to suit the needs of some stakeholder.
- **Algorithms:** The idea that computational abstractions can be manipulated using formal language.
- **Social Impacts:** The idea that computing provides new powers and knowledge that can influence society.

Our core learning objective is for students to be able to create an algorithm that manipulates data to answer real-world questions. Students must express their algorithms using a practical programming language – in this case, Python. Further, they must be able to navigate and describe data in contexts related to their own careers and across other fields. Finally, they should be able to discuss the social impacts, relevant stakeholders, and ethics of the questions and answers they find.

Figure 1 illustrates the hierarchical and connected nature of two of the courses' topics, Algorithms and Abstractions. Algorithms, fundamentally, describe *actions* in a specific *sequence*. Depending on the formality of the algorithmic agent, these actions can vary in complexity and atomicity. Higher-order constructs empower algorithms with the ability to branch (make *decisions*) and repeat (*iterate*). Abstractions are concretely represented as variables or *properties*, which formally have *types* and quantitative characteristics. We refer to variables as properties to avoid confusing students with the algebraic term. Higher-order organizational structures such as homogeneous *lists* and heterogeneous *dictionaries* allow for construction of complex data. Connecting these two ideas is *state*, the collection of properties whose structure is defined by Abstraction and whose values are used or changed dynamically during the performance of an Algorithm. Taken together, these concepts describe our ideal "notational machine" for students, unifying their understanding of control structures with the data structures being manipulated.

The topics of algorithms, abstraction, and social impacts were chosen for a combination of philosophical and pragmatic reasons. Philosophically, the algorithmic constructs in Figure 1 are embodied in the Turing machine model of computation and are elements of imperative programming languages. In addition to giving robust meaning to "computation", the algorithmic elements are sufficient for our learners to develop cognitive skills in process-oriented thinking, which we view as one component of the "thinking" in computational thinking. Pragmatically, time constraints prevent us from including more advanced programming concepts (e.g., objects, functions). Philosophically, the abstraction elements in Figure 1 carry through on our commitment to presenting computation as being a tool for answering questions about the real world. These abstraction elements are sufficient to define instances of complex entities using structured collections with information-bearing properties. In addition to helping students see the world through the lens of information, the abstraction elements stimulate cognitive skills in the management of complexity through "layers of abstraction", which we view as the second component of the "thinking" in computational thinking. Pragmatically, time constraints limit the number of structures that we can introduce, and the ones we did include were influenced by our choice of Python as the programming language. Philosophically, the social impacts topic encourages our learners to think about the ways in which computation shapes decisions made in the real world and influences the character of that world. This course component is in line with calls for computer science education to include the social and ethical dimensions of computing. Pragmatically, we were influenced by the requirement of our university's general education guidelines to include a components on ethics.

In the original version of the course, we also taught a unit on creating functions to encapsulate reusable chunks of code. However, time restrictions forced us to remove those lessons. Instead, modules and functions are introduced as mechanisms for reusing existing code written by other developers, without any class time devoted to creating functions. In many cases, we attempt to provide this material on functions in individual lessons with students.

1.3 Course Context

Orthogonal to the course content, the course context is centered around Data Science, supported directly by the CORGIS project. In the course, students must extract meaning from data using computational analyses. Our argument to students is that most disciplines benefit from Data Science, since the world is becoming increasingly data-driven. Students use datasets provided through the CORGIS project in order to create visualizations (e.g,. histograms, line plots) and perform simple statistical calculations (e.g., average, sums, counts). These kinds of analyses, while not statistically rigorous or computationally difficult, provide a range of appropriate introductory-level problems and projects. Students interpret these visualizations in a social context, further supporting the core learning objectives. The course context provides the hook for deploying and evaluating the CORGIS project.

1.4 Course Structure

As shown in Figure 2, there are several distinct phases to the course: introduction, hands-on conceptual activities, BlockPy, Mini-project, and Final Project.

Module 1 is a brief overview of the course, presenting the major topics and the cohort system (which uses peer learning). After their introduction to the course, students originally used NetLogo[1] and then the Blockly Maze game[2]. In later iterations of the course (starting with the Fall 2016 semester), NetLogo was replaced with the CORGIS Visualizer, and the Maze activity was moved

[1] https://ccl.northwestern.edu/netlogo/
[2] https://blockly-games.appspot.com/maze

Fig. 2. Overview of Course

Module	Topics	Time Period
1	Course Overview	Days 1-4
2	Computational Modeling and Abstraction	Days 5-9
3	Algorithms and Programming	Days 10-14
4	Python and Big Data	Days 15-19
5	Mini-project	Days 20-22
6	Final project	Days 23-29

to Module 2. These applications introduce the overarching ideas of Abstraction and Algorithms, respectively.

In Module 2, students begin learning fundamentals of computational abstraction and basic control structures, working mostly with paper-and-pencil and kinesthetic activities. In one lesson, groups of 5 (referred to here as a "cohort") work together to write instructions for a "card-sorting robot", executed by the course staff on playing cards in order to demonstrate the basics of algorithm design and the ambiguities of natural language. The course staff also performs a "play" that models the state of the program, the program counter, and console for a simple algorithm to process a list of data. During execution of the program, the instructor regularly stops the lesson and has students predict the next state. There are also a series of smaller pencil-and-paper activities for topics relating to decision, iteration, and abstraction.

During Module 3, students must complete a series of programming assignments in BlockPy, a block-based programming environment for Python [1]. These assignments incorporate real-world datasets from the CORGIS project, and students create charts and compute simple statistical information (averages, counts, thresholds, etc.). By the end of Module 3, students are working in the "text mode" of BlockPy, to prepare them for the transition to Module 4. This module builds on their Python experience by having them complete similar programming assignments, except this time in a professional desktop IDE named Spyder[3]. They continue to use Spyder in Modules 5 and 6, where they complete a mini-project and then their final project. During the mini-project, students work in their cohort to answer questions about the CORGIS State Crime dataset. As a group, they create visualizations and a presentation, which they are individually responsible for presenting as a video. This is largely practice for their final project, which is a month long, individually-paced activity to create a 5-minute video presentation answering questions related to a dataset of their choosing using Python-based visualizations.

Throughout the course, most of the modules end with a day on Social Impacts. In the first module, students review final projects from previous semesters that are relevant to their career interests, and must discuss and critique the social impacts of the video. In the second module, students watch a video about an ethical scenario where a computer potentially caused an accident, and they are asked to debate within their cohorts the ethical ramifications. The third module ends with a brief lesson about various ethical frameworks (e.g., Utilitarianism, Duty, Common Good, etc.) and students apply these frameworks to the debate from the previous module. In both the mini-project and the final project, students are expected to discuss the social impacts of the questions and answers that they determine.

[3]https://pythonhosted.org/spyder/

Students are assigned daily readings accompanied by reading quizzes. The readings, written specifically for the curriculum by the course staff[4], introduce the material for the day with a few pages of prose and images. These quizzes are short, multiple-choice assignments that are expected to take only a few minutes. They are administered and automatically graded by Canvas. Students are allowed 3 attempts before the quiz closes

The class meets twice a week for 75 minutes. A typical day of the course begins with a short introductory lecture on the material. Students then work in their cohorts or individually to complete the interactive component (e.g., paper-and-pencil activity, collaborative discussion, programming assignment, project work). Finally, the class comes back together to discuss the lessons of the day and any challenges that occurred. Most lessons include a required homework assignment meant to reinforce students' learning and prepare them for the next class.

1.4.1 Cohorts. On the third day of the course, students are grouped into collaborative cohorts of 5-6 students. Each undergraduate teaching assistant (UTA) is responsible for two cohorts. A graduate teaching assistant is responsible for managing the UTAs, and in turn is overseen by the course instructors.

Most early classwork activities are explicitly collaborative, and students are expected to generate a single answer from their cohort. During the programming section of the course, most assignments require individual submissions, but students are allowed to work together and get help. Students are forbidden from sharing their solutions, but are encouraged to share their thought process and provide support. Near the end of the course, students complete a group "mini-project" to prepare for their individual final project. This mini-project allows a cohort to create questions and visualizations collaboratively, although they are required to submit individual presentations.

2 RESEARCH QUESTIONS

Creating a new course for non-Computer Science majors provided a unique opportunity to test and grow the CORGIS project. At a high level, this course allows us to study the question: What is the impact of a Data Science context, as opposed to other course components, on student motivation and course outcomes? In order to answer this major research question, I break down the problem into multiple parts.

2.1 What is the Impact of a Data Science Context on Motivation?

In Chapter 2, I argued that a Data Science context would be able to serve multiple aspects of motivation. As previously described, the MUSIC Model states that motivation comes through five different avenues: sense of empowerment (agency), sense of usefulness, sense of success (self-efficacy), interest, and a sense of being cared for. In the Computational Thinking course, I seek to understand how a students' motivation to engage in the course changes over the semester, particularly in relation to the content, context, and scaffolds of the course.

(1) What course components are particularly effective or not effective at providing motivation?
(2) How does students' motivation change over the course of the semester?
(3) How does motivation compare between genders?
(4) How does Data Science appeal to students compared to other potential introductory course contexts?

[4]http://think.cs.vt.edu/book/

2.2 What is the Impact of a Data Science Context on Course Outcomes?

Although motivation is an important end goal, course designers consider a range of course outcomes when evaluating the success of a course. When designing our new course, we were interested in achieving a number of objectives beyond motivation. These objectives establish the criteria for the success of a course oriented around Data Science. In Chapter 2, I reviewed literature that argues that increased motivation will lead to improved course outcomes. Therefore, I am also interested in whether students' motivation has any connection to course outcomes, whether in terms of correlation or prediction. Ultimately, I look at course outcomes to answer two questions:

(1) Is a course designed around a Data Science context viable with respect to key course outcomes?
(2) What aspects of motivation correlate with course outcomes at the beginning and end of the semester?

In order to do this analysis, I will collect several different course outcomes:

Course Performance Measured as their final grade in the course as a percentage out of 100 in categories of the assignment types: attendance, classwork, homework, projects, and cumulative course grade.

Project Performance A number based on the students' performance on the final Data Science project, assessed via a rubric.

Procrastination Measured as a number indicating how many times the student handed in an assignment late.

Intent to Continue Students' self-reported interest in continuing engaging with computing.

3 DATA COLLECTION

Data has been collected over five semesters. During that time, the type and quality of the data has changed.

Figure 3 gives an overview of the data collection timeline. Different kinds of data are collected over the duration of the course. At the start of the semester, student demographics are collected to support our learner analysis. A set of motivational surveys are administered at the beginning, middle, and end of the course. In the second module, and then later at the end of the course, a number of assessment instruments are used to measure learner outcomes. During the BlockPy module, we collect a large quantity of user interaction log data as students write solutions to simple programming problems. Attendance is also logged throughout the semester. Finally, a number of participation-based classwork, homework, and other course indicators are collected during most phases of the course. Classwork, homework, and reading quizzes do not occur during Modules 5 and 6 because students work on projects.

All data was collected with full compliance from our Institutional Review Board. Students were informed that data would be collected from the various computational tools. Consent was gathered using paper release forms. When students did not give consent to the use of their data, their data was removed from the analysis process. After collection, all data was anonymized, and are only reported in aggregate.

3.1 Surveys

Surveys were used to determine students' motivation, representing the core data of this research project. Although self-reported data has threats to validity, it is a convenient and practical way to gather quantitative data about students' attitudes towards the course. Surveys were administered every semester of the course, integrated directly into the design of the course. In Chapter 2, I

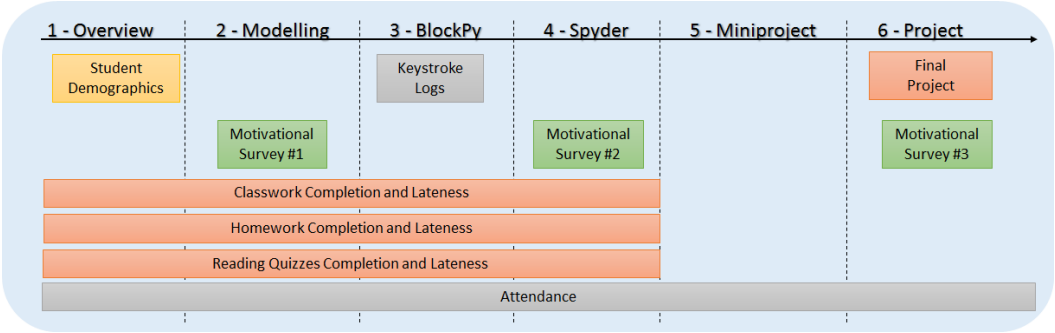


Fig. 3. Data Collection Timeline

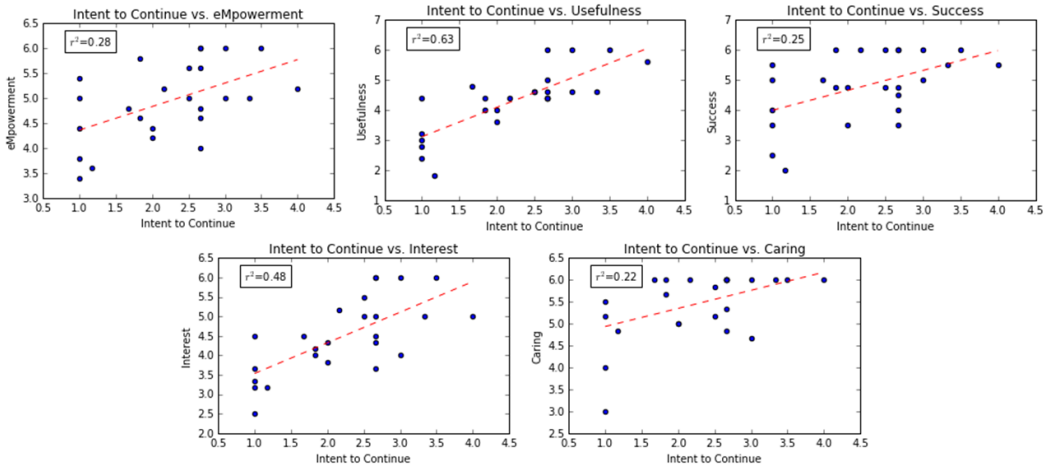
described the 5-component MUSIC model, which was the motivational framework for developing all of the surveys. The MUSIC model has an associated instrument, named the Music Model of Academic Motivation Instrument (MMAMI) or the Music Inventory, that can be used to quantitatively measure these attributes. Early on, the survey mirrored the Music Inventory [5] much more closely, sometimes incorporating exact items from the instrument. However, over time, this survey evolved to better answer specific research questions. This evolution is parallel to the evolution and growth of the course.

Surveys were administered in-class as a non-optional activity. In the early semesters, the survey was given at three points in the course (near the beginning, near the middle, and at the end). However, during the third and fourth offerings (Fall 2015 and Spring 2016), the survey was only administered at the beginning and the end. This change was reversed in the latest offering (Fall 2016) so that finer-grained data could be gathered. Students received classwork credit for participating in the survey (regardless of what answers they gave). Although students were required to complete the survey, the first question of the survey asked if they consented for their responses to be used for research purposes. In order to match up survey responses with students' course outcomes, students were required to include their email address – however, this information was stripped before analysis so that students would be anonymized. To further protect students, survey data was not analyzed until after final grades were assigned.

All of the data before the Spring 2016 semester should be considered preliminary results, and was largely used to guide development of the course and the survey. For example, data (shown in Figure 4) gathered in Spring 2015 suggests that students' sense of the usefulness of the material is more closely correlated with their intent to continue in computing than their interest in the material (N=35). The data on which that figure is based comes from a survey more strongly based off MMAMI, which means that it only asks about motivation at the course level (referring generically to "coursework"), not at the course component level (as in, referring to specific course components). The r^2 between each MUSIC component and the students' reported intent to continue was calculated using linear regression, and are shown in the figure in the top-left corner of each graph. Usefulness had the highest coefficient of determination at .63, followed by Interest at 0.48; both of these were much higher than eMpowerment, Caring, and Success, which were all below .3.

The current version of the survey was created in Fall 2015, and has been used since then. This survey uses more in-depth questions to identify students' intent to continue, uses a more sophisticated series of questions to isolate student motivation with respect to course components, and uses a Likert scale with more points. The complete text of the current survey is included in Appendix A. However, to summarize the survey, there are three main parts.

Fig. 4. Students Intent to Continue Computing vs. Their Self-reported Motivation



3.1.1 Motivation × Course Components Questions. The first part of the survey is five series of five 7-point Likert statements (Strongly Disagree to Strongly Agree). Each series relates to a different part of the MUSIC model ("I believe it will be interesting to..."), and each Likert statement relates to a different part of the course:

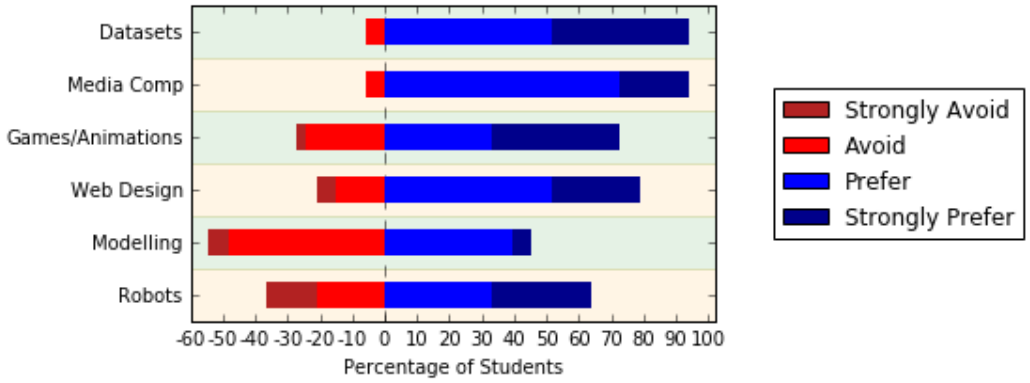
- "... learn to write computer programs" - course content related to algorithms.
- "... learn to work with abstraction" - course content related to abstraction.
- "... learn about the social impacts of computing" - course content related to social ethics.
- "... work with real-world data related to my major" - course context related to Data Science.
- "... work with my cohort" - course scaffold related to the collaborative nature of the course.

So, for example, a student would be asked to rate their agreement with a statement such as, "During the course, I felt that it was useful to my career to learn how to program" or "During the course, I felt it was interesting to learn about social ethics of computing". These five elements were chosen as some of the most clear and important elements of the course that would be visible and understandable to a student at both the beginning and ending of the course. Although there are other course elements that would be desirable to gather data on (e.g., the block-based environment), there is a limited number of questions that we can ask students. These questions get at the three main course content objectives, the course context, and one of the most visible scaffolds available to students. As the cross-product of the five motivational components and the five course components, this results in a total of 25 questions.

The 7-point Likert scale was chosen to increase the continuity of the data. Technically, data collected through Likert scales must be treated as ordinal or nominal data, rather than continuous. However, by using a large number of options, along with a sufficiently large sample size, it becomes more appropriate to use parametric statistical tests [7]. Many of the analyses favor nonparametric tests, but in some cases parametric tests and continuous measures (e.g., the mean instead of the median) are used instead.

3.1.2 Course Outcomes Questions. The second part of the survey instrument asks three sets of questions related to student course outcomes. These questions are also 7-point Likert scales. Unfortunately, the last two sets of questions (preference for course contexts and overall rating)

Fig. 5. Student Preference for Various Possible Introductory Contexts (Spring 2015)



were not included in the Spring 2016 survey, and so does not present any baseline. In the Spring 2016 version of the instrument, students were asked a single question: whether they intended to continue to learn more computing (either formally or informally), recommend the class to others, or to apply their new knowledge in their career. In the Fall 2016 version, this first question was split into three separate questions, to better refine student responses. The 7-point Likert scale offers students a choice from "Strongly Agree" to "Strongly Disagree".

The next quantitative question introduced in the Fall 2016 version asks students about their preference ("Strongly Prefer" to "Strongly Avoid") for various introductory course contexts. These contexts were chosen based on alternative ways that the course could have been taught: working with data sets related to their major (Data Science), working with media (Media Comp), making games and animations (Game Dev), making websites (Web Dev), making scientific models (Modelling), controlling robots (Robots), and making phone apps (Mobile Dev). Figure 5 shows the preliminary results from an almost identical survey question in the Spring 2015 version of the instrument, albeit on a 4-point Likert scale instead of a 7-point scales. The small sample size makes this data unsuitable for more summative analysis, but was useful at the time for verifying that the course's trajectory was on a good track.

The last set of quantitative questions asked students to give an overall rating for the course and their instructor. Although I was not particularly interested in students' assessment of the instructor, student satisfaction with the course is a relevant course outcome. These two questions are rated on a 7-point Likert scale from "Terrible" to "Excellent".

3.1.3 Free Response Questions. The third part of the survey instrument has five open-ended qualitative questions relating to the components of the MUSIC model, phrased to ask for particularly extreme examples of motivating and demotivating aspects. For example, the first question is "So far, what parts of the course seem particularly interesting or boring to you?" While the first half of the survey provided structured data about the components of the course, this section provided data about the "stand-out" parts. The data collected was analyzed using a grounded theory qualitative coding method to establish recurring themes in the course components – for instance, a large percentage of students reported that the block-based environment made them feel particularly successful.

3.2 Assessment and Other Course Data

A number of course outcomes are based on assessments and activities completed in the course. In this subsection, the general grading schema of the course is described, along with the individual final assessments.

3.2.1 Course Performance. The final course grade was calculated based on a combination of students' completion of classwork, homework, and reading quizzes, their attendance in the course, and their project scores. Classwork and homework, representing the largest time span and quantity of work, make up the largest proportion of the final grade, at 60%. Attendance and reading quizzes make up comparatively less, at 10% each. This leaves 20% total for all of the projects.

Classwork and homework that were not gradeable by automatic means were graded manually by the teaching assistants. Graded assignments were given individualized feedback, and students were strongly encouraged to resubmit corrected work without penalty. This represents a mastery-based learning approach, which was hoped would encourage students to learn the material more closely than they would otherwise. Reading quizzes were structured as automatically verifiable multiple choice questions so that they did not need manual grading.

Daily attendance was recorded using a Canvas plugin by the Undergraduate Teaching Assistants on a daily basis. The attendance score was calculated as a ratio of the number of days the student attended class and the number of days in the semester (typically 28 days). If a student had an excused absence (e.g., because of illness), the day was subtracted from the number of days in the semester for their final attendance score.

The course used a two-tier system for managing late assignments: due dates and lock dates. Reading quizzes are due before a class starts, classwork is due at the end of class, and homework was due before the start of the next class (projects have variable due dates depending on the project). However, submissions for a module become unavailable at the lock date, one week after the module ends. Students are not penalized for submissions past the due date, but are strongly urged and frequently reminded to follow the due dates as closely as possible. When a student fails to submit work by the lock date, they receive a zero on the assignment. Late submissions are recorded automatically within Canvas and used to calculate the frequency of students' lateness.

3.2.2 Final Project Performance. As previously described, the final project for the course is a month-long assignment where students use a dataset typically related to their career interests to conduct a computational analysis and answer questions. Once they have developed answers to their questions, students create a 5 minute video presentation (e.g., by using PowerPoint). This presentation is assessed using an 8-item rubric, which students are encouraged to reference while they are developing the presentation. The components of the rubric are as follows:

Abstraction What is the relationship between a real-world entity and the data?
Data Structure How is the data for the project organized? What fields in the data are particularly relevant to the stakeholder?
Questions What questions were explored?
Visualizations How did students graphically present the results of their analysis?
Answers What were the answers to the questions? How does the student interpret them?
Limitations What are limitations of the data provided, and the kinds of analyses that could be performed?
Social Impacts Who is interested in the results of the analysis, and who is affected? What potential concerns or problems can arise with regards to the analyses' impact?
Communication Were the slides well constructed? Did the student convey them clearly?

The complete text of the project rubric is available in Appendix **??**. Appendix **??** gives an example project created by the instructor, shown to students.

4 DATA ANALYSIS

In Section 4.3, I listed a set of seven sub-questions, divided into two categories, that would be used to answer the last overarching research question of my dissertation: What is the impact of a Data Science context, as opposed to other course components, on student motivation and course outcomes? Each subsection in this section presents data and associated analyses that can answer one of these sub-questions. As previously described, the data for this research project was collected over five semesters, but primarily only the latest and largest course offering will be analyzed to answer these questions – the constantly evolving nature of the course makes it tenuous to compare most results across semesters. Table 1 gives an overview of the demographic data for the five iterations of the course captured so far. In an ideal experimental setting, one would find balanced representation across majors, years, and genders, and the sub-groups would be both large and equally sized. Although the demographics are not quite ideal (the course makeup is dominated by women and freshmen), the large sample size still makes this the most compelling population to review results for.

4.1 Effective Course Components for Motivation

What course components are particularly effective or not effective at providing motivation?

The five major components of the course provide varying opportunities throughout the semester to foster motivation in the five different aspects of the MUSIC model. Figures 6, 7, 8, 9, and 10 show (respectively) students' interest, perception of usefulness, success expectancies, perception of instructors' caring, and sense of empowerment across course components. The five groups of bars correspond to a major course component: learning to Program, learning to work with Abstraction, learning about the social impacts of computing (Ethics), working with real-world Data related to their major, and working with their Cohort. Each group has three rows, one for each time the survey question was administered. To answer this subquestion, consider the last bar in each group, representing their self-reported motivation at the end of the semester.

To supplement these quantitative results, the free response questions were qualitatively coded based on positive and negative versions for each of the MUSIC components. Table 2 shows the positive codes and Table 3 shows the negative codes. The three columns divide the results by the three times the survey was administered over the semester. The codes are clustered by the positive MUSIC elements (Easy, Empowering, Useful, Interesting, and Cared For) and the negative MUSIC elements (Hard, Limiting, Useless, Boring, and Uncared For). Each individual code is given along with the frequency of that code in parentheses. Only codes that appeared five or more times were included. Note that there were no codes that appeared five or more times for the "Uncared For" tag, across the entire semester. Again, for this question, we only consider the results of the last column (the last survey administered).

4.1.1 Interest. Figure 6 indicates that students were generally interested in all five course components. The Data Science context, however, was particularly interesting to students, with no negative results. Working within a cohort was also viewed positively. Interest in learning the course content (programming, abstraction, and ethics) had more negative results, but not a substantial number (less than 10%).

Many students thought that programming was interesting, with no corresponding vocal minority saying that it was boring. However, abstraction was identified as boring. BlockPy was considered boring by a small group of students, but more than twice as many considered it interesting to

Table 1. Summarizations of Demographic Data in the Course over Time

(a) Gender of Students in the Computational Thinking Course

	Fall 2014	Spring 2015	Fall 2015	Spring 2016	Fall 2016	Total
Female	7	21	11	24	62	125
(column %)	(27%)	(54%)	(35%)	(48%)	(61%)	(50%)
Male	19	18	20	26	40	123
(column %)	(73%)	(46%)	(65%)	(52%)	(39%)	(50%)
Total	26	39	31	50	102	248
(row %)	(10%)	(16%)	(12%)	(20%)	(41%)	(100%)

(b) Year of Students in the Computational Thinking Course

	Fall 2014	Spring 2015	Fall 2015	Spring 2016	Fall 2016	Total
Freshmen	3	5	4	9	50	71
(col%)	(12%)	(13%)	(13%)	(18%)	(49%)	(29%)
Sophomore	7	12	5	18	19	61
(col%)	(27%)	(31%)	(13%)	(36%)	(19%)	(25%)
Junior	8	11	11	16	18	63
(col%)	(31%)	(28%)	(35%)	(32%)	(18%)	(25%)
Senior+	8	10	11	7	15	51
(col%)	(31%)	(26%)	(35%)	(14%)	(15%)	(21%)
Total	26	39	31	50	102	248
(row%)	(10%)	(16%)	(13%)	(20%)	(41%)	(100%)

(c) College of Students in the Computational Thinking Course

	Fall 2014	Spring 2015	Fall 2015	Spring 2016	Fall 2016	Total
Architecture	0	7	5	21	15	48
(column %)	(0%)	(18%)	(16%)	(42%)	(15%)	(19%)
Liberal Arts	9	24	15	24	69	141
(column %)	(35%)	(62%)	(48%)	(48%)	(68%)	(57%)
University Studies	1	0	2	2	2	7
(column %)	(4%)	(0%)	(6%)	(4%)	(2%)	(3%)
Agriculture	0	1	0	1	1	3
(column %)	(0%)	(3%)	(0%)	(2%)	(1%)	(1%)
Science	8	5	3	1	11	28
(column %)	(31%)	(13%)	(10%)	(2%)	(11%)	(11%)
Business	1	0	3	1	1	6
(column %)	(4%)	(0%)	(10%)	(2%)	(1%)	(2%)
Engineering	7	2	3	0	3	15
(column %)	(27%)	(5%)	(10%)	(0%)	(3%)	(6%)
Total	26	39	31	50	102	248
(row %)	(10%)	(16%)	(13%)	(20%)	(41%)	(100%)

work with. The prevalence of the BlockPy tag across the qualitative results make it a candidate for inclusion in future versions of the major course components – students clearly identified BlockPy as a major course component.

Fig. 6. Students' Interest with Regards to Course Components over 3 Surveys

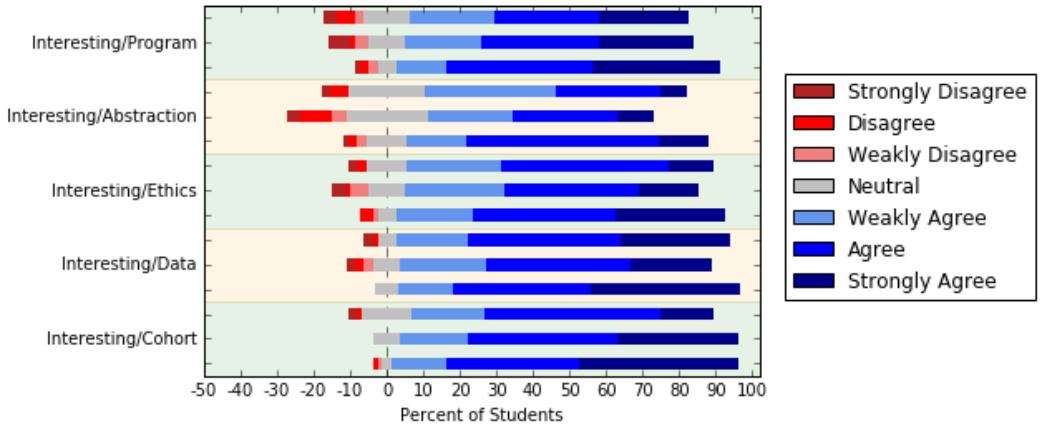
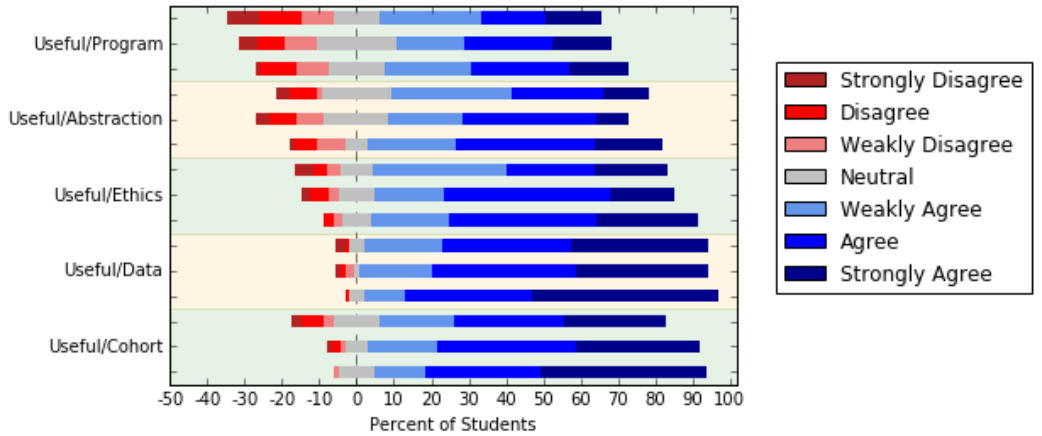


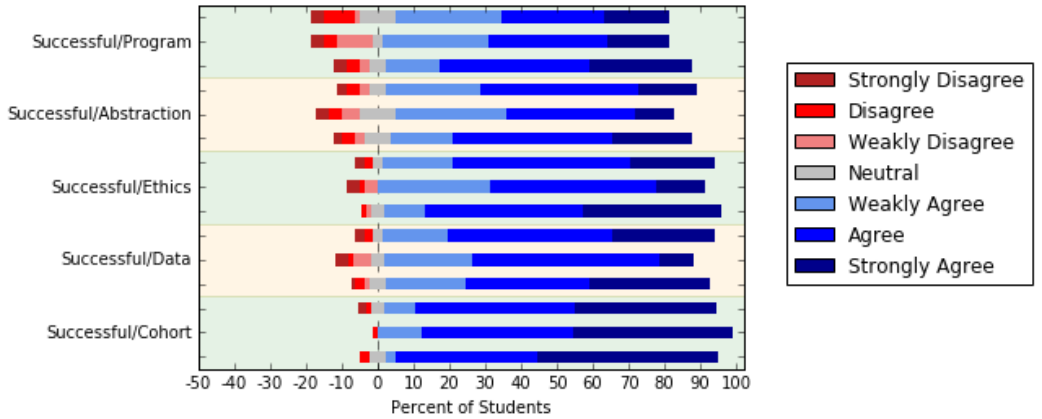
Fig. 7. Students' Perception of Usefulness with Regards to Course Components over 3 Surveys



4.1.2 Usefulness. Figure 7 shows a similar result for Usefulness as compared to Interest. The Data Science context and working within the cohort were once again almost completely positive. The course content showed more stratification. Students reported relatively high Usefulness for learning about Ethics in computing; in fact, the number of positive responses was actually comparable to the Usefulness of working within a cohort. However, the number of positive responses for the Usefulness of the Abstraction material was lower (by almost 10%), and Programming material was even lower still (by more than 20%). This validates the core value of the Data Science context: even if students are not particularly motivated by the course content, the course context can provide a sense of Usefulness.

Although many students suggested the Usefulness of learning to program, there was a vocal minority that felt it was particularly useless to their long-term career goals. The rest of the qualitative results for the Usefulness category reinforce the value of the Social Impacts material, the Data Science context, and working within cohorts. Even Abstraction makes an appearance in the list

Fig. 8. Students' Perception of Their Success with Regards to Course Components over 3 Surveys



of popular codes. The high frequency of positive Usefulness tags gives further credence to the argument that the course is designed as a Useful experience for learners.

4.1.3 Success. Figure 8 shows similar levels of self-efficacy across all the course components. Students seemed to find themselves more successful with regards to the Ethics material and working within a cohort, but relatively less successful with Programming, Abstractions, and working with Real-world Data. This matches my own understanding of the difficulty of the course components, so the results are unsurprising. The encouraging takeaway from this figure is that most results are positive, indicating that most students did not consider the material too challenging for themselves.

Unlike the other negative qualitative tags, the Hard group had considerably more items, both compared to other negative tags and to the positive Success tag. However, it is worth considering that "Hard" and "Easy" are not necessarily negative and positive in the way that tags like "boring" and "interesting" are. Ideally, a course should be in a zone of "just hard enough", rather than being particularly hard or easy. Still, it is useful to see what students are vocal about, although the results are somewhat polarizing. Roughly 16.6% of the students suggested that programming was hard, and about half as many as that suggested that programming is easy. Meanwhile, 6.8% students suggested that the social impacts were difficult while 4.9% suggested that the social impacts were easy. Considering the relative difficulty of Programming course content versus Social Impacts, the skew is not particularly surprising. Curiously, despite receiving the highest percentage of negative quantitative results, Abstraction did not appear frequently in the free response section. Also missing was working with data, which suggests that the technological scaffolding for the Data Science context was appropriate.

4.1.4 Caring. Figure 9 presents an interesting result. Almost universally across course components, students felt that instructors cared that they learned that component. Similarly, there were no high frequency codes for the "Uncared For" tag. One interpertation of this is that the instructors did an excellent job at expressing their caring for the students' learning. Of course, as a question that rates the instructor, students may be influenced to give biased answers. The major advantage of including this question in the analysis is to attempt to control for students' appreciation or dislike of the instructor. The idea being, if the student had particular feelings about the instructor, they would express it in this question rather than in one of the other four MUSIC components.

Fig. 9. Students' Perception of Instructors' Care of their Learning with Regards to Course Components over 3 Surveys

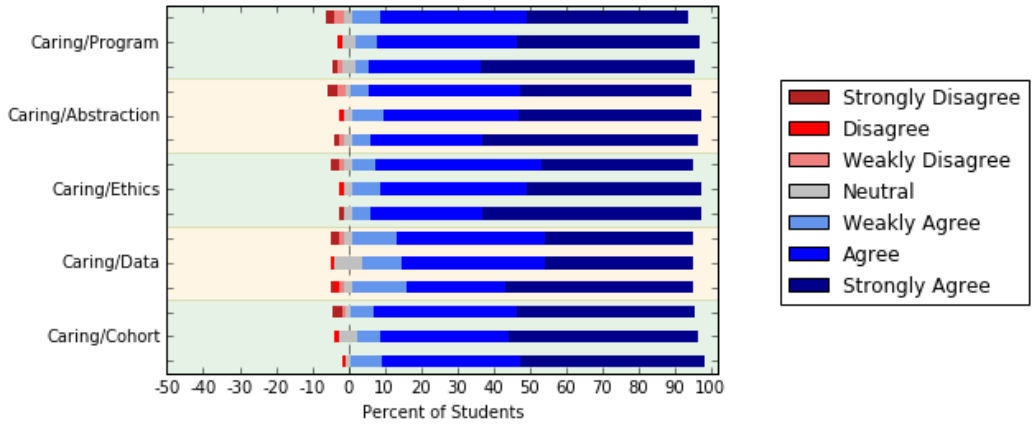
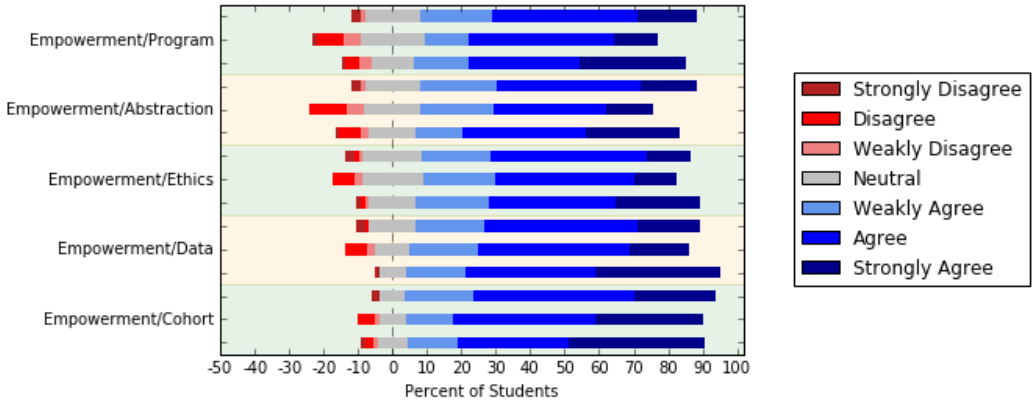


Fig. 10. Students' Perception of their Empowerment with Regards to Course Components over 3 Surveys



Ultimately, the results shown have little to tell about students' motivation with regards to Data Science as a context, but at least suggest that the instructors themselves were not a weak element of the course.

4.1.5 Empowerment. Figure 10 continues the pattern established in the Interest and Usefulness questions. The Data Science context provided the most sense of empowerment, followed by the Ethics material and working within a cohort. Programming provided less empowerment, but more so than Abstraction. It is important to recognize that the differences between the Abstraction and Data components are only about 10% in total. In general, this shows that the course empowers the students, even if there are slight differences between the components.

Students point to the variety of datasets available as one of the most empowering features of the course. However, a vocal minority of students wanted more datasets, or felt that the datasets they had were insufficient; unfortunately, students did not make recommendations for new data sources or identify gaps in the existing dataset selection. There is a continuing struggle to provide a sufficient

Table 2. Frequency of Positive Codes across MUSIC Components and Surveys

Category	First	Middle	Last
Easy	everything (11) abstraction (6) homework (5)	blockpy (11) programming (10)	blockpy (17) programming (8) social implications (5)
Empowering	data (18) nano (12) multiple solutions (10) everything (9) maze (8) deadlines (7)	data (17) everything (5) projects (5)	data (20) projects (14) multiple solutions (5)
Useful	data (17) cohort (10) abstraction (9) computers (9) new thinking (9) problem solving (7) visualization (6) programming (6) social implications (6) algorithms (5)	programming (23) data (12) social implications (12) python (7) new thinking (7) abstraction (6) problem solving (6) cohort (6)	programming (28) social implications (15) data (13) new thinking (9) cohort (8) abstraction (7) everything (7)
Interesting	maze (27) abstraction (12) programming (12) real world (9) algorithms (7) social implications (6) new thinking (6) visualization (5)	programming (23) python (16) blockpy (10) data (8) real world (6) social implications (5)	programming (25) blockpy (12) data (9) spyder (8) social implications (8) real world (6) python (6), projects (6) majors (5), cohort (5)
Cared for	learning (20) TA (17) answers (16) enthusiasm (10) feedback (5) attentive (5)	helpful (33) learning (23) TA (14) answers (10) office hours (9) instructors (8) communication (8) walking (7) questions (7) enthusiasm (6)	helpful (29) instructors (24) TA (23) answers (15) learning (14) office hours (14) communication (7) enthusiasm (6) walking (5)

number of datasets, and it is difficult to determine when a saturation point has been reached. Regardless, many students responded quite favorably to their freedom in selecting questions and visualizations when completing their projects. Although more work is needed to expand the CORGIS collection, these results are encouraging.

Table 3. Frequency of Negative Codes across MUSIC Components and Surveys

Category	First	Middle	Last
Hard	algorithms (9) nothing (7) maze 10 (7) maze (7) programming (6) precision (5)	programming (17) spyder (11) python (6) transition (5) dictionaries (5) visualization (5)	programming (17) spyder (13) social implications (7) python (7)
Limiting	data (5)		data (6)
Useless	nothing (10) programming (8)	programming (17) nothing (5)	programming (16)
Boring	lecture (5)	abstraction (6) social implications (5) visualization (5)	abstraction (6) blockpy (5)
Uncared For			

4.2 Trends in Student Motivation

How does students' motivation change over the course of the semester?

The previous subsection analyzed the quantitative and qualitative results from the last survey, in order to highlight the importance of students' motivation at the end of the course. However, student motivation is not a fixed trait, but one that develops over the course of the semester in response to the learning experience. In this subsection, I note trends and patterns in the data related to students' motivation. It is important to keep in mind the placement of the three surveys. The first occurs shortly after the start of Module 2, on Day 7 (right before they begin working with BlockPy). The second is administered at the end of the class before the Mini-project begins (right after they have finished working in Spyder on the practice problems). The third is administered on the final day of the course (after final projects, and all other course work, have been submitted). Most of the motivation/course components pairs followed one of two patterns: either they constantly increased, or they followed a $\vee$-shaped curve (indicating a decrease in the middle of the semester, and a jump at the end). The frequency of the latter pattern may suggest a more general pattern of fatigue during the middle of the semester. Figures 6, 7, 8, 9, and 10 show these trends in each pairing of course components and motivational components.

Interest in learning about abstraction, learning about ethics, and working with real-world data all follow the aforementioned $\vee$ shape: they increase from the start of the semester to the end, but with a sharp decrease in the middle. Interest in programming follows a more monotonic pattern, although there is little increase from the start to the middle; although throughout the qualitative data, there is a recurring presence of students strongly interested in programming. It is difficult to infer the trend from the start to the middle: one possibility is that students are generally less motivated during the middle phase of the course, where they are completing daily programming problems, and forget the other course content and context. It is possible that the Data Science context is even a detrimental force, as students become bored with the repeated problems dealing with the same few data sources (weather, earthquakes, classic books, and theater data). The qualitative data seems to reinforce this conclusion, with a number of tags related to this repetition. It is curious that none of this fatigue extends to their interest in programming, but it is worth noting that their interest was already relatively lower than most of the other course components. Although the first pair of

surveys are difficult to explain, the second pair show a more positive result. Given the timing of the surveys, the sharp increases suggest a powerful effect from the open-ended final project, as students regain interest in the context and content.

Students' sense of Usefulness increased for each component consistently across the semester, with the exception of learning about Abstraction. Much like Interest in learning about Abstraction, that element followed the $\vee$ shape, suggesting once again that the middle section of the course caused student motivation towards Abstraction to drop off. The qualitative data shows an even more interesting trend: at the beginning of the course, students more vocally ascribed Usefulness to working with data and learning to work in a cohort. However, by the middle of the course (and repeated at the end), students became more vocal about the Usefulness of learning programming and Social Implications. Although working with Data stayed frequent, working in a cohort slipped down dramatically. Although students perceived that the context has tremendous value in terms of Usefulness, it seems that many students also percieve an innate usefulness to programming.

Students' self-efficacy follows a pattern similar to students' Interest. Abstraction, Social Impacts, and Data follow a $\vee$ shape, while Programming stays neutral before increasing. Working within a cohort follows a $\wedge$ shape, decreasing overall during the course. The quantitative data seems to be somewhat at odds with the qualitative data, which suggests that programming was among the hardest elements of the course, even to the end. To explain this, it is helpful to consider the nature of the survey questions: the quantitative data asks if students felt they were successful, while the free response question asks what parts of the course were particularly hard or easy. One interpretation of these results, therefore, is that even though the programming may have been difficult, students' felt they were up to the challenge. It is interesting to note the positive impact of the projects on student self-efficacy across the contexts and content components.

Empowerment follows the most distinct $\vee$ shaped trend during the semester, in every component. The trough of this trend is most likely because the middle survey comes when students have just spent the past few weeks completing many small programming problems, fairly constrained assignments compared to the freedom of the final project. This design for the course was intentional, to give students a consistent and finely-tuned base of knowledge to prepare them for the open-ended, self-directed final project. Still, the context of working with real-world data, particularly in the projects, came across in both the middle of the semester and the end within the free response data. It is telling that students only suggested "multiple solutions" (referring to how many programming problems can be solved in different ways) during the beginning and end of the course; this reflects the fairly narrow bounds of the middle section of the curriculum. Somewhat discouraging is that the overall trend of empowerment, with respect to learning to program and abstraction, is negative; this is somewhat alleviated by the overall positive trend of the data context.

Students' sense of caring had high values consistently across the entire semester. Almost from the beginning, students pointed out a number of ways that instructors showed they were interested in their learning, were helpful in guiding them, or provided steps to the answers. Although this result is pleasant, it does not say much about the value of the Data Science context.

4.3 Motivation across Genders

How does motivation compare between genders?

Gender remains an important issue in introductory computing, particularly because of the low percentage of women entering and staying in computing [2]. Therefore, it is both useful and interesting to identify differences in motivational reactions between genders. Although other demographic indicators were collected, the relatively small population (roughly 100 students) makes it difficult to compare other subgroups such as major, college, or even year. Even dividing

the population in half affects assumptions needed to use parametric tests (that the sample size is sufficient to invoke the central-limit theorem). Therefore, I use the non-parametric Kruskal-Wallis test for each motivation/course component pair, for the first and last surveys, between each gender. A significance level of $\alpha < .05$ was chosen to determine if results were significant. Table 4 shows the survey items where there was a significant difference between male and female responses. Each row of the table indicates the survey (first or last), motivational component, and course component that was different; the last column indicates the absolute difference in means between the two genders, and the penultimate column indicates which gender was higher. The absolute mean difference can be roughly taken as the differences between individual levels of agreement on the Likert scale: a difference of 1, for example, represents the change from "Agree" to "Strongly Agree".

Table 4. Significant Results of Kruskal-Wallis Test for Motivational and Course Components across Multiple Surveys between Genders

Survey	Motivational Component	Course Component	Higher	Difference in Means
First	Caring	Abstraction	Females	0.47
First	Caring	Ethics	Females	0.50
First	Caring	Cohort	Females	0.61
First	Caring	Program	Females	0.62
First	Caring	Data	Females	0.63
First	Successful	Cohort	Females	0.74
First	Usefulness	Data	Females	0.95
Last	Successful	Program	Males	0.87

At the beginning of the semester, women scored significantly higher in a number of categories. Most obviously, the entire group of items related to the Caring motivational component had women scoring roughly half a level of agreement higher. In general, this suggests several possibilities. One possibility is that women were more attentive and open to the instructors' attempts to make students feel cared for. Another is that men were actually less cared for than women and they accurately perceived this (though neither instructor reported anything like this). A third possibility is that women overestimated the amount of caring by their instructors. Women were also more than a half level higher in terms of their self-efficacy to work in a group, suggesting a similar set of possibilities: perhaps that the female students were more attuned to the group work, that the male students were less cared for, or that the women overestimated. Finally, female students felt that it would be more useful to work with real-world data, by almost a full level of agreement. This is consistent with results that suggest female students may be more driven by introductory material that can be connected to solving real-world problems [2]. If so, this can be seen as a major benefit of the Data Science context.

However, over the course of the semester most differences in gender disappeared, leaving only a single item with a significant gender difference (which, coincidentally, was also the only result where the males scored higher). By the end, males were almost a full level of agreement higher than females in terms of their self-efficacy at learning to program. It is impossible to tell if males are over-estimating their skills or women are under-estimating, but it should be noted that there was no significant differences between genders in actual performance of most course activities. It is also worth noting that women scored significantly higher on the final project, according to a One-way ANOVA, by almost a full letter grade (9.36 point difference in means). Although it is spurious to connect students' ability to program too closely to these outcomes, it is still telling that students seem to have gender-based differences in ability to self-evaluate.

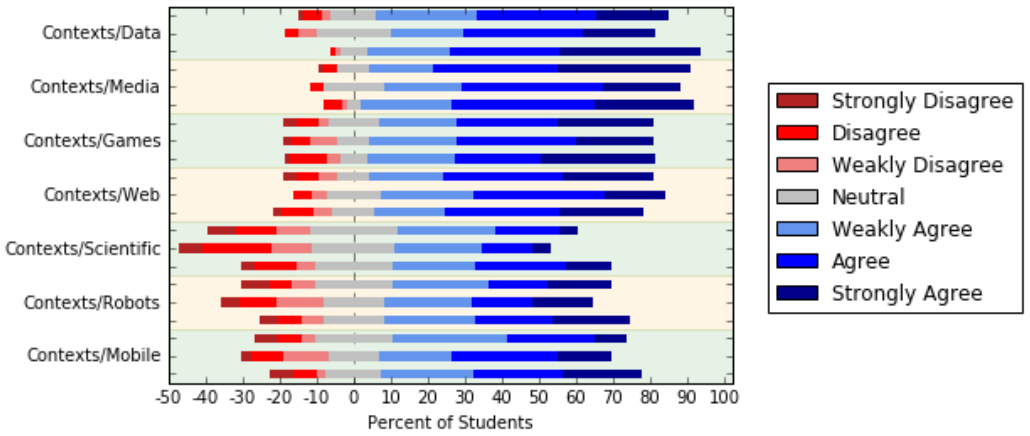


Fig. 11. Students' Preference for Potential Course Contexts over 3 Surveys

4.4 Data Science vs. Introductory Contexts

How does Data Science appeal to students compared to other potential introductory course contexts?

Figure 11 shows the results of a survey question about students' preference for various potential course contexts. Each of the potential contexts was briefly described to students in a short sentence (e.g., Media Computation was described as "Working with pictures, sounds, movies"), and students were asked how much they would prefer or avoid a course taught in that context. Scientific computing was easily the most poorly rated context, followed by controlling robots and then Mobile App development; these three contexts were significantly ($\alpha < .05$) lower than both Media Computation and Data Science. Although Media Computation was consistently a popular context, the Data Science context was the most preferred choice by the end of the course (although there was no significant difference between the two in a matched-pairs T-test).

This survey question revealed a number of interesting significant differences between genders (non-parametric Kruskal-Wallis test, with $\alpha < .05$). Men rated roughly a full level of preference higher than women for the contexts of Game Design, Controlling Robots, Scientific Modelling, Mobile App Development, and Web Development. These results seem to strongly be in favor Data Science and Media Comptuation as preferred contexts for designing gender-agnostic courses.

There are three major threats to validity from the methodology for answering this question. First, students were given little explanation for the contexts. Although the contexts are described in a sentence, there is hardly time to give concrete examples or go into any serious depth. It is difficult to know what students imagine these contexts to be like, especially given their inexperience with computing in general. Second, students know that the instructors are biased in their own preference, since they are enrolled in an introductory computing course constructed around a Data Science context – the students may simply be telling the instructors what students think the instructors want to hear. Third, it is possible that some students may be biased themselves in favor of a Data Science context, since they may be taking the course specifically for this pre-existing reason.

4.5 Viability of a Data Science Context

Is a course designed around a Data Science context viable?

Although academic motivation within a course is important, there is a fundamental question of whether a course structured around a Data Science context can work. In order to answer

this question, I loosely follow criteria established by Guzdial [3] when he evaluated his Media Computation curriculum. Specifically, I take his Retention, Gender, Learning and More-Computing hypotheses as different ways of critically analyzing a course. The Plagarism Hypothesis (that there would be fewer reported incidences of cheating) was not evaluated for our course, since we had no expectations or problems related to plagarism.

The Media Computation project collected data over a 10 year period in order to evaluate their hypotheses, so it is still early to compare our Data Science course, which has been around for a quarter of that time. These hypotheses are also being evaluated post hoc, which means that different criteria may have been imagined when designing the Data Science course, compared to the Media Computation course; it is possible that other criteria may be more appropriate. Still, the results should prove to be a barometer of the progress of the course so far.

4.5.1 Retention Hypothesis. The first hypothesis evaluated was the Retention Hypothesis. This hypothesis uses a measure called "DFW" rate - the number of "D" and "F" grades, combined with the number of students who withdrew, divided by the total number of students initially enrolled [3]. Ideally, a course should minimize the DFW rate (without artifically decreasing the courses' difficulty). Guzdial targeted a failure rate of no more than 15%, and achieved an actual rate of around 12%. These numbers were established to compare the Media Computation course with existing introductory computing courses (which were not contextualized for non-majors).

Virginia Tech collects and publishes data on grade distribution and drop-out rates across every course [9]. This makes it easy to compare the Introduction to Computational Thinking course with courses that fulfill similar requirements. Figure 12 shows the grade distributions of courses across the university that also fulfill the Quantitative and Symbolic Reasoning credit (also known as "Area 5" courses). Also included is the traditional introductory computing class for Computer Science majors ("Intro to Software Design"). Our own course is rendered in bold ("Intro to Comp Thinking"). The colors of the bars represent clustering performed via a K-Means algorithm to group courses with similar DFW percentages.

As can be seen, the Computational Thinking course is in the lowest DFW% group, at roughly 8%. The black error bars (indicating the standard deviation) show that the vast majority of the offerings were below the 15% threshold successfully. This compares well with the Media Computation course, and favorably to the traditional introductory computing course. Interesting to also note is the "Comp Sci, Lib Arts" course, another attempt at an introductory computing course for non-majors, which shares a DFW% more similar to the Media Computation course than our own.

4.5.2 More-Computing Hypothesis. The second hypothesis tested in [3] was the More-Computing Hypothesis, which tested whether students would continue to take computing courses after the Media Comp course. Unfortunately, our Data Science course has no direct successor. Further, the relatively young nature of the course makes it difficult to gather meaningful longitudinal data that speaks to students' continuing to another CS course. However, included in the survey as an engagement outcome is a set of three questions meant to determine whether students intend to continue with computing. Figure 13 shows students' agreement with the following three statements:

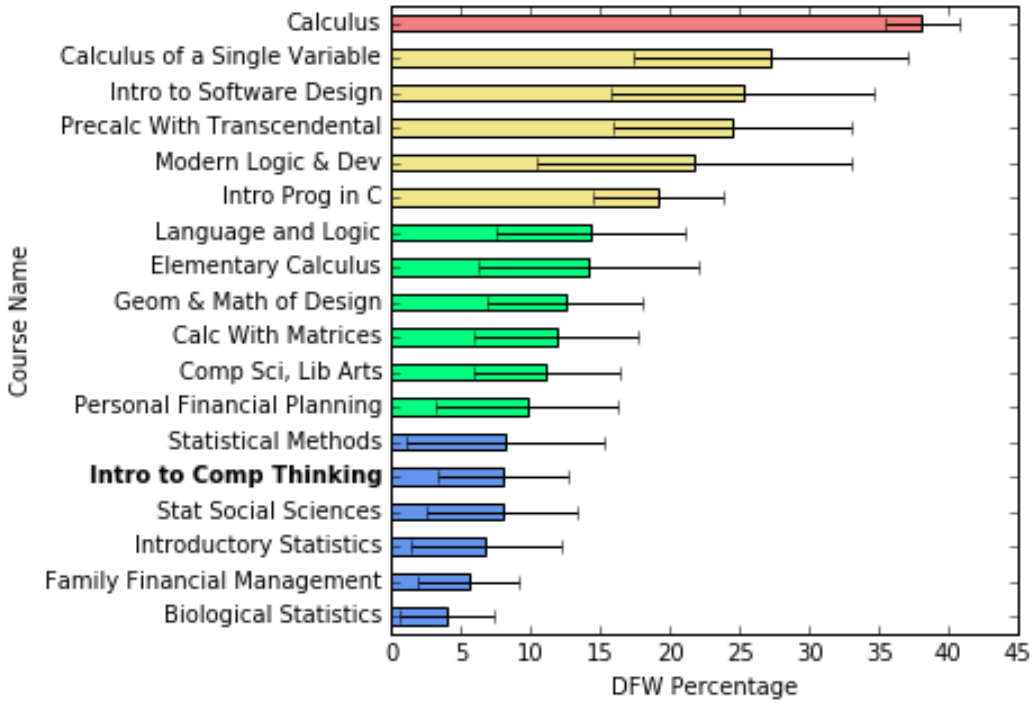
Continue/Apply "I will directly apply what I have learned in my career."

Continue/Learn "I will try to learn more about computing, either through a course or on my own."

Continue/Recommend "I will recommend this class to others."

Only half the respondents indicated an intent in continuing to learn about computation and apply computation to their long-term careers. However, closer to three-quarters of the respondents were willing to recommend the class to their friends. It is difficult to understand what concrete

Fig. 12. Percentage of DFW (Ds, Fs, and Withdrawls) across All Area 5 Courses since Fall 2014



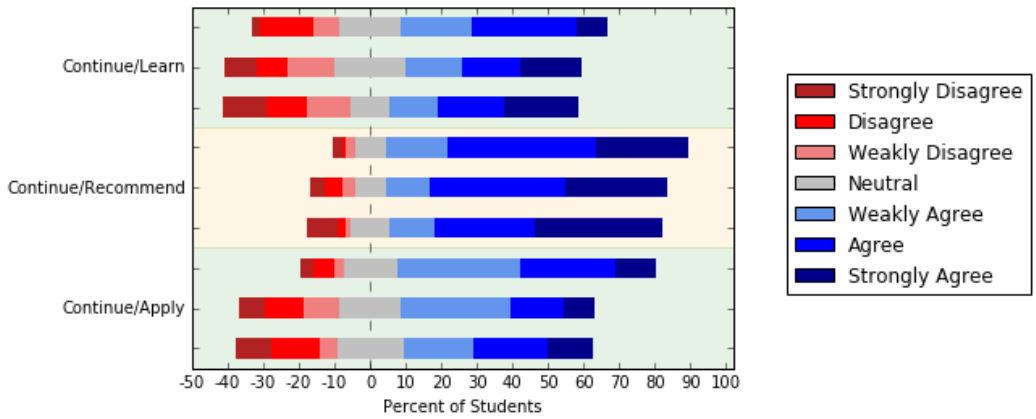


Fig. 13. Student Response to Intent to Continue Questions over 3x Surveys

action this will translate into – will students enroll in more computationally-oriented courses? Will they take advantage of free online courses? Further research is necessary to determine whether students actually follow through on their intent. However, we believe that it is useful to be able to set goal posts for such future research.

The trend in student responses over the semester is consistent across the three separate questions. The number of students intending to continue or recommend decreased from the first survey point to the second. The number of students intending to continue or recommend stayed constant from the middle to the last surveys, but students on both sides became more strongly polarized. That is, students who previously indicated a "weak agreement" with the statement now indicated "agreement", students that indicated a "weak disagreement" now indicated a "disagreement", and so on. The project helped solidify students' intent to continue, while the programming assignments generally decreased their intention.

The different form of our data makes it unsuitable to compare our own course to Media Computation for this hypothesis. It is worth pointing out that Guzdial considered their course to not meet the expectations for the More Computing Hypothesis. Of course, neither of our courses consider the more computing hypothesis a particularly high-value objective [4]. Still, measuring and improving the number of students interested in continuing in computing is an ongoing research direction worth pursuing.

4.5.3 Gender Hypothesis. Gender imbalance in Computer Science continues to be a serious problem, impacting the culture and productivity of the discipline. As a course explicitly for non-majors, the course designers hoped to draw in genders equally. Reviewing the historic data in Table 1, the instructors were pleased to find that roughly 50% of the students across the five semesters were women. In fact, the most recent iteration of the course was 61% women. This level of gender parity is close to the ideal, although there is hope in the future to more closely approach an even split. Reviewing the results of the Media Comp course, it can be seen that the Media Computation course achieved similar demographics.

4.5.4 Learning Hypothesis. The last measure, the Learning Hypothesis, suggests that students would achieve some level of proficiency at the end of the semester. Guzdial notes the many difficulties in meaningfully comparing knowledge across introductory courses [8]. Efforts to create an accurate instrument have struggled, and there is still disagreement about what exactly should be measured and whether the quantity measured is appropriate [6].

It is beyond the scope of this dissertation to introduce a conclusive instrument for measuring student performance across introductory courses. Instead, I present the formal assessment of the course to attempt to measure student performance at the end of the semester. These results are reported here in order to at least justify that some level of learning occurred within the course. Earlier, we established three major course learning objectives:

(1) Students should be able to create algorithms that manipulate data using a practical programming language (Algorithms).
(2) Students should be able to navigate and describe data (Abstractions).
(3) Students should be able to discuss social impacts of their computations (Social Impacts).

The major form of assessment for these objectives was the Final Project, which was graded using an 8-item rubric. Figure 14 shows the results for each rubric element, which are quite positive. The majority of students (85%) received an Excellent or Good rating on each rubric element.

The Algorithms learning objective relates to the Questions, Answers, and Visualizations components of the rubric. Students performed worst on the Questions component out of all of the components, where they are tasked with creating questions that can be solved through computational analysis of their data. Although the vast majority of students had no problems creating questions, many students failed to properly justify their questions and argue for their importance using evidence. Fortunately, most students were able to competently create visualizations and use them to answer the questions they generated. Most students lost points because they used

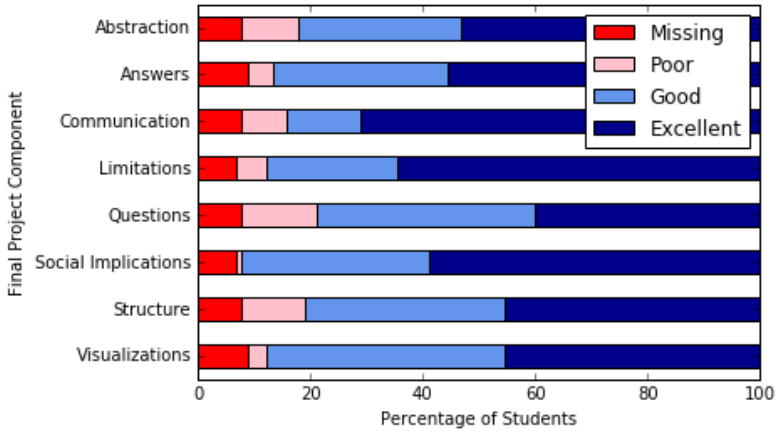


Fig. 14. Final Project Grade Distribution (F16)

an insufficient breadth of visualizations, insufficiently labeled their diagrams, or interpreted the results of their visualization incorrectly.

The Abstractions learning objectives relates to the Abstraction, Limitations, and Structure components of the rubric. Although the results are also positive here, some students did struggle with explaining the finer details of their data and justifying their relevance to their project. Although they could often list some relevant fields of their data that they used in their projects, they sometimes had difficulty connecting those abstracted fields to real-world entities and ideas. In terms of the structure of their data, students were easily able to identify the types of individual fields, but often gave confusing explanations for the lists and dictionaries those fields were composed in. Students' limitations were sometimes too generic rather than specific to their data – for example, students would simply report that their project was limited by the quantity or detail of the data, rather than a unique characteristic.

The Social Impacts objective related directly to the Social Implications component of the rubric. Students were capable of identifying and justifying stakeholders who would find their results relevant, but many students failed to explain conflicts between those stakeholders using the ethical frameworks they had previously learned.

Without externally validated instruments for measuring computational knowledge, it is difficult to conclude that learning did or did not occur within our course. Certainly, students performed at some arbitrary level of proficiency. The instructors were satisfied by this level, although there is hope to continue improving these outcomes. Of course, given that many of these students were non-STEM majors with no prior computing background, the fact that many of these students completed the project successfully is compelling, albeit informal, evidence of learning. Ultimately, I leave it to the reader to determine for themselves if they are satisfied with these measures.

4.6 Correlating Motivation and Course Outcomes

What aspects of motivation at the beginning and end of the semester correlate with course outcomes?

In Sections 3.1.2 and 3.2, I established a number of course outcomes that I hoped to connect to the motivational data. Each of these outcomes was compared to each motivation/course component pair to find their Pearson correlation, which should indicate the association between the two measures. A significance level of $\alpha < .05$ was chosen, and only significant results are included in this section.

Table 5 shows the correlation for data collected during the Spring 2016 semester in a pair of tables. The first table represents the data for one course outcome: students' intent to continue (which in that semester, was represented by only one survey item). Each column represents a Motivational element of the MUSIC model (eMpowerment, Usefulness, etc.) and each paired row represents a course component. Each paired row consists of the result from either the first survey or the last survey. The intersection of a row and column represents the correlation of that course/motivation component with the Intent to Continue course outcome for one of the two surveys. This format was chosen due to the high number of significant correlations related to this particular course outcome, and to highlight a particular trend in the types of motivation/course components that have a high correlation with this outcome. In particular, it can be seen that the course content (Programs, Abstraction, and Ethics) are the only course components with any correlation: the Data Science context does not have any statistically significant association with students' intent to continue. The highest correlations are from the students' sense of Usefulness of the material, followed by their Interest and Self-efficacy (which is predicted by Guzdial in reviewing Eccles work on educational decision-making [3]). This suggests that, when attempting to influence students' intent to continue with computing, the context is less important than the content. This is contrary to original hypotheses that the course context would have significant connections to students' intent to continue learning.

Table 5. Correlation between Course Outcomes vs. MUSIC/Course Components in the Spring 2016 Semester

(a) Pearson's Correlation between Students' Intent to Continue vs. MUSIC/Course Components in the Spring 2016 Semester across Two Surveys (First and Last)

Component	Survey	M	U	S	I	C
Abstraction	Last	**0.455***	**0.690***	**0.706***	**0.504***	0.176
	First	0.25	**0.537***	**0.582***	**0.463***	-0.209
Cohort	Last	0.059	0.087	0.002	0.044	0.057
	First	-0.157	0.236	0.08	0.228	0
Data	Last	0.105	0.293	0.276	0.276	0.202
	First	-0.011	0.023	0.031	-0.055	-0.307
Ethics	Last	0.284	**0.465***	**0.465***	0.298	0.165
	First	0.05	0.21	0.162	0.015	-0.155
Programs	Last	**0.441***	**0.828***	**0.701***	**0.648***	0.052
	First	0.224	**0.497***	**0.466***	**0.534***	0.009

(b) Pearson's Correlation between Various Course Outcomes vs. MUSIC/Course Components in the Spring 2016 Semester across Two Surveys (First and Last)

Course Outcome	Motivation	Course Component	Survey	Correlation
Final Project	Caring	Data	First	-.383
Quizzes Grade	Successful	Data	First	.452
Final Grade	Successful	Programs	Last	.401
Final Project	Successful	Programs	Last	.458
Final Project	Useful	Programs	Last	.419
Final Project	Successful	Abstraction	Last	.345
Attendance	Successful	Abstraction	Last	.335
Attendance	Useful	Ethics	Last	-.343

The second table holds the remaining course outcomes. Most of these correlations are low, and so are not particularly compelling. In particular, most of them make little sense – why does a students' perception of the instructors' caring whether they learn to work with data negatively correspond to their Final Project grade? Why would students' sense of the usefulness of learning ethics negatively correspond to their attendance? Although explanations could be hypothesized, I consider it more likely an artifact of applying statistical tests across too many features, which can incorrectly identify significance when there is none. For the remaining outcomes, there is some credence to the idea that students sense of their own Successfulness in the course content would be correlated with course outcomes such as the Final Project and the Final Grade. However, none of the results seem to have much importance.

Table 6 shows the significant correlations for the major course outcomes in Fall 2016. First, it is necessary to point out the relatively low correlations in the table, much like the previous semester, suggesting limited trust-worthiness of the data. In fact, none of the outcomes are particularly well-correlated with motivation/course component items, either at the beginning of the semester or at the end. That said, it is telling that the most highly correlated items are those related to learning about programming – particularly with regards to Interest and Usefulness. The most well-correlated outcome is again the Intent to Continue questions (applying computation and learning computation). The only two cognitive outcomes with relatively high correlations are students' Final Grades and their Classwork grades, which correlate with students' self-efficacy at learning to program – which suggests only that students have some ability to self-assess. Many of the previously established cognitive outcomes fail to even appear in the table, representing the limited connection found between motivation and cognitive outcomes.

Table 6. Correlation of Outcomes vs. MUSIC/Course Components in Fall 2016 Semester

Course Outcome	Motivation	Course Component	Survey	Correlation
Attendance	Caring	Program	First	.316
Recommend Course	Interesting	Cohort	First	.304
Continue Applying	Interesting	Program	First	.368
Continue Applying	Useful	Program	Last	.390
Continue Applying	Caring	Program	Last	.375
Continue Applying	Caring	Abstraction	Last	.377
Final Grade	Successful	Program	Last	.396
Quizzes	Successful	Abstraction	Last	.304
Classwork	Successful	Program	Last	.354
Continue Learning	Interesting	Program	First	.405
Continue Learning	Interesting	Program	Last	.341
Continue Learning	Useful	Program	Last	.406
Continue Learning	Successful	Program	Last	.354

Overall, the major result that I glean from this data is that the course content, particularly with regards to students' interest and sense of usefulness, are the most important components connected to students intent to continue. These two semesters worth of results seem to suggest that few connections can be drawn from motivational data and cognitive course outcomes. However, the engagement course outcomes (i.e. intent to continue) can still be seen as valuable targets, if nothing else.

REFERENCES

[1] Austin Cory Bart, Javier Tibau, Eli Tilevich, Clifford A Shaffer, and Dennis Kafura. 2016. Implementing an Open-access, Data Science Programming Environment for Learners. In *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, Vol. 1. 728–737.

[2] Wendy M DuBow, Beth A Quinn, Gloria Childress Townsend, Rosario Robinson, and Valerie Barr. 2016. Efforts to Make Computer Science More Inclusive of Women. *ACM Inroads* 7, 4 (2016), 74–80.

[3] Mark Guzdial. 2013. Exploring hypotheses about media computation. In *Proceedings of the ninth annual international ACM conference on International computing education research*. 19–26.

[4] Mark Guzdial and Allison Elliott Tew. 2006. Imagineering inauthentic legitimate peripheral participation: an instructional design approach for motivating computing education. In *Proceedings of the second international workshop on Computing education research*. 51–58.

[5] B. D. Jones. 2009. Motivating students to engage in learning: The MUSIC model of academic motivation. *International Journal of Teaching and Learning in Higher Education* 21, 2 (2009), 272–285.

[6] Andrew Luxton-Reilly. 2016. Learning to program is easy. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*. 284–289.

[7] Geoff Norman. 2010. Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education* 15, 5 (2010), 625–632.

[8] Allison Elliott Tew. 2010. *Assessing fundamental introductory computing concept knowledge in a language independent manner*. Ph.D. Dissertation. Georgia Institute of Technology.

[9] Virginia Tech Office of Institutional Research & Effectiveness. 2016. Distribution of Grades, Average QCA by Course and Instructor. (2016). Available from: http://www.ir.vt.edu/.

Received February 2007; revised March 2009; accepted June 2009