Delineating Factors that Influence Student Performance in a Data Structures Course

Halil Bisgin, Murali Mani, Suleyman Uludag
Department of Computer Science, Engineering and Physics, University of Michigan-Flint, Flint, Michigan, 48502
Email: {bisgin,mmani,uludag}@umich.edu

***Abstract*—- (Full paper, Research category) The stakeholders in the computer science education have been trying to pinpoint the most relevant factors and variables contributing significantly to student recruitment, success, and retention in computing degrees over the last several decades. The goal of these efforts is to intervene using these controllable variables as early as possible and at key transition junctions for the most benefit to the students. Yet, there is still not a general agreement, let alone a consensus, neither on the most important factors nor on the degree of their contributions. While we cannot provide a panacea to this elusive and challenging problem, in this paper, we report our initial study, analyses, and results of what we intend to transform into a longitudinal undertaking to delineate success factors in computing programs in general, and in CS2 (data structures) courses in particular. Our analyses confirm some of the findings from the literature, such as CS1 performance as the most important contributor for CS2 success as well as prior programming experience, and perception of computing. At the same time we diverge from some of the literature on such factors as high school GPA or CS0 performance. Our conclusions corroborate one of the rare consensus in the computing education literature that more efforts should still be exerted in this important topic. We also provide a new and novel taxonomy of the factors for success in computing programs as well as a short summary of the ACM Guidelines over time.**

***Index Terms*—CS0, CS1, CS2, Success factors for introductory computing courses.**

I. Introduction

Computer science education literature has documented significant differences across universities with regards to how the core programming courses up to the data structures are offered including the content, number of courses, order of the topics, the programming language, pedagogical methodology, prerequisites, delivery methods, etc. ACMs 1978 Computing Curricula introduced the CS1 (first course about the programming fundamentals) and CS2 (data structures and abstraction) terminology. This terminology has been used in the literature by educators without a consensus about the content of these courses. CS0 designation has been added later to denote a computing class for non-majors or a prerequisite course for CS1. In this paper, we report the results of an assessment study conducted at a regional campus of a major public state university to delineate relevant factors of student success in a CS2 course. The results span over the past six regular consecutive semesters since curricular changes were introduced in our CS2. The study considers both the institutional data, such as demographics, prior academic exposure and performance in both computing and mathematics, and student self-reported data, such as involvement in computer science research and clubs, motivation, and perception about the difficulty of material.

In our curriculum, we offer different pathways for students to take CS2. Our CS2 course is offered in C++ and our students in CS2 come from one of the two majors: Computer Information Systems (CIS) or Computer Science (CSC). All our students take at least two programming courses before CS2: (a) a 100-level course in C++ that covers basic control structures, and (b) one of two 200-level courses that cover object oriented programming. Our CIS majors tend to take the 200-level course in Java and our CSC majors tend to take the 200-level course in C++. Besides this, there is a CS0 course, which can be waived, and is offered either in VB, Python, or COBOL.

In order to provide the context, we survey the literature and provide a novel classification of different factors influencing the success in CS2 course, and hence in the overall computing degree and retention rates. We provide a detailed multifaceted statistical analysis of the data to provide insight into the intricate relationships among various factors. We believe that our study provides a good point of reference and a streamlined look at the introductory computing pathways as well as the CS2 success factors and sets the stage for more standardized compare/contrast studies in the future. More concerted efforts are needed to be able to make more inroads towards a better understanding of the most relevant and controllable factors.

The rest of the paper is organized as follows: Section II provides a summary of the related work in terms of a taxonomy of factors reported in the literature for the success in computing classes and programs. We provide an overview for the evolution of the ACM Curriculum Guidelines since its inception in Section III. The methodology we employed in collecting data, the success factors, and the framework of our statistical analyses are explained in Section IV. We provide a detailed explanation of the results in Section V followed by a discussion of the major points in Section VI. Section VII concludes the paper.

II. Related Work

Fig. 1 provides a novel taxonomy of the factors reported in the literature as contributing to success in introductory computing classes, including CS2, where $\mathcal{ID}$ indicates that

978-1-5386-1174-6/18/$31.00 ©2018 IEEE

corresponding item came from our institutional data, and Qx

- **(1) Data-Driven**
  - Academic
    - Credits Taken [1] $\mathcal{ID}$
    - GPA $\mathcal{ID}$
    - Prior Programming [2]--[5] $\mathcal{ID}$, $Q6 - Q11$
    - Prior Computer Experience [6] $Q5, Q13, Q14$
  - Observational
    - Programming Behavior
      - Syntactical [7]--[10]
      - Semantical [11], [12]
    - Compilation Behavior [8], [10]
    - Typing Behavior [13]
    - Keystrokes [14]
  - Demographic and Background Traits $\mathcal{ID}$, $Q1 - Q4$
- **(2) Test-Driven**
  - Academic
    - Computing
      - CS0 [1], [15]--[19] $\mathcal{ID}$
      - CS1 [20] $\mathcal{ID}$
      - Early Term Data [21], [22]
    - Standardized
      - ACT [15], SAT, etc.
    - Mathematics [1], [2], [15], [23] $\mathcal{ID}$ and Science
  - Psychological and Social [12]
  - Cognitive [24], [25]
    - Metacognitive [26]
- **(3) Self-Reported (Behavioral)**
  - Perception of Computing [6], [27] $Q6, Q15, Q17$
  - Comfort Level [2], [3], [28]
  - Work [2], [3] and Learning Style
  - Goal-orientation [26], [29]
  - Self-efficacy [2], [3], [26], [30]
  - Motivation [31] $Q17$
  - Self-esteem [32]
  - Attribution of Success [2]
- **(4) Pedagogical**
  - Subgoal Learning Theory [4]
  - Cognitive Apprenticeship [33]
  - Feedback [34], Peer Assessment [35]
  - Early/Late Objects [28], [36]
  - Supplemental and Peer Instruction [29], [37], [38]
  - Flipped Classroom [39]
  - Team-based Learning [40]

Fig. 1: Factors Contributing to Success in Computing Classes. denotes the source in our study came from our survey question x. We also note that a big portion of the degree attrition takes place during these introductory computing classes and thus these factors also determine the success rate of computing programs as well. We recognize 4 major categories of factors. Data-driven factors are those that can be obtained from a variety of sources such as institutional analysis departments or the registrar about the basic information on students. Test-driven factors are those that can be obtained after students take curricular or extra-curricular tests. The third major category is about the behavioral approaches of the students and they are usually self-reported by them through surveys or questionnaires. The final category contains such pedagogical factors as different curriculum delivery methodologies or learning and teaching styles employed in the classroom.

III. ACM Guidelines for undergraduate Computer Science and Information Systems Curricula

Since the 1960s, ACM, along with other leading professional and scientific computing societies, has provided curricular recommendations for computing disciplines. The current recommendations found at [41] provides undergraduate curriculum guidelines for five sub-disciplines of computing: (a) computer engineering (b) computer science (c) information systems (d) information technology, and (e) software engineering. At our university, we offer two majors: computer science and computer information systems. Therefore we focus on the development of the ACM curricula guidelines for computer science (CS), and information systems (IS) over the years.

According to the 1973, 1983, 1997 and 2002 recommendations [42]–[45], programming, data, file and object structures are required to be covered in the IS curriculum. The 2010 recommendations suggest that programs that want to include a sequence of programming courses can do so similar to the computer science curricular guidelines. The topics related to programming that are recommended to be covered include programming control structures, algorithm development, classes, and data structures such as arrays, stacks, queues and trees. The mathematical topics that are recommended to be covered include introduction to calculus, introductory statistics and discrete mathematics.

The different ACM recommendations for computer science include the 1968, 1978, 1991, 2001 and 2013 [46]–[50]. While there are differences between the different recommendations, they all share several commonalities as well. For instance, they all require math topics such as introductory calculus, probability, linear algebra and discrete structures. Further, they all include basic programming concepts, search and sort methods, recursion and data structures.

The 1991 report, which was the first joint report by ACM and IEEE-CS, identifies programming to denote the entire collection of activities that surround the description, development, and effective implementation of algorithmic solutions to well-specified problems, and that programming is an extension of the basic skills that students and professionals normally use in day-to-day communication. Therefore, undergraduates should develop an early understanding of programming.

The 1978 curriculum recommends a four course sequence for programming courses: Computer Programming I (introduce problem solving methods and algorithm development),

followed by Computer Programming II (introduce algorithmic analysis, string processing, recursion, search and sort methods, and simple data structures), followed by introduction to file professing (introduce bulk storage devices), followed by data structures and algorithmic analysis (to utilize algorithmic analysis in the selection of methods for data manipulation). The 2001 report identifies six possible implementations of the programming curriculum: (a) imperative-first (b) objects-first (c) functional-first (d) breadth-first (e) algorithms-first, or (f) hardware-first.

It was observed in the 2013 report that safer or more managed languages such as Java were being preferred (over C), dynamic languages such as Python and Javascript are also being used. Further, Visual languages like Alice and Scratch, are being favored to provide a syntax-light introduction to programming, mainly for non-majors. Also, in order to dispel the notion that there is a single best programming language, some introductory course sequences provide a presentation of alternative programming paradigms, such as scripting .vs. procedural programming, functional .vs. object-oriented programming etc.

IV. Methodology

In this study, we targeted a group of students who have taken CS2, data structures, since Fall 2015. Prior to Fall 2015, we offered CSC 375 as a course covering both data structures and algorithms, and as a core course for our CSC majors. There were two significant curricular changes that were introduced in Fall 2015: (a) CSC 375 was split into two courses – CSC 375 that covered only data structures (similar to ACM CS2 terminology), and CSC 379 that covered algorithms, and (b) CSC 375 was added as a core course for CIS majors as well. Students who take CS2 (our revised CSC 375 course) have taken different combinations of courses because of the different majors (CIS and CSC), as well as because of the choices available. In our study, we investigate factors relevant to student success in CS2. We define *success* in a course as the academic performance (grade) in that course. Therefore success is not a binary outcome, rather, we study how successful the students are (higher course grade means greater student success), and the factors relevant to higher course grade. In our study, we first focused on the institutional data such as different courses taken and the grades obtained in the different courses. Secondly, we incorporated available institutional data with self-reported data, which were collected through a survey.

A. Subjects

We had 147 students that had a grade for CS2 over the past six semesters since Fall 2015. This includes 140 undergraduate students in two majors (CIS and CSC), and 7 "fast-track" graduate students who had an undergraduate degree in a non-CS field. As for the self-reported data, we sent out a questionnaire that included 17 questions to find out their prior exposure to programming in different settings, motivation, perception, and involvement in related extra-curricular activities. We collected responses from 62 students (58 undergraduate and 4 graduate) who had a grade for CS2, for which we performed statistical analysis for the combination of aforementioned two types of data.

B. Success Factors

Since each student taking CS2 has a history of pre-requisites in terms of Math and CS courses, we retrieved their grades for five MATH courses (MTH 111, MTH 118, MTH 121, MTH 122) and six CS courses (CIS 173, CIS 170, CSC 122, CSC 175, CSC 275, CSC 276). Due to the major/program differences, students can choose either MTH 118 or MTH 122 for Calculus concepts which made us create a new variable called MTH82 to quantify math success. Students also have the option to choose any of the CS0 level courses (Visual Basic, COBOL, and Python) from which we took the highest grade and named it CS0. In our curriculum CS1 is offered as a two course sequence. Students first learn basic control structures using C++ programming language in CSC 175 (we refer to this as CS1A). After this, students learn object-oriented programming concepts in either C++ (CSC 275) or Java (CSC 276). We took the highest grade if a student took both CSC 275 and CSC 276 and denoted it as CS1B. The course variables/factors used in the paper and the different courses that correspond to each variable are shown in Table I. If a variable corresponds to more than one course, we choose the highest grade that a student received among these courses. The variable CS1A denotes the grade obtained in CSC 175; the variable CS1B denotes the higher grade between CSC 275 and CSC 276. Note that students typically would have taken only one of the courses corresponding to such a variable.

TABLE I: Course variables and corresponding courses. The value for a variable denotes the grade obtained in the corresponding course(s).

Variable	Course Number (Credit Hours) and Title
CS0	CSC 122 (2): Introduction to Programming (Python) CIS 170 (3): COBOL Programming CIS 173 (3): Visual Basic for Windows
CS1A	CSC 175 (4): Problem Solving and Programming I (C++)
CSC275	CSC 275 (4): Problem Solving and Programming II (C++)
CSC276	CSC 276 (4): Problem Solving and Programming II (Java)
CS1B	CSC 275 (4): Problem Solving and Programming II (C++) CSC 276 (4): Problem Solving and Programming II (Java)
CS2	CSC 375 (3): Data Structures (C++)
MTH111	MTH 111 (3): College Algebra
MTH118	MTH 118 (4): Calculus for Management and Social Sciences
MTH121	MTH 121 (4): Calculus I
MTH122	MTH 122 (4): Calculus II
MTH82	MTH 118 (4): Calculus for Management and Social Sciences MTH 122 (4): Calculus II

The institutional data included grades for the different courses (Table I), as well as the students' high school GPA (variable denoted HSGPA). In addition to the variables derived from institutional data, we came up with 17 questions in Table II by which we aimed at incorporating student perspective into our analysis.

TABLE II: Survey Questions.

Q1	Are you a transfer student? (Yes/No)
Q2	If you are a transfer student, where did you transfer from? (Yes/No)
Q3	Have you switched majors before? (Yes/No)
Q4	Are you a first generation college student? (Yes/No)
Q5	Is any of your family members involved in computing? (Yes/No)
Q6	How much do you like programming? (1 is like a lot, 5 is dislike a lot)
Q7	I took programming class(es) in High School. (0 none, up to 4 classes)
Q8	I was exposed to programming in high school through other classes. (1 significantly, 5 none)
Q9	I was exposed to programming in high school through clubs. (1 significantly, 5 none)
Q10	I am/was employed in a job that involved programming. (1 significant, 5 none)
Q11	I participated in a faculty's research project with some programming tasks. (1 significant, 5 none)
Q12	I meet with my advisor before registering for courses? (1 is almost always, 5 is almost never)
Q13	I was involved with ACM or other computer science clubs. (1 significantly, 5 none)
Q14	I spend time on gaming (1 significant time, none)
Q15	How difficult did you find CSC375? (1 is very light, 5 is very heavy)
Q16	I learned/am learning a great deal in CSC375 (1 great deal, 5 minimal)
Q17	I had a strong desire to take this course (1 strongest, 5 weakest)

C. Statistical Analysis

Our analysis consists of three steps: i) correlation analysis to assess the relationship of factors and success in CS2, ii) linear regression models to see the contribution of selected factors toward the prediction of CS2 success, and iii) t-test to quantify the effect of CS1B, MTH82 and major choices in CS2 success.

In the first step, we measured the Pearson correlation between the student success in CS2 and the variables/factors we identified. For the institutional data analysis, we considered the grades for the different courses as well as HSGPA. For the self-reported data analysis, we considered the survey responses, in addition to the grades for the different courses and HSGPA. The number of students for each correlation analysis denotes the number of students with a value for that variable (for course variables, this denotes the number of students with a grade for that course). We also reported the significance level of each correlation coefficient.

In the second stage, we built linear regression models to assess the ability of each potential factor to predict CS2 success. Similar to the correlation analysis above, we built models on different set of data considering factors with higher correlations and readiness in Math and CS ($r \geq 0.5$).

In the final step, we split students into groups to measure the impact of their CS1B choice, MTH82 choice and major on CS2 success. Namely, we performed independent t-tests on the groups which were formed on the basis of CS1 selection, MTH selection and major. We had students who took either Java or C++ before CS2 in the CS1B case, students who took either MTH 118 or MTH 122 in the MTH case, and compared CSC majors against CIS majors in the third case.

V. Results

In this section, we present our results from correlation and regression analyses for the institutional and self-reported datasets. For the self-reported data analysis, we had responses from 62 out of the 147 students. This analysis included the responses as well as HSGPA and course variables. Then, we illustrate our findings from the t-test for Java and C++ comparison, MTH 118 and MTH 122 comparison, and comparison of different majors.

A. Institutional Data

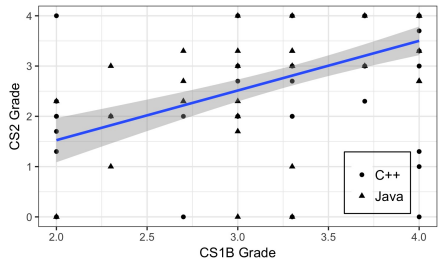
For 147 students, we retrieved all available grades for MTH and CS courses as well as their HSGPA. The sample size varies for different variables as a student may not have the grade for some courses (either the student did not choose that course as part of a choice, or the student transferred in the course from another college, or the course was waived for the student). See that in Table III, the sample size for CSC275 is 65, and the sample size for CSC276 is 43. We list the correlation coefficient for each factor in Table III in which we highlight the factors which have $r \geq 0.50$ and p-value≤ 0.05.

TABLE III: Correlations.

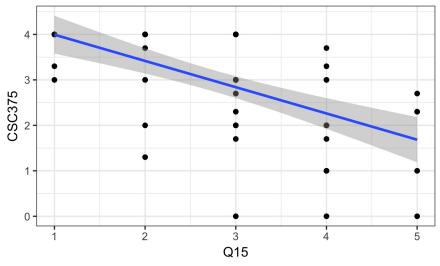
Factors	r	p-value	Sample Size
HSGPA	0.18	.36	76
MTH122	0.53	p<.001	58
MTH121	0.32	.05	67
MTH111	0.03	.82	44
MTH118	0.57	p<.001	36
MTH82	0.50	p<.001	93
CSC275	0.48	p<.001	65
CSC276	0.41	.05	43
CS0	0.19	.31	87
CS1A	0.21	.22	90
CS1B	0.53	p<.001	105

The highest correlation was 0.57 which was for MTH 118. MTH 122, CS1B and MTH82 also had correlations at least 0.50. Accompanying p-values for all these correlations were far less than 0.05, which demonstrated the significance of these relatively strong mutual relationships. We observed lower correlations for CS0 and CS1A. The relationship between CS1A success to CS2 success is illustrated in Fig. 2d. Furthermore, we summarize the performance of students in CS1A in Fig. 2e, where median grades are represented by the bold horizontal line, and 25^{th} and 75^{th} percentiles correspond to lower and upper edges of the box, respectively.

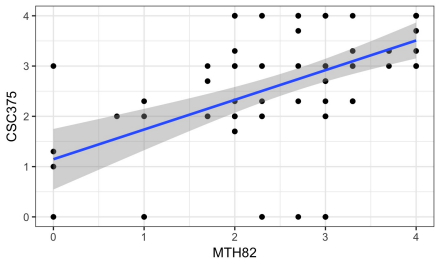
Although CS0 and CS1A did not demonstrate that strong relationship as that of CS1B, we wanted to see if the entire CS prerequisite chain could be a predictor of the success in CS2. Therefore, we built a multiple linear regression model that takes CS0, CS1A, and CS1B as independent variables or factors for the prediction of success in CS2. We performed this analysis on 55 students who had grades for these courses. As it can be seen in Table IV, CS1B stood out as the strongest



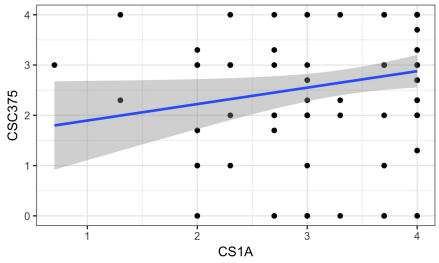
(a) CS1B Success vs CS2 Success



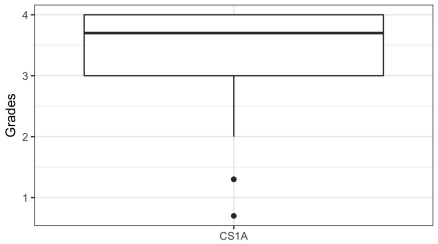
(b) Q15 vs CS2



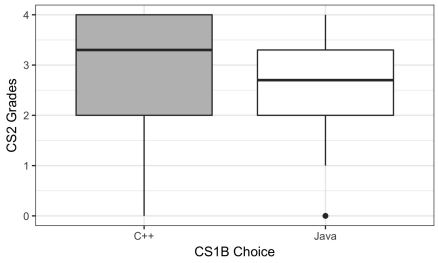
(c) MTH82 vs CS2



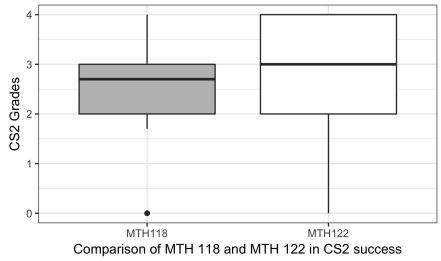
(d) CS1A vs CS2



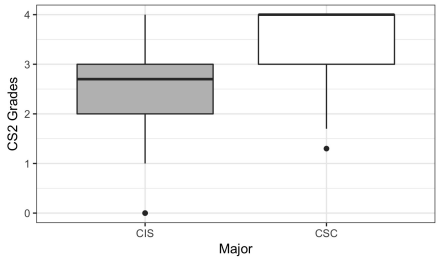
(e) CS1A Grade distribution



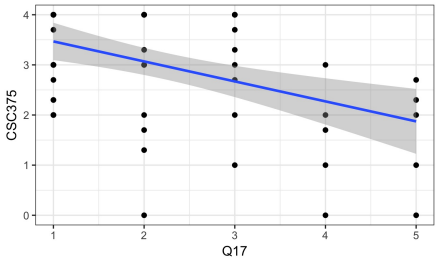
(f) Grade summary in CS2 for C++ and Java students



(g) CS2 Grades for MTH118 and MTH122 takers



(h) Grade summary in CS2 for CSC and CIS Majors



(i) Q17 vs CS2

Fig. 2: Results from our statistical analysis.

predictor with the highest coefficient with a p-value of 1.08E-07, which again showed its significance among CS courses.

TABLE IV: Computer Science Prerequisites.

	Estimate	Std. Error	t value	Pr($>\|t\|$)
(Intercept)	-1.60	1.36	-1.18	.24
CS0	0.21	0.35	0.60	.55
CS1A	0.15	0.19	0.80	.43
CS1B	0.91	0.21	4.42	p<.001
summary	df: 55, R^2: 0.33, p<.001			

As both MTH 118 and MTH 122 were found to be positively and significantly correlated and they both cover Calculus concepts for different majors, we further delved into these two Math courses. Then we combined these two courses into MTH82 as described in Section IV-B and observed a significant (p<.001) correlation of 0.50, which is also illustrated in Fig. 2c.

Although MTH82 had relatively lower correlation value w.r.t. individual MTH 118 and MTH 122 correlations, it not only showed a higher significance, but also increased the sample size. Therefore, we incorporated MTH82 as an indicator of Math-readiness along with CS courses above and built another linear model. Table V shows the summary of the multiple linear regression analysis in which MTH82 significantly (p-value: .002) demonstrates its predictive power (coefficient: 0.43) against CS courses and CS1B still plays an important role (coefficient: 0.86). This model seems to have more variance explained in CS2 success, as the R^2 value in Table V is higher than the R^2 value in Table IV.

TABLE V: Math and Computer Science Prerequisites.

	Estimate	Std. Error	t value	Pr($>\|t\|$)
(Intercept)	-1.88	1.33	-1.42	.16
MTH82	0.43	0.13	3.36	.002
CS0	0.13	0.34	0.39	.70
CS1A	0.05	0.18	0.27	.79
CS1B	0.86	0.22	3.90	p<.001
summary	df: 43, R^2: 0.5053, p<.001			

In our final analysis for the institutional data, we wanted to study a relatively long run trajectory which involved the high school GPA in addition to Math and CS factors. Due to the limitations in data availability, we excluded CS0 to increase the

sample size and built another multiple linear regression model. Our findings in Table VI still shows a statistically significant (p-value: .001) linear model with an $R^2 = 0.36$, but high school GPA does not seem to contribute as much as other predictors to the CS2 success. On the other hand, MTH 82 and CS1B continue to be strong factors with coefficients 0.45 (p-value: .01) and 0.66 (p-value: .01), respectively.

TABLE VI: High School GPA and Prerequisites.

	Estimate	Std. Error	t value	Pr($>\|t\|$)
(Intercept)	-1.16	1.32	-0.88	.39
HSGPA	0.08	0.35	0.23	.82
MTH82	0.45	0.17	2.66	.011
CS1A	0.10	0.23	0.42	.68
CS1B	0.66	0.26	2.56	.01
summary	df: 41, R^2: 0.36, p-value: .001			

B. Self-reported Data

As we mentioned earlier, we had 65 students who responded to 17 questions, 12 of which were in Likert scale. At the time of this study, there were three students with missing grades for CS2 course, which resulted in 62 as the sample size. Therefore, we calculated Pearson correlations for the questions from 6 through 17 in addition to the available institutional variables such as high school GPAs and grades for those 62 students. Table VII lists our findings with corresponding p-values and sample sizes.

TABLE VII: Correlations.

Factors	r	p-value	Sample Size
HSGPA	0.19	1.0	36
Q6	-0.25	.71	62
Q7	0.19	1.0	62
Q8	0.20	1.0	62
Q9	0.02	1.0	62
Q10	-0.08	1.0	62
Q11	0.05	1.0	62
Q12	0.19	1.0	62
Q13	0.00	1.0	62
Q14	-0.22	1.0	62
Q15	-.61	p<.001	62
Q16	-0.14	1.0	62
Q17	-0.44	.009	62
MTH122	0.64	.016	24
MTH121	0.44	.18	33
MTH118	0.70	.13	13
MTH111	0.04	1.0	17
MTH82	0.62	p<.001	37
CSC275	0.55	.016	34
CSC276	0.52	1.0	12
CS0	0.30	1.0	31
CS1A	0.40	.15	42
CS1B	0.54	.003	46

In our survey, the 15^{th} question, which was intended to ask the student perception in the difficulty level of the CS2 course, showed the highest correlation with a very low p-value (p<.001). The negative correlation indicates that as the student finds the course lighter (1 is very light, 5 is very heavy), s/he has a higher grade in CS2. We also illustrate this phenomenon in Fig. 2b. Similarly, we should also note that Q17, which asked the desire to take the course (1 strongest, 5 weakest), showed a considerable negative correlation (-0.44) as illustrated in Fig. 2i.

In this particular subset of the data, we observed that Math courses showed relatively higher correlations again, but correlation coefficient for MTH 118 did not reach the same significance level as it reached in the whole institutional dataset. Similar to the earlier analysis, MTH82, which is the combination of MTH 118 and MTH 122, had a high correlation with p<.001. We also noticed that CS1B consistently showed a positive correlation with CS2 success.

By using all available questions in the survey, we built a multiple linear regression model. We only excluded the 2^{nd} question which was specifically asking the name of the institution the student came from. Questions Q1, Q3, Q4, and Q5 in Table VIII are "Yes/No" questions for which we have coefficient estimates when their values are "Yes". Finally, we obtained a linear model with $R^2 = 0.50$ and p-value=.003, which not only reinforced the significance of Q15, but also discovered the importance of Q8, programming exposure in high school (coefficient: 0.26, p-value: .04).

TABLE VIII: Survey Questions.

survey	Estimate	Std. Error	t value	Pr($>\|t\|$)
(Intercept)	4.20	0.94	4.45	p<.001
Q1 Yes	-0.09	0.29	-0.32	.75
Q3 Yes	-0.25	0.30	-0.83	.41
Q4 Yes	-0.02	0.28	-0.06	.95
Q5 Yes	-0.25	0.29	-0.86	.40
Q6	0.17	0.15	1.17	.25
Q7	0.07	0.11	0.60	.55
Q8	0.26	0.12	2.14	.038
Q9	-0.11	0.13	-0.80	.43
Q10	0.05	0.09	0.53	.60
Q11	0.01	0.14	0.10	.92
Q12	-0.002	0.09	-0.02	.98
Q13	-0.04	0.12	-0.35	.73
Q14	-0.04	0.10	-0.36	.72
Q15	-0.52	0.13	-4.05	p<.001
Q16	0.01	0.14	0.1	.92
Q17	-0.27	0.15	-1.82	.08
summary	df:45, R^2: 0.50, p-value: 0.003			

Lastly, we wanted to see how the self-reported data and the institutional data together could help us predict the student success in CS2. More specifically, we picked Q8 and Q15 due to their significance and incorporated with high school GPA, MTH82, CS1A, and CS1B. Although high school GPA has never been found to be a significant factor, we tried to mimic the earlier analysis reported in Table VI.

Since we were limited by the number of students who responded to our survey, we performed this analysis on a relatively smaller subset, which had a sample size of 17. The resulting linear model with R^2 of 0.67 and a p-value $< .05$ showed that Q15 had the highest prediction power (coefficient: -0.71) being the only significant factor among all variables considered. Even though the coefficient for the high school GPA is the second highest, it was not found to be significant

due to its hight p-value. As opposed to earlier analyses, CS1B was not found to be an effective predictor, which had the least coefficient value with a high p-value.

TABLE IX: Significant Questions with Readiness.

	Estimate	Std. Error	t value	Pr($>\|t\|$)
(Intercept)	0.26	3.35	0.08	.94
HSGPA	0.52	0.54	0.96	.36
MTH82	0.36	0.43	0.83	.42
Q8	0.32	0.19	1.71	.12
Q15	-0.72	0.29	-2.50	.03
CS1A	-0.04	0.74	-0.06	.96
CS1B	0.22	0.42	0.53	.61
summary	df: 11, R^2: 0.67, p-value: 0.03			

C. Choice of CS1B: Java vs. C++

At our institution, our students have two choices for CS1B, which covers object-oriented programming: CSC 275 (in C++) or CSC 276 (in Java). In our institutional data, we had 63 students who took only CSC 275, 40 students who took only CSC 276, and two students who took both CSC 275 and CSC 276. For studying whether the choice of C++ or Java impacted their CS2 success, we removed those two students. Fig. 2a illustrates their success in CS2 and Fig. 2f summarizes the grades of both groups. In order to quantitatively assess whether the distribution of CS2 success is significantly different for these two groups, we performed independent t-test based on the hypotheses below.

$$H_0 : \mu_{C++} = \mu_{Java} \tag{1}$$
$$H_A : \mu_{C++} \neq \mu_{Java} \tag{2}$$

As our research question is whether the choice of programming language has different effects, the null hypothesis means that they do not have different effects. Therefore, we investigated if any difference between the means of the two groups significantly differed from zero. We then found out that means for two samples differ from each other with a p-value of .07, which concludes that difference is not significant.

D. Math Pathways: MTH 118 vs. MTH 122

While students have a choice between Java and C++ in CS1B, they are expected to take either MTH 118 or MTH 122 for Calculus II concepts depending on their majors. Therefore, students diverge at this point in the Math pathway before they all reach to the level of CS2. As it can be seen in Fig. 2g, CS2 success not only has more variance, but also includes higher scores for MTH 122 students. In order to assess the significance of the difference between these two groups, we constructed the following hypotheses:

$$H_0 : \mu_{MTH118} = \mu_{MTH122} \tag{3}$$
$$H_A : \mu_{MTH118} \neq \mu_{MTH122} \tag{4}$$

We performed the t-test on those who took only MTH 122 (n=57) or only MTH 118 (n=35), which resulted in exclusion of a student who took both. Then we found out that the difference in CS2 success is not significant (p-value: .13).

E. Impact of Major: CSC vs. CIS

As CIS program is less programming intensive, we also looked into the impact of major towards CS2 success. Major information was available for the survey data for which Figure 2h summarizes the grade distribution of each group in CS2 and visually demonstrates the gap. However, for a more quantitative assessment, we followed the exact procedure in sections above by customizing it for majors. Namely, we changed the hypotheses as follows.

$$H_0 : \mu_{CSC} = \mu_{CIS} \tag{5}$$
$$H_A : \mu_{CSC} \neq \mu_{CIS} \tag{6}$$

In this case, sample sizes for CSC and CIS majors were 38 and 19, respectively. In contrast to the CS1B analysis and the MTH pathway analysis, we found that means for two samples actually differed from each other significantly (p-value: .002).

VI. Discussion

In this study, we examined several factors which we identified through our institutional data as well as self-reported data to study their impact on CS2 success. While we aimed to measure the effect of earlier courses taken and the high school GPAs in the institutional data, we also wanted to reveal the effect of any self-reported factor through a survey.

Among the CS courses, CS1B has been the most informative factor in predicting CS2 success. In both institutional and self-reported data, which has smaller sample size, CS1B showed a moderately strong and significant correlation with CS2. As CS1B introduces object oriented programming (OOP) and it is the most recent programming course to be taken before CS2, students with a good understanding of these concepts tend to succeed in building data structures in CS2. Since choice of Java or C++ in CS1B did not show a significant difference, it can be inferred that the level of understanding of OOP concepts play more important role than the language itself.

Although CIS and CSC majors are offered the same set of programming courses, i.e., CS0, CS1A, and CS1B, Math requirements are discipline-specific. Namely, students are to take either MTH 118 or MTH 122 depending on their majors; and therefore, MTH 118 and MTH 122 can be associated with CIS and CSC majors to some extent, respectively. We observed moderately high correlations with CS2 for both courses. MTH 82, which was the combination of those two courses was found to be significant predictor for CS2 success in the institutional data. We also know from the literature that high school math is a strong predictor of success in introductory computer science courses [2] and our findings here confirm a similar fact between college level math courses and CS2 success. However, MTH 118 and MTH 122 groups did not show a significant difference in their CS2 success, whereas we found a significant difference between CIS and CSC groups in the survey data despite their association with these Math courses. Even though

CIS majors had higher average dislike score (2.68/5 vs. 2.03/5) for Q6, this discrepancy might be due to both unavailability of the grades and the difference in sample sizes.

One of the predictors in the institutional data in addition to grades was the high school GPA by which we attempted to see its predictive power towards the success in a 300-level course. Even though it was considered a good predictor for an introductory level computer science course [51] in earlier work, our analysis did not find it as a significant factor despite its positive effect.

Another important observation we made in this study is that student perception of the difficulty level in CS2 (Q15) plays a major role and goes beyond all factors with a significantly high coefficient in the linear model. Even though Q15 demonstrates slightly lower correlations than Math courses, it outperforms Math component (MTH82) of the linear model and proves its reliability across different types of analyses.

One of the caveats of our study has been the varying sample sizes due to the partially available data in the course of linear regression analyses. For instance, Q8 about programming exposure in high school, which has a low but significant correlation with CS2, did not contribute significantly to the linear model despite its high coefficient (0.32). We observed a similar phenomenon for the high school GPA. Even though it did not show any significant correlations as opposed to Q8, it had the second highest coefficient (0.52) in the same model with a very low significance. Besides such discrepancies caused by removal of some students from the analysis, we believe that our overall sample size needs to be increased for more reliable conclusions.

Given the limitation in the number of students, our current list of factors was not able to explain the total variance in CS2 success. We observed R^2 values ranging from 0.36 to 0.66, which still leave some room for improvement. We believe the unexplained portion might be either due to our selective approach, which utilized the ones which had high correlations, or unidentified factors. Regardless, it warrants further research for a better explanation.

VII. Conclusions and Future Work

In the current study, we presented existing paths leading into the CS2 courses along with the taxonomy of success factors in the literature. We further provided a summary of changes in the ACM curriculum recommendations. The results from our assessment study not only revealed the importance of OOP and Calculus training, but also warranted further research for success rate of different majors. Student responses also highlighted a strong pattern that would account for their success depending on their perceived difficulty of the course. We believe that the contributions and findings presented herein have pedagogical value for future curriculum changes.

References

[1] L. Lambert, "Factors That Predict Success in CS1," *J. Comput. Sci. Coll.*, vol. 31, no. 2, pp. 165–171, 12 2015. [Online]. Available: http://dl.acm.org/citation.cfm?id=2831432.2831458

[2] B. C. Wilson, S. Shrock, B. C. Wilson, and S. Shrock, "Contributing to success in an introductory computer science course," in *Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education - SIGCSE '01*, vol. 33, no. 1. ACM Press, 2001, pp. 184–188. [Online]. Available: http://portal.acm.org/citation.cfm?doid=364447.364581

[3] B. C. Wilson, "A study of factors promoting success in computer science including gender differences," *Computer Science Education*, vol. 12, no. 1-2, pp. 141–164, 2002.

[4] B. B. Morrison, A. Decker, and L. E. Margulieux, "Learning Loops: A Replication Study Illuminates Impact of HS Courses," in *Proceedings of the 2016 ACM Conference on International Computing Education Research - ICER '16*. ACM Press, 2016, pp. 221–230. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2960310.2960330

[5] A. Tafliovich, J. Campbell, and A. Petersen, "A student perspective on prior experience in CS1," in *Proceeding of the 44th ACM technical symposium on Computer science education - SIGCSE '13*. ACM Press, 2013, p. 239. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2445196.2445270

[6] S. Bergin, R. Reilly, S. Bergin, and R. Reilly, "Programming: Factors that Influence Success," in *Proceedings of the 36th SIGCSE technical symposium on Computer science education - SIGCSE '05*, vol. 37, no. 1. ACM Press, 2005, p. 411. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1047344.1047480

[7] E. S. Tabanao, M. M. T. Rodrigo, and M. C. Jadud, "Predicting at-risk novice Java programmers through the analysis of online protocols," in *Proceedings of the seventh international workshop on Computing education research - ICER '11*. ACM Press, 2011, p. 85. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2016911.2016930

[8] M. C. Jadud and M. C., "Methods and tools for exploring novice compilation behaviour," in *Proceedings of the 2006 international workshop on Computing education research - ICER '06*. ACM Press, 2006, p. 73. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1151588.1151600

[9] C. Watson, F. W. Li, and J. L. Godwin, "Predicting Performance in an Introductory Programming Course by Logging and Analyzing Student Programming Behavior," in *2013 IEEE 13th International Conference on Advanced Learning Technologies*. IEEE, 7 2013, pp. 319–323. [Online]. Available: http://ieeexplore.ieee.org/document/6601941/

[10] ——, "No tests required," in *Proceedings of the 45th ACM technical symposium on Computer science education - SIGCSE '14*. ACM Press, 2014, pp. 469–474. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2538862.2538930

[11] A. S. Carter, C. D. Hundhausen, and O. Adesope, "The Normalized Programming State Model," in *Proceedings of the eleventh annual International Conference on International Computing Education Research - ICER '15*. ACM Press, 2015, pp. 141–150.

[12] ——, "Blending Measures of Programming and Social Behavior into Predictive Models of Student Achievement in Early Computing Courses," *ACM Transactions on Computing Education*, vol. 17, no. 3, pp. 1–20, 8 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3135995.3120259

[13] J. Leinonen, K. Longi, A. Klami, and A. Vihavainen, "Automatic Inference of Programming Performance and Experience from Typing Patterns," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16*. ACM Press, 2016, pp. 132–137. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2839509.2844612

[14] A. Ahadi, R. Lister, H. Haapala, and A. Vihavainen, "Exploring Machine Learning Methods to Automatically Identify Students in Need of Assistance," in *Proceedings of the eleventh annual International Conference on International Computing Education Research - ICER '15*. ACM Press, 2015, pp. 121–130. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2787622.2787717

[15] M. Brown, "CS0 As an Indicator of Student Risk for Failure to Complete a Degree in Computing," *J. Comput. Sci. Coll.*, vol. 28, no. 5, pp. 9–16, 5 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2458569.2458572

[16] J. K. Doyle, "Improving Performance and Retention in CS1," *J. Comput. Sci. Coll.*, vol. 21, no. 1, pp. 11–18, 10 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1088791.1088795

[17] M. Haungs, C. Clark, J. Clements, and D. Janzen, "Improving first-year success and retention through interest-based CS0 courses," in *Proceedings of the 43rd ACM technical symposium on Computer*

Science Education - SIGCSE '12, 2012, p. 589. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2157136.2157307

[18] C. Dierbach, B. Taylor, H. Zhou, I. Zimand, C. Dierbach, B. Taylor, H. Zhou, and I. Zimand, "Experiences with a CS0 course targeted for CS1 success," in *Proceedings of the 36th SIGCSE technical symposium on Computer science education - SIGCSE '05*, vol. 37, no. 1, 2005, p. 317.

[19] C. Marling and D. Juedes, "CS0 for Computer Science Majors at Ohio University," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16*. ACM Press, 2016, pp. 138–143. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2839509.2844624

[20] H. Danielsiek and J. Vahrenhold, "Stay on These Roads: Potential Factors Indicating Students Performance in a CS2 Course," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16*. ACM Press, 2016, pp. 12–17. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2839509.2844591

[21] S. N. Liao, D. Zingaro, M. A. Laurenzano, W. G. Griswold, and L. Porter, "Lightweight, Early Identification of At-Risk CS1 Students," in *Proceedings of the 2016 ACM Conference on International Computing Education Research - ICER '16*. ACM Press, 2016, pp. 123–131. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2960310.2960315

[22] L. Porter and D. Zingaro, "Importance of early performance in CS1," in *Proceedings of the 45th ACM technical symposium on Computer science education - SIGCSE '14*. ACM Press, 2014, pp. 295–300. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2538862.2538912

[23] J. Konvalina, S. A. Wileman, and L. J. Stephens, "Math proficiency: a key to success for computer science students," *Communications of the ACM*, vol. 26, no. 5, pp. 377–382, 5 1983. [Online]. Available: http://portal.acm.org/citation.cfm?doid=69586.358140

[24] M. B. Rosson, J. M. Carroll, and H. Sinha, "Orientation of Undergraduates Toward Careers in the Computer and Information Sciences," *ACM Transactions on Computing Education*, vol. 11, no. 3, pp. 1–23, 10 2011. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2037276.2037278

[25] A. Lishinski, A. Yadav, R. Enbody, and J. Good, "The Influence of Problem Solving Abilities on Students' Performance on Different Assessment Tasks in CS1," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16*. ACM Press, 2016, pp. 329–334. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2839509.2844596

[26] A. Lishinski, A. Yadav, J. Good, and R. Enbody, "Learning to Program: Gender Differences and Interactive Effects of Students' Motivation, Goals, and Self-Efficacy on Performance," in *Proceedings of the 2016 ACM Conference on International Computing Education Research - ICER '16*. ACM Press, 2016, pp. 211–220. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2960310.2960329

[27] C. Schulte and M. Knobelsdorf, "Attitudes towards computer science-computing experiences as a starting point and barrier to computer science," in *Proceedings of the third international workshop on Computing education research - ICER '07*, 2007, p. 27.

[28] A. Ehlert and C. Schulte, "Comparison of OOP first and OOP later," in *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education - ITiCSE '10*, 2010, p. 108.

[29] D. Zingaro and Daniel, "Examining Interest and Grades in Computer Science 1," *ACM Transactions on Computing Education*, vol. 15, no. 3, pp. 1–18, 7 2015. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2809889.2802752

[30] V. Ramalingam, D. LaBelle, and S. Wiedenbeck, "Self-efficacy and mental models in learning to program," in *SIGCSE conference on Innovation and technology in computer science education - ITiCSE '04*, vol. 36, no. 3, 2004, p. 171.

[31] U. Nikula, O. Gotel, and J. Kasurinen, "A Motivation Guided Holistic Rehabilitation of the First Programming Course," *ACM Transactions on Computing Education*, vol. 11, no. 4, pp. 1–38, 11 2011. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2048931.2048935

[32] D. Cukierman, "Predicting Success in University First Year Computing Science Courses," in *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '15*. ACM Press, 2015, pp. 248–253.

[33] A. Vihavainen, M. Paksula, and M. Luukkainen, "Extreme apprenticeship method in teaching programming for beginners," in *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11*. ACM Press, 2011, p. 93. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1953163.1953196

[34] C. Ott, A. Robins, and K. Shephard, "Translating Principles of Effective Feedback for Students into the CS1 Context," *ACM Transactions on Computing Education*, vol. 16, no. 1, pp. 1–27, 1 2016. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2883588.2737596

[35] P. Machanick, "Peer assessment for action learning of data structures and algorithms," in *Proceedings of the 7th Australasian conference on Computing education-Volume 42*. Australian Computer Society, Inc., 2005, pp. 73–82.

[36] A. Ehlert and C. Schulte, "Empirical comparison of objects-first and objects-later," in *Proceedings of the fifth international workshop on Computing education research workshop - ICER '09*. ACM Press, 2009, p. 15. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1584322.1584326

[37] S. Horwitz, B. Ryder, M. Sweat, S. H. Rodger, M. Biggers, D. Binkley, C. K. Frantz, D. Gundermann, S. Hambrusch, S. Huss-Lederman, E. Munson, S. Horwitz, S. H. Rodger, M. Biggers, D. Binkley, C. K. Frantz, D. Gundermann, S. Hambrusch, S. Huss-Lederman, E. Munson, B. Ryder, and M. Sweat, "Using peer-led team learning to increase participation and success of under-represented groups in introductory computer science," in *Proceedings of the 40th ACM technical symposium on Computer science education - SIGCSE '09*, vol. 41, no. 1. ACM Press, 2009, p. 163.

[38] T. Newhall, L. Meeden, A. Danner, A. Soni, F. Ruiz, and R. Wicentowski, "A support program for introductory CS courses that improves student performance and retains students from underrepresented groups," in *Proceedings of the 45th ACM technical symposium on Computer science education - SIGCSE '14*. ACM Press, 2014, pp. 433–438. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2538862.2538923

[39] D. Horton and M. Craig, "Drop, Fail, Pass, Continue," in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education - SIGCSE '15*. ACM Press, 2015, pp. 235–240. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2676723.2677273

[40] A. Elnagar and M. Ali, "A modified team-based learning methodology for effective delivery of an introductory programming course," in *Proceedings of the 13th annual conference on Information technology education - SIGITE '12*. ACM Press, 2012, p. 177.

[41] ACM. Curricula recommendations.

[42] J. D. Couger, "Curriculum recommendations for undergraduate programs in information systems," *Commun. ACM*, vol. 16, no. 12, pp. 727–749, Dec. 1973.

[43] J. F. Nunamaker, Jr., J. D. Couger, and G. B. Davis, "Information systems curriculum recommendations for the 80s: Undergraduate and graduate programs," *Commun. ACM*, vol. 25, no. 11, pp. 781–805, Nov. 1982.

[44] G. B. Davis, J. T. Gorgone, J. D. Couger, D. L. Feinstein, and H. E. Longenecker, Jr., "Model curriculum and guidelines for undergraduate degree programs in information systems," Tech. Rep., 1997.

[45] J. T. Gorgone, G. B. Davis, J. S. Valacich, H. Topi, D. L. Feinstein, and H. E. L. Jr., "IS 2002 model curriculum and guidelines for undergraduate degree programs in information systems," *CAIS*, vol. 11, p. 1, 2003.

[46] W. F. Atchison, S. D. Conte, J. W. Hamblen, T. E. Hull, T. A. Keenan, W. B. Kehl, E. J. McCluskey, S. O. Navarro, W. C. Rheinboldt, E. J. Schweppe, W. Viavant, and D. M. Young, Jr., "Curriculum 68: Recommendations for academic programs in computer science: A report of the acm curriculum committee on computer science," *Commun. ACM*, vol. 11, no. 3, pp. 151–197, Mar. 1968.

[47] R. H. Austing, B. H. Barnes, D. T. Bonnette, G. L. Engel, and G. Stokes, "Curriculum '78: Recommendations for the undergraduate program in computer science - a report of the acm curriculum committee on computer science," *Commun. ACM*, vol. 22, no. 3, pp. 147–166, Mar. 1979.

[48] "Computing curricula 1991," *Commun. ACM*, vol. 34, no. 6, pp. 68–84, Jun. 1991.

[49] "Computing curricula 2001," *J. Educ. Resour. Comput.*, vol. 1, no. 3es, Sep. 2001.

[50] "Computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science," *Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society*, 2013, 999133.

[51] D. F. Butcher and W. A. Muth, "Predicting performance in an introductory computer science course," *Commun. ACM*, vol. 28, no. 3, pp. 263–268, Mar. 1985.