Infusing Artificial Intelligence Concepts into Introductory Computing Courses[*]

Ingrid Russell[1], Abu Saleh Md Tayeen[1], Zdravko Markov[2]
[1]Computing Science Department
University of Hartford
West Hartford, CT, 06117
{irussell,tayeen}@hartford.edu
[2]Computer Science Department
Central Connecticut State University
New Britain, CT 06050
{markovz}@ccsu.edu

Abstract

We present a model to enhance introductory computer science (CS) education by integrating Artificial Intelligence (AI) concepts early in the curriculum. This approach aims to improve student engagement and motivation by (i) connecting computational thinking (e.g., algorithmic reasoning, data representation) to real-world problems through AI, (ii) implementing hands-on projects that illustrate core CS principles and human problem-solving, and (iii) developing an adaptable framework for applying computing concepts across diverse challenges. Building on a prior NSF-funded project on AI-focused, project-based learning, our model scales and adapts AI-themed projects for introductory courses. In this work, we present our model, sample projects, and preliminary experiences using them.

[*]Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

1 Introduction

Artificial Intelligence (AI) has permeated all aspects of our lives and has significantly transformed the way we live and work. It is therefore critical to prepare a talented workforce capable of ensuring the country's competitiveness in this global economy. Computer science graduates are widely in demand, and part of that demand is that they are conversant with the skills and knowledge required to address cutting-edge problems. The recent explosion of AI and the increasing demand for AI professionals, make it imperative that our students be prepared for the workforce. In addition, while recent data from the Department of Labor projects a bright job outlook and a great demand for a computing workforce, many universities are still experiencing significant declines in first-to-second year retention of computer science majors [20].

While computing technology and applications have changed significantly over the last few decades, the most common approach for teaching introductory courses is still programming-focused [19, 12]. Despite efforts to change the perception, students still equate computer science with programming and many do not see its applicability. These disturbing retention figures combined with the need to prepare a workforce conversant with AI skills pose significant challenges and calls for a re-evaluation of our teaching models, particularly in the introductory level courses. The need for transformations in undergraduate computing education has recently been recognized by several studies including some sponsored by NSF [3, 10, 18, 1].

Our project, *Infusing Artificial Intelligence Concepts into Introductory Computing Courses* (**Project InfuseAI**) intends to address the need to revitalize computer science curricula, enhance the learning experience of students in introductory computer science courses, and motivate further study in computing by (1) applying and connecting core Computational Thinking principles such as algorithmic reasoning, data representation, and computational efficiency to real-world challenges, with AI serving as the central theme, (2) implementing a set of hands-on laboratory projects in a wide range of applications using real-world problems which, when solved computationally, will illustrate both human problem-solving behavior and fundamental computer science concepts and (3) developing, applying, and testing an adaptable framework for the presentation of the above core concepts, allowing students to learn and apply core computing concepts to a broad range of societal challenges and opportunities.

While programming is an important skill, the introductory courses are often the first exposure students have to the field, and as such these courses should embrace the fundamentals of computer science. This includes the mathematical and logical background necessary for problem-solving using computers or computation in general. Teaching these fundamentals in computer science is now widely acknowledged as the most important component of teaching introductory computer science courses. The basic ideas behind this approach are nicely articulated "as a way of solving problems, designing systems, and understanding human behavior that draws on concepts fundamental to computer science" [4]. This definition significantly broadens the classical goals of computing education and allows for new ideas that may help improve student experiences in introductory computing courses.

Project InfuseAI builds on a successful NSF-funded project that explored the project-based approach to teaching the introductory artificial intelligence course using the unifying theme of Machine Learning (ML), and is an effort to create a model for revitalizing undergraduate computer science at the introductory computing level. We present work that involves the development of a model for the presentation of core computational topics based on a scaffolded approach to scale down the earlier developed AI-themed projects. Our basic idea is to use interesting application-based real-world problems which, when solved computationally, will illustrate both human problem-solving behavior and fundamental computer science concepts. This approach is not new to computer science education, but it has usually been applied at later stages in the curriculum. In such courses, interesting themes are used to unify the material. These themes are then presented in a project-based teaching environment so that they further enrich the student experiences with practical approaches and tools for problem-solving. Our previous NSF-funded experience is based on such an approach, where we incorporated ML as a unifying theme for introductory Artificial Intelligence courses through hands-on lab projects. As a theme, ML provides methodology and technology to enhance real-world applications.

2 Literature Review

Introducing AI in the introductory computer science courses takes two major approaches. The first one is based on using AI tools and techniques to enhance the student experiences in learning fundamental computer science concepts. Most of the research in this area is related to the recently gaining popularity of Large Language Models (LLM) and Generative AI. In [11], the authors discuss the recent changes in math education that emphasize teaching different representations of math problems that help students better understand the concepts behind them. Applying this approach in introductory computer science means exploring different representations of a programming problem, like higher-level text descriptions, flow charts, UML diagrams, or code. The idea is, instead of writing code to solve a problem given in another representation, to use AI-generated code that students explain or modify in higher-level terms, thus allowing them to better understand the solution and the programming concepts behind it. A similar approach is taken in [8], however more focused on how LLMs work and on finding proper prompts to get the solution and then verifying and explaining it. Students learning programming in CS1 are given a problem and tasked to design proper prompts for the AI system to generate code, which they verify. Thus, effective prompt engineering helps students better understand the solution to the problem and fundamental CS concepts such as software verification. In [13, 14], the authors developed an introductory CS course entirely based on the use of LLMs and other AI tools. It's an online course where AI assists students with learning the course curriculum. These AI tools include LLMs that explain code and evaluate code style and a chatbot for answering course-related inquiries. In [21], the authors developed an introductory programming course called CS1-LLM that integrates AI tools into the course curriculum. Along with the basic CS material, the course teaches students the basic principles of AI tools like GitHub Copilot and LLMs, and how to use them to solve programming problems.

The second approach to integrating AI in the introductory programming courses is based on introducing AI or ML problems into the course curriculum. The authors of [2] suggest an approach to teach CS1 topics in the context of solving AI problems. The exploration of AI is focused on weekly lab assignments and multi-week projects. The labs and projects task students to write programs that use AI and ML algorithms such as decision tree learning, matrix operations on datasets, and neural networks. They are provided with the code implementing these algorithms and incorporate it into their projects using the programming concepts they learn in CS1. In [9], the authors developed the curriculum of an introduction to computing course that may be taught to CS majors and non-majors. The course topics are explored by AI block lessons and a project. The lessons cover Introduction to AI, AI agents, and Introduction to ML, and in the project, students write a program to simulate a rocket landing using some AI techniques such as rule-based agents and genetic programming. For the AI components of their programs, students are provided with code, which they integrate using the basic programming concepts they learn in the introductory programming course.

Most of the existing approaches use AI and ML either to facilitate learning of the basic concepts in CS1/CS2 or to include AI and ML topics directly in the curriculum. Our approach differs in that it emphasizes fundamental computational thinking concepts such as algorithms, data representation, and computational efficiency, which are taught in CS1/CS2, by showing how they can be used to build AI and ML applications. In this way students can better understand these concepts and their importance for real-world computer science applications.

3 Project InfuseAI

Research has shown that participatory or project-based learning methods can level the playing field for different types of students. For example, computer science and engineering have historically been less accessible for female and underrepresented minority students. As a result, these students are underrepresented in most computer science and engineering departments in this country. A shift to a learning environment that values interactivity, cooperation, and collaboration can result in a broader range of students feeling more comfortable and, by extension, can lead to greater persistence and success [22, 15, 7]. Furthermore, studies have shown that the choice of context or problem domain of assignments and examples used in class can have a dramatic impact on student motivation and in turn on the quality of their learning [5, 16]. A problem domain a student relates to and finds relevant leads to deeper understanding and hence a smoother transfer to other domains [6]. Studies have also shown that female undergraduate computer science students tend to be more interested in real applications of computing as opposed to computing for its own sake [22, 23]. Our projects involve machine learning systems and span a wide range of application areas that instructors can choose from.

We build on an earlier NSF-funded project, that explored the project-based approach to teaching introductory artificial intelligence courses using the unifying theme of Machine Learning, a Computational Thinking technique that has been influential in many disciplines, as noted by many [17]. This was a multi-institutional effort that engaged a community of 20 scholars from a broad range of universities working together on the creation of curricular material, and on the development, implementation, and testing of 24 student projects. Each project involved the development of a machine learning system in a specific application. The applications included network security, recommender systems, game playing, intelligent agents, computational chemistry, robotics, conversational systems, cryptography, web document classification, vision, data integration in databases, bio-informatics, pattern recognition, and others.

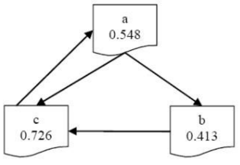
Project InfuseAI focuses on presenting core computational topics through a scaffolded approach. The goal is to adapt and scale down previously developed AI-themed projects for use in introductory computer science courses, specifically CS1 and CS2. Through the design and implementation of systems that enhance commonly deployed applications, our innovative model for teaching introductory computing provides a simple and elegant means to communicate the power of the core ideas of computing in a manner that engages students in experiential education. We believe that these project-based curricular materials will stimulate student interest early in their studies, have a dramatic impact on their motivation, enhance their learning experiences, and motivate further study in computing. Guided by the results of our experiences with the prior work and by the positive results of its assessment, we use a scaffolded approach to scale down the AI projects and adapt a component of each for use in the introductory computer science courses.

3.1 Objectives

The difficulties mentioned earlier associated with the introductory computer science courses, combined with the fact that AI provides a rich set of applications that can be used to stimulate student interest and promote greater participation, are the motivating factors for this project. Our work uses the iterative design-implement-test approach. The specific objectives are: (i) Enhance the student learning experience in the introductory computer science courses by applying and relating fundamental Computational Thinking concepts of algorithmic thinking, data representation, and computational efficiency to real-world problems and to a broad range of challenges and opportunities; (ii) Increase student interest and motivation to study computer science early in their studies by providing a framework for the presentation of core computing concepts that emphasizes the strong connection between computer science and other fields; (iii) Highlight the rich set of applications that can be used to stimulate student interest and promote greater participation, motivating them to pursue further study in computing; (iv) Prepare students for the AI workforce; (v) Assess the effectiveness of our approach in achieving our goals.

3.2 Outcomes

These objectives are accomplished through the development, implementation, and testing of a suite of adaptable and self-contained, hands-on, team-oriented laboratory projects that can be closely integrated into introductory courses that would supplement introductory computer science texts. The specific project outcomes are: (i) A sustainable and adaptable framework that allows students to learn and apply core computing concepts to a broad range of real-world problems; (ii) A lab manual that includes curricular material and supporting resources for the introductory computer science courses; (iii) An instructor's manual that provides a sample syllabus to go with each project and guidance for the adoption and adaptation of this curricular material. The instructor manual will include sample solutions as well as supporting material such as code and documentation; (iv) An assessment report of the effectiveness of the model.



$$A = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \qquad A^T = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

$$\lambda P = A^T P$$

$$\lambda = 1.325$$

$$P = (0.548 \quad 0.414 \quad 0.726)^T$$

$$1.325 \begin{pmatrix} 0.548 \\ 0.414 \\ 0.726 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0.548 \\ 0.414 \\ 0.726 \end{pmatrix}$$

Figure 1: Example graph for power iteration

3.3 Sample projects

In this section, we present two sample projects for CS1/CS2, which are mandatory for computer science and computer engineering majors, and are commonly taken by students from other fields such as science, math, and business.

3.3.1 Social Network Analysis

In this project, students develop an interactive Java program to create and edit graphs. Along with implementing a basic Graphical User Interface (GUI), students must design the internal graph structure and compute node prestige scores using a ranking algorithm. This project reinforces core computer science concepts such as event-driven programming, GUI design, Euclidean geometry (points, distances, segments), and data structures like trees and graphs. Students will also apply key computational math skills, including matrix operations and iterative methods. The prestige score algorithm draws from real-world applications in social network analysis and web search, offering practical relevance and hands-on experience with fundamental programming and algorithmic techniques. Below is a summary of the project given to students.
Project Description: Write a graph drawing program in Java. Use the program discussed in class (Graph.java, Lines.java, Dots.java) and make the following changes and additions: (i) Add an arrowhead at the end of the edges (two short lines starting at the endpoint). You may use ideas and code from Lines.java; (ii) Add two new buttons: Create and Delete. When Create is pressed, allow the user to add nodes and edges by clicking. When Delete is pressed, allow the user to remove nodes and edges by clicking on them. Include a label that clearly shows the current mode (Create or Delete). The third button, “Print adjacency matrix”, should function as in the original version. To detect clicks near nodes or edges, use distance calculations—such as point-to-point and point-to-line distance (refer to Dots.java and Lines.java); (iii) Add

a button that, when pressed, computes and prints the prestige score of the graph nodes. Use the power iteration algorithm described below. *Computing*

Pseudocode 1: Computing prestige score using power iteration

```
1 $P \leftarrow P_0$
2 do
3     $Q \leftarrow P$
4     $P \leftarrow A^T Q$
5     $P \leftarrow \frac{P}{\|P\|}$
6 while $\|P - Q\| > \epsilon$
```

prestige score by power iteration: The Pseudocode for computing prestige score by power iteration is shown in Pseudocode 1. The matrix A is the adjacency matrix that the program creates for the current graph. P and Q are vectors, where each component of P represents the prestige score of the corresponding graph node. The norm used in computing $\|P\|$ and $\|P - Q\|$ is the Euclidean length of the vector and ϵ is a small constant that controls the convergence of the algorithm. P_0 can be initialized with all 1s. Figure 1 shows an example graph and the matrices used in the equations with values obtained by power iteration. You are given a program implementing the power iteration algorithm for the graph shown in Figure 1. To complete the project, you need to extend this program to work not only with 3×3 matrices but also with matrices and vectors of any size.

3.3.2 Non-GUI based Adventure Game

This project provides a practical framework for applying object-oriented programming (OOP) design by modeling entities (e.g., players, enemies) through classes and objects. Students will reinforce core Java concepts, including variables, data types, control structures, and input handling, while also exploring basic AI techniques, such as prediction models, to enhance gameplay. The project thus provides a comprehensive learning experience in Java programming and AI integration.

Project Description: Design and write code in Java to implement a non-GUI-based interactive Adventure game. Some of the key features of the game are the following:

World navigation: The player navigates a fantasy world where random enemy encounters occur. Each encounter starts a simple combat sequence, and the player chooses actions like “attack” to defeat enemies and continue their journey.

Table 1: Survey Results during Spring 2025

The student project contributed to my overall understanding of the material in the course.	86%
The student project was at an appropriate level of difficulty given my knowledge of computer science and programming.	100%
After completing this project, I feel that I have a good understanding of the fundamental concepts covered in this course.	75%
The student project took a reasonable amount of time to complete.	88%
The student project was an effective way to introduce some basic artificial intelligence concepts.	63%
I have a firm grasp of the problem-solving techniques covered in this course.	100%
I would like the opportunity to apply some of the AI problem-solving techniques in the future.	100%
I had a positive learning experience in this course.	88%

Combat System: Both the player and enemies have key attributes such as health, attack power, and defense. The player also has a limited number of health potions to restore health. Combat is turn-based: on each turn, the player can choose to attack or use a potion, while the enemy can attack or defend before attacking. Damage is calculated based on attack and defense values, reducing health accordingly. The game continuously tracks the player's health, potion count, and alive status, while also updating the enemy's stats in real time during combat.
Game Flow: The game begins with the player entering a name and starting their journey through a fantasy world. As they explore, random enemy encounters trigger turn-based combat. Battles continue until either the player's or the enemy's health drops to zero. The player can attack or use a health potion to recover. The game ends when the player's health reaches zero or they choose to stop their journey. Incorporate the following features into the adventure game using AI to enhance the enemy's capability.
Player's Move Prediction: The game will employ a simple classification model, such as Naive Bayes, to predict the player's next action—whether to attack or use health potions to heal—based on the player's previous decisions. By analyzing patterns in the player's behavior, such as frequent attacks or healing, the enemy will adapt its strategy accordingly. For example, if the player consistently attacks, the enemy might predict this and choose to defend first and counterattack in response. If the enemy predicts that the player will use health potions, then it will attack the player with maximum power.

4 Evaluation Plan

A preliminary evaluation of the two projects was conducted in the introductory courses, yielding positive results. Each of the projects has been implemented in CS1 or CS2 in at least one of the two. Most recently, and in order to evaluate students' reception to our approach and to evaluate its effectiveness, we designed a short survey that was given in CS1 at the end of the spring 2025 semester. The survey includes several Likert-scale questions along with two qualitative questions. The Spring 2025 CS1 class consisted of 11 students, with a 73% participation rate.

Below is a summary of the most recent survey results, conducted during spring 2025 using the Interactive Game project. The second column represents the percentage of students who agreed or strongly agreed. Overall, student responses were positive, as can be seen in Table 1. Similar results were reported in prior years [anonymous]. Included in the short survey are two open-ended questions allowing students to provide feedback:

Q1: Describe what you liked best about the student project.

Q2: Describe what you liked least about the student project.

Students enjoyed the hands-on approach. They commented that the project was fun and a fair assignment based on what they had learned in the course. They enjoyed building the different parts of it. A student stated that while they struggled a little with some parts of the project, it was fun to complete. They enjoyed the opportunity to customize. The responses to Q2 centered around wishing to do more with AI and create more of the classes themselves. Based on the survey responses and informal interactions in class, the feedback was very encouraging. While the sample data so far is small to draw significant conclusions, given these preliminary positive experiences and the feedback we received, the proposed project will allow us to develop more such programming projects and do a more comprehensive assessment at both institutions.

5 Conclusion

Integrating AI concepts into introductory computing courses offers a transformative path to addressing key challenges in CS education. By expanding beyond a programming-centric curriculum, Project InfuseAI aims to boost retention, broaden participation, and prepare students for an evolving tech landscape. Through real-world applications and a scaffolded, project-based approach, the initiative fosters engagement and deeper understanding of core computational concepts. Emphasizing computational thinking, problem-solving, and real-world applications supports the rising demand for AI-skilled professionals. By building on proven approaches and adapting them to introductory courses, this project offers a sustainable model for revitalizing undergraduate computing education. It aims to equip students for future challenges while fostering a more diverse and engaged computer science workforce, strengthening the field's global competitiveness.

References

[1] Sherif G Aly et al. “Computer Science Curricula 2023 (CS2023): Rising to the Challenges of Change in AI, Security, and Society”. In: *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 2.* 2024, pp. 852–853.

[2] Steven Bogaerts. “AI and Parallelism in CS1: Experiences and Analysis”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 37.13 (July 2024).

[3] Scott P Chow, Tanay Komarlu, and Phillip T Conrad. “Teaching testing with modern technology stacks in Undergraduate Software Engineering Courses”. In: *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1.* 2021, pp. 241–247.

[4] Center for Computational Thinking at Carnegie Mellon. *Computational Thinking.* https://www.cs.cmu.edu/ CompThink/index.html. 2012.

[5] Joan Dabrowski and Tanji Reed Marshall. “Motivation and Engagement in Student Assignments: The Role of Choice and Relevancy. Equity in Motion.” In: *Education Trust* (2018).

[6] Raymond A Dixon and Ryan A Brown. “Transfer of Learning: Connecting Concepts during Problem Solving.” In: *Journal of Technology Education* 24.1 (2012), pp. 2–17.

[7] Eric Eaton and Susan L Epstein. “Artificial Intelligence in the CS2023 Undergraduate Computer Science Curriculum: Rationale and Challenges”. In: *Proceedings of the AAAI Conference on Artificial Intelligence.* Vol. 38. 21. 2024, pp. 23078–23083.

[8] Amanda S. Fernandez and Kimberly A. Cornell. “CS1 with a Side of AI: Teaching Software Verification for Secure Code in the Era of Generative AI”. In: *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1.* SIGCSE 2024. Portland, OR, USA: Association for Computing Machinery, 2024, pp. 345–351. ISBN: 9798400704239. DOI: `10.1145/3626252.3630817`. URL: `https://doi.org/10.1145/3626252.3630817`.

[9] Adrian A. de Freitas and Troy B. Weingart. “I’m Going to Learn What?!? Teaching Artificial Intelligence to Freshmen in an Introductory Computer Science Course”. In: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education.* SIGCSE ’21. Virtual Event, USA: Association for Computing Machinery, 2021, pp. 198–204. ISBN: 9781450380621. DOI: `10.1145/3408877.3432530`. URL: `https://doi.org/10.1145/3408877.3432530`.

[10] Christopher Hundhausen, Phillip Conrad, and Olusola Adesope. “Designing and Assessing Authentic Software Development Projects in Undergraduate Computing Education”. In: *Proceedings of the 2022 Conference on United Kingdom & Ireland Computing Education Research*. 2022, pp. 1–2.

[11] Lorraine Jacques. “Teaching CS-101 at the Dawn of ChatGPT”. In: *ACM Inroads* 14.2 (May 2023), pp. 40–46. ISSN: 2153-2184. DOI: `10.1145/3595634`. URL: `https://doi.org/10.1145/3595634`.

[12] Amruth N Kumar and Rajendra K Raj. “Computer Science Curricula 2023 (CS2023): The Final Report”. In: *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 2*. 2024, pp. 1867–1868.

[13] Rongxin Liu et al. “Teaching CS50 with AI: Leveraging Generative Artificial Intelligence in Computer Science Education”. In: *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. SIGCSE 2024. Portland, OR, USA: Association for Computing Machinery, 2024, pp. 750–756. ISBN: 9798400704239. DOI: `10.1145/3626252.3630938`. URL: `https://doi.org/10.1145/3626252.3630938`.

[14] Rongxin Liu et al. “Teaching with AI (GPT)”. In: SIGCSE 2024. ACM, 2024, p. 1902.

[15] Chad N Loes. “The Effect of Collaborative Learning on Academic Motivation.” In: *Teaching & Learning Inquiry* 10 (2022).

[16] Ellie Lovellette, Dennis J Bouvier, and John Matta. “Contextualization, Authenticity, and the Problem Description Effect”. In: *ACM Transactions on Computing Education* 24.2 (2024), pp. 1–32.

[17] Tom Michael Mitchell. *The discipline of machine learning*. Vol. 9. Carnegie Mellon University, School of Computer Science, Machine Learning ..., 2006.

[18] Rajendra K Raj et al. “AI, Security, and Society: Lessons From CS2023 For Information Technology Education”. In: *Proceedings of the 25th Annual Conference on Information Technology Education*. 2024, pp. 123–124.

[19] Susan Reiser and Jeff Lait. “What’s New for SIGGRAPH Educators in CS2023: Undergraduate Computer Science Curricular Guidelines”. In: *SIGGRAPH Asia 2024 Educator’s Forum*. 2024, pp. 1–2.

[20] Chris Stephenson et al. *Retention in computer science undergraduate programs in the us: Data challenges and promising interventions*. ACM, 2018.

[21] Annapurna Vadaparty et al. “CS1-LLM: Integrating LLMs into CS1 Instruction”. In: *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*. ITiCSE 2024. Milan, Italy: Association for Computing Machinery, 2024, pp. 297–303. ISBN: 9798400706004. DOI: `10.1145/3649217.3653584`. URL: `https://doi.org/10.1145/3649217.3653584`.

[22] Timothy J Weston, Wendy M Dubow, and Alexis Kaminsky. “Predicting women’s persistence in computer science-and technology-related majors from high school to college”. In: *ACM Transactions on Computing Education (TOCE)* 20.1 (2019), pp. 1–16.

[23] Jue Wu and David H Uttal. “Diversifying computer science: An examination of the potential influences of women-in-computing groups”. In: *Science Education* 108.3 (2024), pp. 957–980.