2023 Annual Conference & Exposition
Baltimore Convention Center, MD | June 25 - 28, 2023
The Harbor of Engineering Education for 130 Years
ASEE
Paper ID #36974

Creating a Blueprint for Success in First-Year Computing

Prof. Frank Kreimendahl, Wentworth Institute of Technology

Frank Kreimendahl is an assistant professor in the School of Computing and Data Science at Wentworth Institute of Technology. He is focused on teaching computer science fundamentals and building stronger resources for student learning. He aims to bring interest and competence to algorithm-driven problem solving in the classroom.

Durga Suresh-Menon

Creating a Blueprint for Success in First-Year Computing

Frank Kreimendahl, Micah Schuster, Durga Suresh-Menon
School of Computing and Data Science
Wentworth Institute of Technology
Boston, MA, USA
kreimendahlf@wit.edu, schusterm@wit.edu, sureshmenond@wit.edu

***Abstract*—First-year computing is the most important pathway to undergraduate degrees and student success in computing, data science, engineering, and related fields. While there is an abundance of research in the area, we still lack retention of students in first-year computing. This is due to many factors including, but not limited to, 1) student math preparation and readiness from high school, 2) adjunct faculty teaching classes where no common coursework is employed, 3) faculty teaching the same courses in different ways, creating inconsistency in material covered, and 4) lack of tutoring for students who are struggling, behind, or do not understand the material.**

This paper presents task force results from a collaboration between faculty and academic support specialists. The paper focuses on two high-impact areas of improvement: standardizing curricula and building support scaffolding outside the classroom. The results, a comprehensive course blueprint, include core resources for a first-semester computing course and recommendations for identification and support of struggling students.

***Index Terms*—computing, first-year, blueprints, CS I, CS II, student success**

I. Introduction

A. Who is Wentworth

Wentworth Institute of Technology is a 39-acre urban campus in the city of Boston where the major programs are engineering, computing, architecture, design, and management.

Mission Statement: Wentworth, the university of opportunity, provides our diverse community of learners with access to educational programs responsive to evolving market needs. Through a uniquely effective, hands-on, experiential, and cooperative education approach, Wentworth prepares graduates who are future-focused and career-ready.

Vision: Placing the student at the center of what we do, our vibrant and diverse campus community of faculty, staff, and students helps each member reach their greatest potential. We seek to maximize the value of our graduates' contributions to global society and their effectiveness as future leaders. [6]

B. Size and Programs

Wentworth has approximately 4000 students who are predominantly pursuing undergraduate majors. The University has five schools: Architecture and Design, Computing and Data Science, Engineering, Management, and Sciences and Humanities. It offers 21 undergraduate degree programs and 11 graduate degree programs. Approximately 300 first-year students take CS I and CS II each year and come from majors in computing, engineering, and management.

C. What are CS I and II at Wentworth

Computer Science I (CS I) is a foundational class that is fundamental to the future success of students in computing majors at Wentworth. Other majors also rely on CS I to introduce students to the basics of programming before branching into major-specific computing courses. CS I covers the fundamental concepts and skills of programming in Java. Students learn and develop skills in problem-solving, algorithm development, program design and structure, code documentation and style, and testing and debugging. Topics include data types and variables, device/file input and output, flow control and functions, use of basic data structures, as well as principles and applications of object-oriented programming.

Computer Science II (CS II) is the second foundational class offered to computing majors at Wentworth. Other majors also rely on CS II to introduce students to the basics of programming before branching into major specific computing courses. CS II covers the fundamental concepts and skills of programming in Java. Students continue to develop skills in problem solving, algorithm development, program design and structure, code documentation and style, and testing and debugging. Topics include object-oriented programming, inheritance and polymorphism, GUI basics with JavaFX, abstract classes and interfaces, generics, collections, recursion, and event-driven programming.

D. Strategic Pillars at Wentworth

Starting in June 2019, our university community undertook an inclusive approach to develop a bold and ambitious strategic plan. The completed plan was unanimously endorsed by the Board of Trustees in February 2020.

The plan established four strategic focus areas – Inclusive Excellence, High Value Learning, a Transformative Student Experience, and Next Generation Partnerships. Within each of these focus areas, the university established goals and related actions to pursue.

Inclusive Excellence: We commit to the continuous development of a campus culture that is increasingly diverse,

equitable, and inclusive. We strive to develop a campus where everyone feels valued and meaningfully supported toward reaching their full potential.

High-Value Learning: We commit to offering a high return on investment by providing an education that equips our graduates with the knowledge that is coupled with a highly valued skill set. We accomplish this through opportunities to address real-world challenges, applied research, social impact projects, cooperative education, collaborative experiences, cross-cultural exchange, and the effective use of advanced technologies.

Transformative Student Experience: We commit to prioritizing the health and well-being of our students and providing opportunities in support of their growth and transformation. Our holistic approach includes an interconnection of high-value learning, with high-quality services, activities, programs, and opportunities that align with student needs, interests, goals, and aspirations.

Next Generation Partnerships: We commit to maximizing the mutual benefits of partnerships with industry, alumni, and various communities. Through these partnerships, we enhance the quality of learning for our students and provide industry with a resource of skilled graduates. Through mutually beneficial forms of engagement, we support the lifelong learning needs of our alumni. We promote economic and social development in communities that include Boston and beyond.

II. Literature Review

The importance of first-year students learning to code has been referenced heavily in literature. The literature referenced below shows the different backgrounds of students who all have one thing in common: the need to learn coding. These works present methods, approaches, and forums that all highlight the need for first-year students, women, at-risk students, and students from all different backgrounds to learn to code.

In approaches like the *guided inquiry approach* [3], the authors argue that a foundational competency in coding is required across engineering disciplines and that this study needs to start in the first year. This paper also talks about the hesitation to code in some engineering disciplines. Foundational computer science concepts help formalize student thinking about both program structure and program design. The method employed here uses guided questioning to reach a solution.

Another area that emphasizes the need for creating successful *first-year experiences for at-risk students* [4], uses an introductory programming language like Alice to create a less intimidating environment than traditional programming languages. While software like Alice is well-documented and provides a user-friendly interface for the students to work with, it does not necessarily provide course consistency between large sections of the courses offered. The objectives of this study was to increase class participation, participation in tutoring sessions, reduce DFW rates, and increase retention.

Another way to increase performance and persistence in first-year CS classes has been through the use of group projects, as discussed by the authors in [5]. In this paper, the authors present an innovative method of using a learning environment coupled with a summative assessment tool. The students not only work in groups, but also present their progress weekly which, in turn, helps with public speaking and presentation skills. All these are very important for a first-year student experience and for introductory programming. This study showed increased attendance rates in the classes and overall course grades increased by 22%.

The author in [7] describes a means to help retain women in CS. This paper reveals that helping female students work in teams increases their sense of belonging in the major and leads to persistence. This paper also supports the claims made by authors in [3].

Perhaps the best support we found for our methodology to employ blueprints came from Cheah in [8]. They performed a literature review of the *Factors Contributing to the Difficulties in Teaching and Learning of Computer Programming* and summarized one of the major factors as *Ineffective Pedagogy*. Their analysis of pedagogy reviewed the sources of ineffective teaching as teaching materials, teaching strategies, and unclear syllabi.

By creating a blueprint, we have consistent course materials and a common syllabus between different sections of each course. The effective pedagogy goal we strive for is driven by course coordination that happens in the CS I and CS II classes, which results in common teaching strategies. Adding to the teaching strategy is the use of embedded tutors in the CS I and CS II classes that provide more scaffolding to the students outside of lectures.

III. Rationale and Methodology

Without a consistent framework for CS I, many students will be unprepared for their subsequent computing courses. This course blueprint, along with the supporting documents, provides the framework for teaching the course in a consistent manner across all sections. The blueprint includes descriptions of the required tools, textbook, topics, assignments, exams, checkpoint questions for faculty and students, and a sample schedule for the course. Instructors at Wentworth use the information contained in the blueprint when forming their teaching plan. The blueprint was completed and first employed in classrooms in Fall 2021.

In this section we will describe the rationale for creating course blueprints and the methodology employed to create them. We used data that was collected from Fall 2016 to Spring 2021. This data shows the performance of our students in the first-year across many different schools.

A. Rationale

The rationale behind this research is three-fold:

1) Increase student retention at our institution
2) Provide a transformative student experience, which is one of our strategic pillars
3) Provide a scaffold to our faculty while providing flexibility of pedagogy

B. Methodology

- Identify when and where students struggled during a semester or across semesters using CS I and CS II grades.
- Identify patterns of success and challenges in the classroom based on semester grades and correlations between grades in different courses.
- Create a template to guide all faculty teaching CS I and CS II to maintain consistency across sections of the same course.
- Provide methods of scaffolding for students who are struggling by providing resources like tutoring, success studio, and peer mentoring.

C. Data

The data collected show students by major in Computer Science, Computer Networking, Applied Mathematics, Cybersecurity, and Engineering. The sample size N consisted of 1830 students from the School of Computing and Data Science, with 1329 being Computer Science majors, 137 Applied Mathematics majors and 361 Computer Networking majors. There were also approximately 500 engineering students from Biomedical Engineering, General Engineering, and Computer Engineering.

TABLE I
CS I AND CS II PASS RATES BY GENDER

Gender	2016	2017	2018	2019	2020	Average
Female	89%	87%	90%	83%	89%	88%
Male	88%	89%	89%	85%	89%	88%

Table I shows the pass rate of the students in CS I and CS II by year and gender. The table shows that there is a high rate of success in these classes, and male and female identifying students perform comparably. It should be noted that in the Fall of 2020, Wentworth offered students the choice of receiving a pass/fail instead of a letter grade. However, the data here show that this did not affect the pass rates. It should also be noted that women make up 21% of the student body at Wentworth, while only 11% of the School of Computing and Data Science students are female identifying.

TABLE II
CS I AND CS II PASS RATES BY RACE/ETHNICITY

Race/Ethnicity	2016	2017	2018	2019	2020	Average
Asian	89%	86%	92%	86%	88%	89%
Black or African American				67%	70%	72%
Hispanic		86%	73%	77%	83%	80%
Nonresident Alien	88%	93%				90%
Other	84%	87%	88%	80%		85%
White	91%	90%	91%	88%	92%	91%
Average	88%	88%	89%	84%	89%	88%

Blank cells indicate sample size is too low.
Blank cell results are included in averages.

Table II shows the pass rates of our students based on race and ethnicity. Wentworth consists of 58% white students and all other races and ethnicities count towards the remaining 42%.

This data shows that white and Asian students are completing CS I and CS II successfully and the students identifying as African American and Hispanic are averaging below 80%. We were encouraged to see that rate of passing increased between 2019 to 2021 for both African American and Hispanic students. With this data, we identified an area where our instruction can improve to better support students of color.

TABLE III
CS I AND CS II PASS RATES BY TRANSFER STATUS

Status	2016	2017	2018	2019	2020	Average
Non-transfer	88%	89%	89%	84%	89%	88%
Transfer	87%	85%	92%	87%	89%	88%
Average	88%	88%	89%	84%	89%	88%

Table III shows the pass rates of transfer students compared to students who started their college education at Wentworth. The cumulative averages are identical, showing no difference in performance between the two categories.

TABLE IV
CS I SUCCESS RATES BY HIGHEST HIGH SCHOOL MATH ACHIEVED

Highest HS Math	Fail	Pass	Withdraw
AP Calculus	6%	93%	1%
Calculus	6%	91%	3%
Pre-Calculus	9%	88%	3%
Other/No Info	9%	88%	3%
Algebra II	22%	71%	7%
Overall	8%	89%	3%

Table IV shows the rates of passing, failure, and withdrawal for CS I students the first time they take the course compared to their high school mathematical achievement. Repeated attempts to take CS I are not included in the data. The table shows that students with a math background only up to Algebra II have a significantly decreased pass rate. This reveals an area in which we can identify and support specific students early.

TABLE V
CS II SUCCESS RATES BY HIGHEST HIGH SCHOOL MATH ACHIEVED

Highest HS Math	Fail	Pass	Withdraw
AP Calculus	6%	90%	5%
Calculus	5%	90%	5%
Pre-Calculus	8%	84%	8%
Other/No Info	9%	85%	6%
Algebra II	7%	90%	3%
Overall	7%	87%	6%

Table V shows the rates of passing, failure, and withdrawal for CS II students the first time they take the course compared to their high school mathematical achievement. Repeated attempts to take CS II are not included in the data. Rather than in table IV, this table shows that CS II success is not strongly impacted by a student's high school mathematics achievement.

TABLE VI

	CS II Outcome		
CS1 Outcome	Fail	Pass, C or lower	Pass, higher Than C
Pass, C or lower	15%	38%	47%
Pass, higher than C	3%	16%	81%

18% of students are required to take CS I but not CS II.

Table VI shows the correlation of student performance between CS I and CS II. 81% of students who performed well in CS I continued to perform well in CS II. Similarly, students who faced challenges in CS I with a grade of C or lower often faced challenges in CS II. For students who passed CS I with a C or lower, 15% of them failed CS II, 38% scored a C or lower, and only 47% passed with higher than a C grade. Improving a student's performance in CS I should have a lasting improvement on their computing education.

Table VII gives a translation of course numbers used in table VIII to course names.

Table VIII shows grade correlations between common courses taken by students in computing majors. CS I and CS II are strongly correlated, whereas CS I and Computer Organization are weakly correlated. All listed courses are taken during a student's first year except for Databases, which is taken during the spring of sophomore year.

We provide some observations about the table as follows:

- High correlations between classes in the same semester, even for classes in different subjects such as CS I and English I
- Positive correlations between all courses taken in the first year
- High correlation between CS and math courses, and CS and engineering courses
- Low correlation between first and second-semester courses, both in CS and engineering
- Very low correlation between Databases and Computer Organization

TABLE VII
COURSE NUMBER TRANSLATIONS

Course #	Name	Semester
COMP 1000	Computer Science I	Fall
COMP 1050	Computer Science II	Fall
COMP 1200	Computer Organization	Spring
COMP 2650	Databases	Spring
ENGL 1100	English I	Fall
ENGL 2200	English II	Spring
ENGR 1800	Programming With Matlab	Fall/Spring
MATH 1750	Engineering Calculus I	Fall
MATH 1850	Engineering Calculus II	Spring

The correlations shown in this table were informative in our approach to designing the blueprint. Most of these correlations were no surprise to us, as weaker correlations appeared between courses that are not prerequisites to one another and courses taken during different semesters.

In section IV we show the blueprint developed based on the data presented in this section. The blueprint consists of all the resources necessary for an instructor to deliver the course, making it easy to share among instructors and also creating institutional knowledge about the coursework in the school.

IV. THE CS I BLUEPRINT

This section describes our CS I blueprint in detail. While the primary purpose of the blueprint is to increase student retention, it has several added benefits for faculty. For new faculty or faculty who have not taught the course at Wentworth before, it clearly shows what materials should be covered. It also gives a framework for coordinators who are making sure that different sections of CS I are covering the same materials.

The blueprint we designed covered all of the details necessary for an instructor to successfully deliver the class:

- ABET assessment questions
- Exams
- Assignments
- Labs
- Lecture slides
- Checkpoint questions for instructors
- Checkpoint questions for students
- Resources for faculty, staff, and students
- Syllabus template
- Sample semester schedule

Tables IX shows brief assignment titles and in-class lab assignment titles used during the semester. The first in-class lab assignment is given in week 6, after the first exam.

A. ABET Assessment Questions

The computer science program is ABET accredited and has built-in assessments in all core courses in computing. CS I and CS II are a part of these core courses.

ABET states specific proficiencies that students must possess at the conclusion of the semester. They are abilities to:

1) Choose the appropriate data type(s) for implementing a given problem.
2) Analyze the behavior of programs involving the fundamental programming constructs.
3) Choose appropriate conditional and iteration constructs for a given programming task.
4) Implement a program that uses fundamental programming constructs.

TABLE VIII
GRADE CORRELATIONS BETWEEN CLASSES FOR CS STUDENTS

	COMP 1000	COMP 1050	COMP 1200	COMP 2650	ENGL 1100	ENGL 2200	ENGR 1800	MATH 1750
COMP 1000								
COMP 1050	0.47							
COMP 1200	0.22	0.42						
COMP 2650	0.21	0.23	0.11					
ENGL 1100	0.42	0.37	0.28	0.23				
ENGL 2200	0.35	0.41	0.33	0.18	0.42			
ENGR 1800	0.55	0.30	0.41	0.49	0.44	0.38		
MATH 1750	0.47	0.40	0.28	0.25	0.46	0.37	0.50	
MATH 1850	0.42	0.48	0.27	0.33	0.36	0.40	0.47	0.58

TABLE IX
WEEKLY PROGRAMMING/LAB ASSIGNMENT TITLES

Assignment #	Name
PA0	Hello World
PA1	Distance Conversion
PA2	Numerical Exercises
PA3	Heron's Formula and Astrology
PA4	Inflation and Guessing Game
PA5	Block Letters and Streaming Average
PA6	Calendar
PA7	No Files
PA8	Files
PA9	Fraction Class
LA1	sum, mean, stdev
LA2	area, perimeter
LA3	norms
LA4	gcd
LA5	fraction arithmetic
LA6	Heron's method

Sample ABET assessment question

Implement the **selfDot** method below, which takes as input an array of integers and returns the sum of the each of the elements squared. An example **main** method, as well as corresponding output, is supplied – you cannot change this method, and the output when the program is run must conform exactly to the sample output.

```
/* sample run:
14
125 */
public static void main(String args[]) {
   int nums1[] = {1, 2, 3};
   int nums2[] = {10, 5};

   System.out.println(selfDot(nums1));
   System.out.println(selfDot(nums2));
}

// put your selfDot method here
```

These outcomes have associated questions/problems that students must be given and assessed on throughout the semester.

B. Exams

The blueprint includes standardized exams for CS I that are shared among all faculty. These exams include some of the ABET assessment questions.

C. Programming Assignments

Complete weekly assignments are included in the blueprints. The Java assignments include skeleton code and unit testing, as well as solutions for the instructor. Assignment specifications are also included.

D. In-class Labs

Starting after the first exam, students have weekly in-class programming labs where they practice writing code to solve small computing problems. These labs offer an opportunity for students to get immediate feedback from instructors, as well as collaborate with their peers.

E. Lecture Slides

An example set of PowerPoint lecture slides are included for instructors. Instructors have creative freedom with their usage of the slides, but they serve a launching point for each instructor to present during lectures.

Table X shows the list of required topics in CS I. Faculty may cover computing information beyond this core set of topics, but it gives the minimum amount that should be covered.

TABLE X
REQUIRED CS I TOPICS

Variables	Conditionals	Arrays
I/O	Expressions	OOP
Types	Testing/Debugging	Exceptions
Strings	Loops	File I/O
Control Flow	Methods	ArrayLists

F. Instructor Resources

The blueprint provides reflective checkpoint questions for instructors to facilitate communication with students. Instructors get a list of questions to interact with students such as "What is working well or not working to help you learn?"

and "Is the pace of this course too slow, just right, or too fast?"

The blueprint also includes a list of resources including contacts for wellness, accessibility, and library services – resources that are outside the scope of the academic material but are still important for student success.

G. Student Resources

Students also receive a list of checkpoint questions to actively reflect on their course progress. They receive a list of university resources to help with tutoring, class registration, study spaces, etc..

The blueprint includes a supplementary programming lab for students to practice running JUnit testing and interpreting the output. There are also plans to build two labs for students to practice using Eclipse's built-in debugger to understand the debugging process.

V. Summary

In this paper, we identified a shortcoming in our first-year computing classes, analyzed the performance of different demographic groups, and built a course blueprint to increase student success.

We summarize that we have built an improved set of teaching and learning tools in the form of a blueprint. Our approach was to build a set of tools that went beyond a standardized set of lectures and assignments. We polled the faculty informally and there is unanimous support for the use of the blueprint. We have created course consistency and this is monitored by the course coordinator of the courses. By having a common syllabus, students in different sections of the course are guaranteed the same material coverage.

We have to further study the efficacy of resources used in the classroom and we plan to summarize that in the future work section, as well as improve the blueprint. We also plan to formalize many of the survey instruments so that we can study the effects of the blueprint in depth and generate data to support the strategic pillars of the institution.

Future Work

This paper and our work has lead us to many interesting questions and further study that we would like to pursue.

A. CS I Group Project

Many of the papers in our literature review talked about the advantage of introducing group projects in CS I. They measured success in retention and persistence rate. We will create one group project for Fall 2023 and study how students respond to it in terms of satisfaction. We will create a survey instrument for an after-project survey.

B. Study/Comparison of Grades

We will perform a study of comparison of grades before and after the blueprint launch. We will also do a year-over-year study of blueprint effectiveness in the Fall of 2023 and 2024.

C. Faculty Satisfaction With Blueprint

We will also survey the first year computing faculty regarding the satisfaction with using the course blueprint. The pedagogy will be presented in a future work.

D. CS II Blueprints Description and Review

We plan to write a paper that describes the details of the CS II blueprint that we have built, which follows a similar pedagogical approach to the CS I blueprint. The CS II blueprint similarly includes faculty and student resources to ensure a consistent and successful approach to teaching the course.

E. Second-Year Computing Blueprints

After building a strong framework for first-year computing courses, we plan to build similar blueprints for our second year of core computer science courses. These include two courses: Data Structures and Algorithms. We hope to extend our students' increased success through their second year of study.

F. Embedded Tutors

We use embedded tutors in all first-year CS courses but have never studied their effect on the CS classroom, especially after the blueprint has been developed. We will study the effects by looking at 1) student retention in the class, 2) student attendance in tutoring sessions and 3) DFW rates in the class.

Acknowledgment

We would like to thank Steven Sherrin and Lisa Keating for collecting the data that was essential for our analysis of student demographics and performance.

References

[1] Rasala, R. Toolkits in First Year Computer Science: A Pedagogical Imperative. *SIGCSE Bull.*. **32**, 185-191 (2000,3)

[2] Goold, A. & Rimmer, R. Indicators of performance in first-year computing. *Proceedings 23rd Australasian Computer Science Conference. ACSC 2000 (Cat. No.PR00518)*. pp. 74-80 (2000)

[3] Bettin, B., Jarvie-Eggart, M., Steelman, K. & Wallace, C. Preparing First-Year Engineering Students to Think About Code: A Guided Inquiry Approach. *IEEE Transactions On Education*. **PP** pp. 1-11 (2022,1)

[4] Armstrong, A. Successful First-Year Experience for At-Risk Students. *Proceedings Of The 2017 ACM SIGCSE Technical Symposium On Computer Science Education*. pp. 45-50 (2017)

[5] Billings, S. & England, M. First Year Computer Science Projects at Coventry University: Activity-Led Integrative Team Projects with Continuous Assessment. *Proceedings Of The 4th Conference On Computing Education Practice*. (2020)

[6] WIT Mission, Vision, and Values. (2023), https://wit.edu/about/president/mission-vision-values

[7] Powell, R. Improving the Persistence of First-Year Undergraduate Women in Computer Science. *SIGCSE Bull.*. **40**, 518-522 (2008,3)

[8] Cheah, C. Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemporary Educational Technology*. **12**, ep272 (2020)