SN Computer Science (2024) 5:720
https://doi.org/10.1007/s42979-024-03074-y

ORIGINAL RESEARCH

AI-Lab: A Framework for Introducing Generative Artificial Intelligence Tools in Computer Programming Courses

Ethan Dickey[1] · **Andres Bejarano**[1] · **Chirayu Garg**[1]

Received: 8 March 2024 / Accepted: 20 June 2024

Abstract

Purpose Over the last year, the ascent of Generative AI (GenAI) has raised concerns about its impact on core skill development, such as problem-solving and algorithmic thinking, in Computer Science students. With the proliferation of these tools, educators must evaluate their role in academia, as neglecting this might culminate in a phenomenon we term the "Junior-Year Wall," where students struggle in advanced courses due to prior over-dependence on GenAI. Our research seeks to answer the question: *"How can educators guide students' interactions with GenAI to preserve core skill development during their foundational academic years?"*

Methods We introduce "AI-Lab," a pedagogical framework for guiding students in effectively leveraging GenAI within core collegiate programming courses. This framework accentuates GenAI's benefits and potential as a pedagogical instrument. Specifically, AI-Lab presents opportunities to use GenAI for tailored support, such as topic introductions, detailed examples, corner case identification, rephrased explanations, and debugging assistance. Through identifying and rectifying GenAI's errors, students enrich their learning process. Additionally, AI-Lab offers strategies for formulating prompts to elicit high-quality GenAI responses and provides mechanisms for educators to explore students' perceptions of GenAI's role in their learning experience.

Results Preliminary anonymous surveys show that at least 54.5% of our students use GenAI for homework. Notably, the framework highlights the risks of GenAI over-dependence as well as introducing its context-specific benefits, aiming to motivate students intrinsically towards balanced usage.

Conclusions The AI-Lab framework underscores the importance of guiding students' interactions with GenAI to maintain core skill development in Computer Science. By fostering an environment where GenAI is used as a pedagogical tool, educators can mitigate the risks associated with over-dependence on GenAI. This approach is premised on the idea that mere warnings of GenAI's potential failures may be misconstrued as instructional shortcomings rather than genuine tool limitations, thus providing a balanced pathway for integrating GenAI into academia.

Keywords Generative AI (GenAI) · Core skill development · Junior-Year Wall · Pedagogical framework · AI-Lab · ChatGPT

Introduction

In recent years, the pervasive presence of GenAI tools across the internet has transformed the educational landscape. Integrating AI tools into educational environments has garnered substantial attention, yielding invaluable insights and recommendations for its implementation [1–4]. Nonetheless, the rapid emergence of prominent GenAI tools, including GitHub Copilot[1] in June 2021 and ChatGPT 3.5[2] in November 2022, has prompted educators from diverse disciplines to

Ethan Dickey and Andres Bejarano have contributed equally to this work.

✉ Andres Bejarano
abejara@purdue.edu

Ethan Dickey
dickeye@purdue.edu

Chirayu Garg
garg104@purdue.edu

[1] Department of Computer Science, Purdue University, 305 N. University Street, West Lafayette, IA 47907, USA

[1] https://github.com/features/copilot.

[2] https://chat.openai.com/.

Published online: 24 July 2024

raise apprehensions regarding students' utilization of these tools for academic advancement [5–7].

Since their emergence, GenAI tools have captured the attention of educators within the realm of computer science. Notably, Puryear and Sprint [8] delved into the impact of *Artificial Intelligence-Driven Development Environments (AIDEs)*, exemplified by tools like GitHub Copilot, within introductory Computer Science and Data Science courses. The authors undertook a comprehensive examination, revealing that Copilot could generate solutions for assignments across these courses, achieving scores ranging from 68% to 95%. However, the authors raised an overarching concern revolving around low code similarity scores when assessing potential plagiarism using tools such as MOSS.[3] Their insights culminated in an encouraging directive, advocating for *"computer science educators to become familiar with how AIDEs work and begin to design their coursework and development workflow to incorporate them"*.

Wermelinger's investigation [9] further contributed to the ongoing dialogue regarding the impact of GenAI on computing education. The study evaluated the capabilities of GitHub Copilot compared to OpenAI Codex's[4] Davinci model[5] for addressing CS1 coursework challenges. Although Copilot's performance did not match the reported achievements of Davinci [10], it *"does generate code (and with some editing, tests, and explanations) that could have been written by humans."* In their concluding remarks, Wermelinger extends an invitation to educators, encouraging them to guide students in comprehending such tools' potential advantages and inherent limitations.

Similarly, Moradi et al. have contributed a parallel perspective by investigating Copilot's utility within the Software Engineering domain. Their study encompassed an evaluation of the precision of generated code concerning fundamental algorithms and data structures. They observed that Copilot faced challenges integrating multiple methods into a cohesive, functional solution. The authors further contended that *"an expert developer is still required to detect and filter its buggy or non-optimal solutions,"* underlining the ongoing necessity for human expertise despite the advancements offered by Copilot [11].

Since November 2022, ChatGPT 3.5 has emerged as the centerpiece of concern for educators grappling with GenAI tools. Its performance trajectory prompted the developers of OpenAI Codex to deprecate it surprisingly quickly.[6] Within a few months of the ChatGPT 3.5 release, instructors overseeing computer programming courses noted its proficiency in accurately solving numerous problems found within core Computer Science curricula [6, 12–21]. Nevertheless, the tool's effectiveness dwindles as the coursework assumes more intricate dimensions, often resulting in inaccuracy and hallucinations. To mitigate these limitations, users can leverage fundamental prompt engineering techniques [21, 22].

Evidently, the pace of progress in the tech industry outpaces academia. The release of ChatGPT 4 in March 2023 was highlighted by the remarkable improvements over its predecessor [23]. As expected, ChatGPT 4 garnered immediate attention from educators. They promptly raised concerns about its impressive capabilities and advocated reevaluating assessment paradigms to accommodate these evolving tools effectively [24]. With the public release of the upgraded ChatGPT, educators now face a dilemma: should they actively promote student use of these tools or discourage their adoption? [18].

The disruption brought about by GenAI tools has resonated deeply within the realm of Software Development. A simple search for *"coding"* on the webpage *There is an AI for that*[7] by March 2024 yields around 60 AI tools tailored for coding-related tasks, encompassing aspects like generation, completion, quality enhancement, productivity augmentation, and documentation. Among these, certain tools specifically designed for coding education have shown promise in enhancing the learning experience [25]. A subset of the tools is freely accessible to the public, seamlessly integrating with conventional *Integrated Development Environments (IDEs)*. As we have underscored earlier, the enduring presence of these tools is indisputable, with current students likely to encounter their widespread application within the tech industry. Mindful of this level of accessibility and the evolving academic landscape in computing education, we have embraced these tools within our courses. Our approach entails equipping students with the prowess to harness these resources effectively while nurturing their foundational skills—such as problem-solving, algorithmic thinking, and debugging— essential for holistic professional growth.

To address this objective, we pose the fundamental question: **"How can educators guide students' interactions with GenAI to preserve core skill development during their foundational academic years?"** Existing guidance within this domain tends to adopt a broader academic perspective, focusing on classroom utilization. Strategies involve empowering students to critically assess the accuracy and efficacy of GenAI outcomes alongside fostering dialogues about ethical considerations [26–28]. In the context of computing education, Bull et al. propose a pedagogical

[3] https://theory.stanford.edu/~aiken/moss/.

[4] https://openai.com/blog/openai-codex.

[5] https://help.openai.com/en/articles/6195637-getting-started-with-codex.

[6] https://platform.openai.com/docs/guides/code.

[7] https://theresanaiforthat.com/.

strategy that acquaints students with how professional software developers harness GenAI within the tech industry. While emphasizing the imperative for students to emerge as adept problem-solvers by graduation, this approach underscores the utilization of scaffolding and fading techniques, iterative assignment modifications, and a gradual immersion trajectory [29]. Remarkably, up until the submission date of this paper, we have encountered no preexisting framework specifically aiding educators in incorporating GenAI tools within computer programming courses. A conspicuous gap among the cited frameworks is the need for delineated instructor steps for effectively integrating students into using GenAI tools while concurrently fostering the development of their core skill set.

This paper introduces the *AI-Lab* framework, designed to seamlessly incorporate GenAI tools into computer programming courses. The framework functions as a structured environment where students can comprehensively understand the advantages and limitations inherent in GenAI utilization. Simultaneously, it furnishes instructors with valuable insights into students' viewpoints regarding using these tools within the present and upcoming courses. AI-Lab features prompt guidelines designed to optimize usage and facilitate effective communication. Additionally, the framework offers an interactive lecture dynamic, empowering instructors to engage students in substantive discussions surrounding course content while illuminating the boundaries and constraints of GenAI.

Note that while much of this framework uses language referring to ChatGPT, we found that both this framework and topical issues we found were present in the 2023 versions of other models such as Bing Copilot, Llama, Claude, and Gemini.

The Junior-Year Wall Problem

In the evolving landscape of Computer Science education, a nuanced challenge emerges as students transition into their junior year, which we have termed the *"Junior-Year Wall."* This concept encapsulates the predicament faced by students who, during their foundational courses, have become accustomed to relying heavily on GenAI tools. As they advance, they encounter a stark increase in the complexity of topics and a significant reduction in the efficacy of GenAI to aid their learning. This transition exposes a critical gap in their foundational knowledge and skills–such as problem-solving, algorithmic thinking, and coding proficiency–essential for mastering advanced courses. The reliance on GenAI in earlier stages of their education may inadvertently foster a surface-level understanding of core concepts, inadequately preparing them for the heightened analytical and independent problem-solving demands of junior and senior-level courses.

Evidence suggests that this over-reliance on technology can lead to underdeveloped critical thinking and learning skills, as seen in academic performance trends and faculty observations [30–32]. Specifically, consideration for possible-over reliance, emphasis of the need to read and review code created by the knowledge that outputted code could be inaccurate, and the need for AI literacy training are brought up by [10, 33–36]. This shift in educational dynamics raises concerns about the long-term implications for students' competencies, potentially impacting their readiness for the professional world and their ability to contribute to innovation in the field. It also prompts a reevaluation of pedagogical strategies to ensure that while GenAI tools are integrated into the learning environment, they complement rather than supplant the development of indispensable computer science competencies [32]. Thus, the *Junior-Year Wall* highlights a pivotal educational challenge and underscores the imperative for a balanced approach to incorporating emerging technologies in STEM education, ensuring that students fully comprehend their discipline rather than having a merely superficial understanding.

In the discourse surrounding the *Junior-Year Wall* within Computer Science education, it is pertinent to acknowledge the developing stage of GenAI tool integration and the consequent lack of empirical evidence directly linking these tools to educational outcomes. However, we propose this discussion based on anecdotal evidence, which, albeit preliminary, suggests significant educational implications. We have observed instances where students utilizing GenAI tools submit assignments with code directly copied from such platforms. When challenged to explain their submitted solutions, these students often lack an understanding of the fundamental concepts and logic underpinning their submissions. This inability to articulate the rationale behind their work indicates a failure to meet the course's learning objectives, emphasizing a superficial engagement with the material. Such observations contribute to the conceptual framing of the Junior-Year Wall, highlighting the need for a thoughtful approach to using GenAI in educational settings. From this, we draw our inspiration for introducing the AI-Lab intervention before students' junior year.

The Role and Implications of GenAI Tools: A Rational Exploration

GenAI tools have undeniably revolutionized the way students approach foundational Computer Science courses. While there is no concrete evidence as of this submission date directly linking the use of these tools to the *Junior-Year Wall* problem, it is rationally conceivable. If students navigate their initial courses heavily aided by GenAI, there's a plausible risk that they might not fully internalize the content. As they advance to more complex subjects, where the

efficacy of GenAI tools becomes significantly reduced, they could be confronted with significant challenges stemming from gaps in foundational knowledge acquired in earlier courses. We do, however, want to mention that when we have caught students cheating using GenAI, they very frequently cannot explain their code nor the concepts involved, indicating they have not met the learning objectives for that assignment. This provides some anecdotal evidence for the eventual failure of such a student who is not meeting the learning objectives as we believe they are as they progress through their major. Furthermore, it is pedagogically unacceptable for us to sit by and wait for the phenomenon to happen with direct evidence (as students will have to fail to observe it), which is why we take this proactive approach.

Proactive Pedagogy: Preparing for Potential Challenges

As we navigate the potential challenges posed by GenAI tools, it's imperative to consider how we can proactively integrate these tools into the curriculum without compromising the depth and quality of student learning. Our discourse is not a critique of GenAI tools. In fact, when judiciously incorporated into the curriculum, these tools can provide invaluable support, enabling students to address coding errors, gain diverse perspectives on concepts, and engage in hands-on practice. Such integration can reflect the real-world dynamics of software developers interfacing with AI tools in the industry.

Our stance is informed by two primary considerations:

1. Educating students on the adept use of GenAI tools, while emphasizing the importance of mastering foundational concepts and illuminating their limitations, can serve as a robust deterrent against potential over-reliance. This proactive approach resonates with established principles of motivation theory [37–39].
2. By anticipating potential educational challenges, we can tailor our pedagogical strategies to ensure comprehensive learning experiences, championing principles of equity and fairness.

Conclusion and Forward Momentum

While not empirically validated, the *Junior-Year Wall* Problem represents a rationally anticipated challenge in Computer Science education. Our role as educators is not merely to react to challenges but to anticipate and prepare for them. From our recent experience, we have interacted with students who submitted results generated using AI yet could not explain them in their own terms (thus, they are not meeting the learning objectives of the assignment, pointing to over-reliance). By recognizing the potential implications of the evolving educational landscape, we can ensure that our students are well-equipped with tools and fortified with the critical thinking and foundational skills essential for their academic journey.

AI-Lab Framework

We introduce the AI-Lab framework as an integral course activity designed to educate students on using GenAI tools within their educational journey. The primary objectives of the AI-Lab are:

1. To introduce students to a meaningful and beneficial usage of GenAI in the educational setting, especially in its potential to enhance educational outcomes.
2. To highlight the fallibility of AI, specifically its potential to produce incorrect outcomes confidently.
3. To emphasize the crucial link between robust skill development and the ability to tackle advanced tasks is beyond the reach of AI assistance.

The lab's implementation spans a range of programming concepts. Initially, students are encouraged to leverage GenAI for addressing questions and enhancing material comprehension. As the lab unfolds, progressively intricate queries are posed, strategically designed to surpass GenAI's problem-solving capabilities and consequently render erroneous responses. Significantly, this phase underscores the imperative of fundamental skill acquisition; any over-reliance on GenAI hampers students' capacity to accomplish tasks beyond its purview. The culmination of the lab incorporates a concise discourse wherein students reflect on their GenAI encounters, fostering critical reflection on AI's limitations and their learning experiences. Figure 1 summarizes all of the steps of the AI-Lab, as outlined in the subsequent subsections.

The effective integration of GenAI tools into computer programming courses can promote equity in the educational landscape. Equity stands as a pivotal catalyst driving our framework's trajectory. By equipping students with the skillset to proficiently navigate GenAI tools, we address the disparity faced by students who lack the privilege of pre-collegiate exposure to such resources. This proactive approach diminishes the ability gap, ensuring students enter higher education more equitably.

It is important to note that the point of the structure of the AI-Lab is that (a) the instructor needs to introduce best practices before experiencing success and failure, (b) the intervention before the post-intervention reflection-type activity, and finally (c) needs to have some sort of introduction to GenAI, in or before the class. As long as this structure and our guidelines are generally followed, the framework is

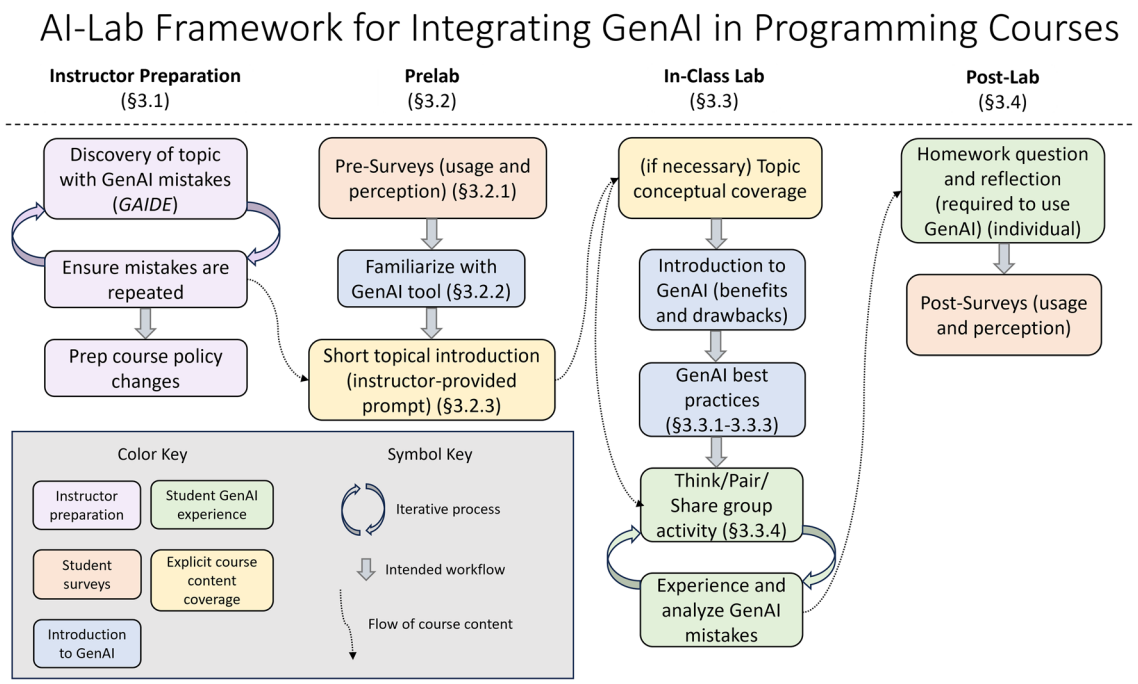

Fig. 1 This figure illustrates the AI-Lab Framework for integrating Generative AI in programming courses, outlined as a structured flowchart. Key phases–Instructor Preparation, Prelab activities, In-Class Lab, and Post-Lab reflections–are further color-coded to indicate different functional purposes. Progression through these phases is shown with solid block arrows, demonstrating the suggested workflow, while circular arrows indicate iterative processes, and dotted arrows indicate points where formal course material integrates into the framework. Please note that while this looks like a rigid structure, an instructor can take the pieces and rearrange them or change them as is necessary for their course, so long as they abide by the foundational principles of each component further discussed in Sect. AI-Lab Framework

flexible for adaptation to specific courses, workshops, lab sessions, or otherwise.

In the appendix section, we provide an example of generated content to run the AI-Lab in a lecture.

Instructor Preparation

The first step is for the instructor to pick a topic from the class suitable to be introduced and discussed using the selected GenAI tool. The topic must be one for which GenAI gives mostly correct conceptual responses but misses correctness, for example, with solutions to problems using non-standard input values. Finding this topic requires time and expertise from the instructor, as multiple iterations may be needed to reach the topic that fits within the requirements of the lab. In our experience, we found that topics requiring multiple non-trivial computational steps of medium difficulty (e.g., Self-Balancing Search Trees and Huffman Coding) are suitable for the AI-Lab. Selecting topics that are of high difficulty will likely result in students not seeing the full potential of GenAI for helping them academically, so a proper balance must be found. An effective method we have found to find such topics is to use the GAIDE Framework [40] for generating course content (lectures, assessments, etc.), during which one is likely to find errors and inconsistencies, along with many examples of success.

We encourage the instructor to review the current course policies to ensure alignment of the usage and expectations of AI in the class. Additionally, the instructor should ensure alignment with the respective institutional or department statements and policies on the matter. That being said, allowing students to use these tools on all assignments should be done with careful assessment of assignments to ensure that students cannot rely completely on these tools for assignment completion, and the instructor needs to find a mechanism for sharing evidence that they used GenAI in the proper way (e.g., sharing chat links to all conversations about assignments).

PreLab

As with any course activity, the instructor holds the crucial responsibility of defining the scope of the AI-Lab. This initial task, conducted offline, necessitates a comprehensive formulation that includes clear objectives, goals, and desired learning outcomes. Additionally, the instructor must diligently select a topic that strategically challenges GenAI's efficacy. A paramount objective of this framework remains,

alongside introducing beneficial use cases, to highlight the pitfalls of unreflective GenAI reliance, which impede skill development. Hence, instructors must meticulously curate a topic that inherently heightens the likelihood of GenAI producing erroneous outcomes. For instance, the realm of coding solutions, well-established for ChatGPT's limitations, provides an illustrative case [41]. Our experience with GenAI tools has revealed that concepts involving pointer manipulation, concurrently managing multiple data structures within a singular algorithm, and asking for illustrative examples of graph problems tend to confound the tool's performance.

The prelab phase consists of three activities that students must complete prior to the ensuing in-person lab session (to be elaborated on in the following subsection). The underlying objective of the prelab phase is to provide students with an initial exposure to the chosen GenAI tool, fostering a sense of familiarity and comfort in its application. The three activities are the following:

First Activity: Perceptions Survey

Students must complete a survey that gauges their perceptions regarding using GenAI tools. The survey seeks to capture students' firsthand experience with GenAI tools and their observations of how their peers use them. This preliminary survey is crucial, furnishing instructors with valuable insights into students' perspectives on GenAI tool utilization within an academic context. We provide the prelab survey questions in an appendix in the full version of the paper.

Second Activity: Familiarity with the Tool

Students must create an account on the chosen GenAI tool and acquaint themselves with it. This process involves providing students with detailed instructions, including illustrative prompts they can directly copy and paste. These example prompts serve as a means for students to engage with the tool's functionality firsthand.

Subsequently, students embark on their tool exploration using self-generated prompts. With chat-based GenAI tools, students can begin with a broad query, such as *"Tell me about yourself,"* and continue the interaction by posing elementary questions. This dynamic mirrors human conversation, allowing students to engage with the tool as if conversing with another human. Other potential prompts are:

- What are some of the places to visit in [your city]?
- Does [your university] actually have anything fun to do?
- What is a good recipe for [some dessert]?
- Help me plan a trip to [a place you want to visit].

Upon establishing proficiency with these introductory inquiries, students are encouraged to transition toward queries that align with their domain of study in computer science. We suggest providing prompts relevant to topics covered in the course curriculum. These prompts serve as a mechanism for students to assess the accuracy and comprehensiveness of the tool's responses. It is imperative to note that these provided prompts are deliberately concise, comprising no more than a single sentence. Students are empowered to pose questions about topics covered in the course, leveraging the tool to revisit, reinforce, or seek further clarification on their learning. Some example prompts are:

- What is Big-O?
- What is a Linked List? Help me visualize a Linked List.
- What is a Left-Leaning Red-Black Tree? Give me an example of a Left-Leaning Red-Black Tree.

Third Activity: Topic Heads-Up

Students will request explanations from the GenAI tool regarding the forthcoming lab session's subject matter. To facilitate this phase, the instructor will give students a prompt to copy and paste into the tool directly. This step affords students a concise preliminary overview of the upcoming lab topic. The prompt's structure adheres to the *"persona prompt pattern,"* [42] a strategy grounded in the observation that GenAI tends to yield enhanced and tailored outcomes when provided with a contextual role to assume.

We propose the following prompt template for students to run in the GenAI tool: *"Act as a [top-level expert in the field you are trying to learn about, with qualifications or titles if necessary]. I want you to introduce me to [topic] briefly. I already know [list all relevant previous knowledge here]. The learning objectives for this section are as follows: [list of learning objectives to be satisfied]."*

We advocate for the active engagement of students in crafting personalized queries aligned with their chosen subject matter. An effective strategy involves leveraging Bloom's Revised Taxonomy [43] to formulate precise learning objectives. Students can articulate these objectives in their vernacular and subsequently use a chat-based tool to rephrase or rearticulate them, employing taxonomy-specific terminology. As an illustrative exercise to unveil these capabilities, students can prompt ChatGPT to rephrase their self-defined learning objectives using the taxonomy. To facilitate this exercise, a designated prompt is suggested: *"Act as a professor of engineering education to rephrase some learning objectives using Bloom's revised taxonomy. The learning objectives are: [List the learning objectives you want]."*

In-Class Lab

In this lab segment, students delve into the diverse applications of GenAI, understanding both its ideal use cases and its limitations. An interactive component is woven into this phase, wherein students undertake the instructive role themselves to acquire mastery over a novel topic. To facilitate this interactive engagement, students must bring their laptops and participate in a synchronized learning experience guided by instructor-led demonstrations.

The class begins with the instructor offering a concise review of the day's content, based on the expectation that students have completed the prelab activities. This introduction sets the stage for the day's main focus: GenAI in education.

The session begins with a brief overview of GenAI, highlighting its core principles and real-world applications. Instructors are encouraged to ask students *"what do you think GenAI is useful for? For example, drafting or rewriting emails."* And then proceed to give them 4 min (approximately) to answer. Encourage all responses, as we have found that as soon as someone says *"drafting my English essays"* or the like, many students begin to relax and contribute in ways that you likely did not realize GenAI was being used for (even if some are slightly concerning, such as using GenAI as a therapist).

This overview is followed by a quick refresher on the basic prompts to ensure everyone is on the same page. The focus then shifts to GenAI's application in Education and Computer Science. In the context of Education, discussions revolve around personalized examples, in-depth explanations, concept simplification, and generating practice questions. The Computer Science segment delves into generating boilerplate code, crafting standard data structures and algorithms, identifying test cases, and aiding debugging. Two key takeaways for students are identifying instances of educational use cases and building confidence in identifying GenAI's mistakes.

In light of these discussions, it becomes imperative for the instructor to guide students in the art of formulating effective prompts. For a detailed discussion on best interacting with these tools, please see the overview in [40]. The discussion could be as follows:

First Step: Specific Context

Each prompt should commence with role and context delineation. These two essential components constitute the prerequisites for eliciting high-quality outcomes, a principle encapsulated in the 'persona prompt pattern' [42]. For instance, consider the prompt: *"Assume the role of a collegiate professor holding a Ph.D. in Computer Science, addressing a sophomore-level class on data structures and algorithms where students possess familiarity with Java."* It is pivotal to consistently specify an expert's role alongside a specified level of proficiency or qualifications. The context element establishes the framework within which the response will be generated, tailored to the educational level of the students. Concluding the prompt by specifying the audience's familiarity with certain concepts enables the tool to utilize appropriate technical terminology and tailor the response to the student's existing knowledge base, thereby enhancing the effectiveness of the interaction.

Second Step: Specific Request

Crafting effective prompts is pivotal for eliciting high-quality outcomes from GenAI interactions. While simple prompts often fall short of yielding optimal results, more elaborate and, at times, quantifiable instructions tend to enhance the quality of GenAI's outputs[44]. For instance, while the prompt *"Give me a set of 10 problems for practicing Huffman encoding."* may not guarantee a flawless set of problems, but it is reasonable to anticipate a subset, typically 2–4 out of 10, will be good quality. It is crucial to acknowledge that perfection is not the norm. Emphasizing specificity in prompts, which often entails integrating multiple directives, can augment the depth and breadth of GenAI's responses. In this example:

- *"give me"* serves as a command, functioning as a specific and directed request aimed at the tool.
- *"a set of 10 problems"* designates a precise numerical value to provide a range of options for selection, facilitating access to a focused collection of tasks (a problem set).
- *"for practicing"* contributes to the overarching objectives. This element tailors the problems to support learning and active engagement with various aspects of the subject matter.
- *"Huffman encoding"* encapsulates explicit terminology that pinpoints the exact subject under scrutiny. It is possible to delve into different levels of specificity, yielding varying degrees of success (typically greater success with heightened specificity). For instance, you could request questions about *"compression trees"* in a broader sense, which would provide more theoretical inquiries. Conversely, soliciting problems concerning *"using frequency analysis to create a Huffman Tree"* would furnish you with a set of scenarios dedicated explicitly to honing this particular skill.

Third Step: Specific Goals

The inclusion of goals within prompts is discretionary. For tasks that are concise and straightforward, the inclusion

of goals may be optional. However, suppose the objective entails formulating an extensive series of questions and examples, generating ideas, or addressing multifaceted tasks. Providing a broad outline of the intended direction is prudent in that case. This practice serves as a navigational aid for the GenAI tool, ensuring it aligns with the overarching purpose of the task at hand.

Experiencing Success and Failure: Group Activity

This module builds on the initial aims of the AI-Lab outlined in the introduction to Sect. AI-Lab Framework, specifically: (1) to introduce students to a meaningful and beneficial usage of GenAI in the educational setting, especially in its potential to enhance educational outcomes, and (2) to highlight the fallibility of AI, specifically its potential to produce incorrect outcomes confidently.

In practice, these objectives are met through a dynamic, in-class interaction where students use the GenAI tool under guidance. This setup not only demonstrates the tool's capabilities but also its limitations, providing a rich basis for discussion and learning. In particular, the instructor can adopt one of two approaches during the interactive class sessions:

1. Direct Interaction: The instructor operates the GenAI tool live, engaging the whole class in deciding the next steps and discussing the outcomes in real-time. This approach fosters a collaborative environment where students can actively participate in steering the direction of the interaction with GenAI, but allows the instructor more control over the discussion and student takeaways.
2. Think/Pair/Share [45]: Students work in small groups to explore GenAI's capabilities and limitations through structured exercises. After working independently or in pairs, students reconvene as a class to discuss their findings and the errors identified, facilitated by the instructor. This approach, while giving less direct control to the instructor, usually provides a broader and more engaging range of experiences for students, allowing them to explore topics they are interested in within the scope of the instructor's selected activities.

For either approach, the activities are designed to be lighthearted and instructive, rather than rigorous. For example, one can provide the following three options for the think/pair/share activity (note that the content selected during this AI-Lab example was (Strongly) Connected Components):

- Tarjan's Algorithm Example: Generate an example of Tarjan's algorithm, review each step, and identify any inaccuracies in the execution.
- Exam Practice Problems on Tarjan's Algorithm: Create practice questions related to Tarjan's algorithm for an exam, along with answers. Critique either the questions or the answers (or both).
- Pseudocode Writing Problems: Using a topic covered this semester, request GenAI create practice problems for writing pseudocode. Critique the response.
  - In particular, this third item can be anything the students have struggled on during the semester. We like to put what students struggled on most during the last exam.

Ultimately, the aim of these activities is not just to demonstrate the tool's capabilities, but to foster a nuanced understanding of GenAI among students. Through guided experimentation and discussion, students will discover unexpected applications of GenAI and learn to navigate its imperfections. These experiences are invaluable as they equip students with the basic skills of prompt engineering and critical assessment, preparing them for more advanced applications in their future academic and professional endeavors. The instructor's firsthand experiences with GenAI tools play a pivotal role in highlighting unique and thought-provoking use cases, enriching the students' learning journey.

Post-Lab

The post-lab segment is a pivotal phase for reinforcing students' comprehension and offering instructors valuable insights into GenAI's integration within academic contexts. This phase encompasses a meticulously designed worksheet tailored to encompass a spectrum of tasks undertaken using GenAI. While the content need not be exclusively Computer Science-oriented, a worksheet section should be dedicated to the domain. This particular section should revolve around the subject matter covered in the class, encompassing activities such as problem-solving, question generation, critical evaluation of GenAI's topical outputs, or similar engagements.

A worksheet of this nature can be relatively straightforward, often comprising a single inquiry pertinent to the topic explored during the in-class lab, to be addressed as part of a subsequent homework assignment. This query should be designed in a two-fold manner: the initial segment prompting students to attempt problem-solving utilizing GenAI and the latter urging them to tackle the same issue via conventional conceptual approaches. From our practical observations, topics that expose GenAI's limitations can engender a pivotal juncture wherein students grapple with its shortcomings and then opt for a traditional problem-solving route. We advocate for instructors to engineer scenarios that tap into GenAI's technical capabilities. These scenarios can encompass diverse facets, such as implementing intricate details in a specific programming language, devising comprehensive test cases,

identifying corner cases, establishing rational boundaries for asymptotic analysis, or seeking real-life applications.

Suppose the instructor opts to assign tasks that are not directly tied to the current topic covered in class. In that case, they can consider providing assignments that pertain to subjects of potential importance or relevance to the course yet may not be fully covered within the confines of traditional lectures. This approach could involve tasks prompting students to independently explore a subject that holds potential benefits but might be excluded from the list of core curriculum topics. For instance, tasks related to web development, unit testing, or other complementary areas could be incorporated to broaden students' skill sets and knowledge horizons.

A significant task related to the covered topic involves having students utilize GenAI to generate problem questions or examples about the subject they have learned:

- Initiate the process by prompting them to devise questions regarding a familiar topic that has recently been taught and is within their comfort zone. Although it is possible to use the just-learned topic, generating practical learning objectives might pose a challenge. Providing learning objectives becomes optional if the topic is relatively straightforward and students have a solid understanding.
- Transition to having students generate questions about the topic introduced in the lab to strengthen their capacity for question generation and evaluation. In this scenario, learning objectives must be provided.
- Encourage students to select two appealing questions and elucidate their reasons for favoring them. Subsequently, instruct them to solve these chosen questions using GenAI and individually. Emphasize the value of employing GenAI after individual problem-solving, as it offers a distinct perspective, ensures topic comprehension, and enables comparison with GenAI-generated responses.
- Direct students to assess GenAI's answers based on the following criteria: technical accuracy, relevance to the posed question, and comprehensiveness of the response in addressing all aspects of the inquiry.
- Depending on the instructor's intended class discussion scope, there is an option to supply 1–5 instructor-generated questions related to the recently taught topic. Students would then solve these questions autonomously using GenAI and analyze the resultant answers. This task aims twofold: reinforce learning through independent application and critically evaluate GenAI's responses.

Concluding the cycle, reintroduce the survey from the initial stage of the prelab. This feedback mechanism aims to gauge the reception of GenAI utilization and provides valuable insights into students' evolving perspectives throughout the learning journey.

Experience and Preliminary Data

We conducted a preliminary AI-LAB experience in the Summer of 2023. We implemented the AI-Lab framework within the context of CS251 - Data Structures and Algorithms, a pivotal undergraduate course offered by the Department of Computer Science at Purdue University. We employed ChatGPT 3.5 as the chosen GenAI tool due to its open availability. Our decision to focus on Huffman Coding as the lab topic was twofold: Firstly, due to its alignment within the course, students should be ensured of a foundational grasp of the requisite underlying data structures. Secondly, Huffman Coding was chosen for its propensity to expose various errors in GenAI outputs during the coding algorithm execution, as evidenced by our experiences with ChatGPT 3.5, while still demonstrating to students the usefulness of GenAI in, for example, giving conceptual explanations for the confusing topic.

Before moving to specifics, the authors want to emphasize explicitly that this was a *preliminary* study, and thus lacks the full analytical weight of a rigorous research study. Our preliminary study sought to pilot the framework and allow us to collect preliminary data which we can use to guide future implementations of the lab in much larger settings (800+ student courses). As such, we do not elaborate on the implementation context, data collection methods, background of participants, etc. beyond what is described above, so as to avoid the illusion that this intends to be a rigorous educational study of the impact of our AI-Lab intervention. This paper only intends to introduce the framework and give some minor evidence that we have tested this framework, along with presenting one statistic from the *pre*-survey that might be startling for readers, as it was for us.

We utilized the subsequent prompt to engage ChatGPT in introducing students to Huffman Coding: *"Act as a computer science professor who has been trained how to teach well. I want you to introduce me to Huffman Coding briefly. I already know String pattern matching and Tries. The learning objectives for this section are as follows: 1. Analyze and illustrate the process by which Huffman coding compresses data. 2. Decompose and evaluate the structure and information presented in a Huffman tree. 3. Compute the bit requirement for a specific set of data using Huffman coding and validate its efficiency."*

In this preliminary experience, we observed a notable attendance of 38 out of 41 students during the in-person session. The session's instructor meticulously navigated through the previously outlined prelab steps, acquainting the students with GenAI, ChatGPT, foundational prompts,

and real-life instances. Progressing into the heart of the session, interactions between students and ChatGPT unfolded, centering on the definitions and significance of Huffman Coding. The instructor adeptly elucidated essential concepts as students engaged with the tool. To exemplify the algorithm's operation, students were prompted to collaborate in pairs, tasking ChatGPT with constructing Huffman Coding Trees for diverse strings. Swiftly, students were compelled to flag ChatGPT's errors–some of which teetered on the realm of absurdity–bringing them to the attention of their partners and instructors. By incorporating the Think/Pair/Share technique [45] on ChatGPT's fallibility, students swiftly engaged in discourse and contrasted outcomes across pairs. Eventually, two groups (owing to logistical constraints) were designated to share their prompts and resultant outputs with the entire class. During this sharing, all groups adeptly discerned the errors, rectified them within the collective setting, and subsequently engaged in reflective discussions regarding the tool's accuracy.

The following is a partial *preliminary* analysis of the pre- and post- surveys conducted during an early iteration of the AI-Lab. The analysis of the anonymous pre-survey data unveiled intriguing insights. Firstly, at least 54.5% of surveyed students were using GenAI tools for homework help. This figure might even be an underestimation, considering potential hesitations in self-reporting. Recent literature and news articles suggest that the actual number of students using GenAI and the variety of their methods might be more extensive than previously assumed by educators [5, 15]. Furthermore, when examining the question about peer usage of GenAI on homework, it was revealed that at least 60% of the surveyed students were aware of fellow students incorporating GenAI tools into their coursework. To elaborate, 2.9% of respondents indicated that 100% of their peers employed GenAI, 17.1% said 75%-99%, 20% said 50%-74%, 8.6% said 25%-49%, 11.4% said 1%-24%, 2.9% said 0%, and 37.1% expressed "uncertainty". Delving deeper into the pre-survey and post-survey findings, a discernible shift in students' perspectives concerning GenAI tool utilization for educational purposes emerged. Regarding the question on perception of GenAI reliability, the pre-survey results per option stood at 0% for 1 (Very unreliable), 41.7% for 2, 44.6% for 3, 11.1% for 4, and 2.8% for 5 (Very reliable). Remarkably, the post-survey outcomes for the same question exhibited alterations: 10.3% for 1 (Very unreliable), 27.6% for 2, 41.4% for 3, 13.8% for 4, and 6.9% for 5 (Very reliable). These variations underscore a perceptual shift in GenAI's reliability. Although derived from a controlled environment within a limited student population, these insights provide a glimpse of achievement in alignment with the proposed framework's objectives.

Discussion and Future Work

The rapid ascent of GenAI tools, mainly since November 2022, has sparked a wave of disruptions within computing academic circles, compelling instructors to recalibrate their strategies, adapt their approaches, or even adopt new stances in response to the evolving landscape [18]. While educators and researchers have anticipated the advent of AI tools in education, the integration of these diverse AI techniques into the intricate fabric of STEM education has posed formidable challenges, as underscored by Xu and Ouyang, who observed that *"the application of AI technology in STEM education is confronted with the challenge of integrating diverse AI techniques in the complex STEM educational system"* [46]. Before the resounding impact of GenAI and its critical reception, AI tools were regarded as relevant but not yet transformative forces in education. As articulated by Zhai et al., *"[Machine Learning] has transformed—but not yet redefined—conventional science assessment practice in terms of fundamental purpose, the nature of the science assessment, and the relevant assessment challenges"* [47].

Nonetheless, drawing from our academic roles and the firsthand experience we have gained with GenAI tools, we posit that GenAI's impact mirrors the transformative shifts witnessed with the advent of accessible platforms such as the Web, Search Engines, and the Internet within households. GenAI appears to be the latest addition to the toolkit available to the *Net Generation* [48], a generation that includes students and instructors, each hailing from the same era of digital ubiquity. The allure of GenAI tools in the contemporary software development landscape is palpable, naturally enticing students to explore their potential. However, the pace of change within the academic realm does not match the rapid release of new GenAI tools in the tech industry. The inevitable consequence could be students appearing to excel in their coursework with apparent ease, giving rise to a situation where *"our students have changed radically. Today's students are no longer the people our educational system was designed to teach"* [49]. This dynamic poses intriguing questions about how educators must adapt their strategies to accommodate the paradigm shift facilitated by GenAI tools.

In this study, we sought to propose a framework for answering the question *"How can educators guide students' interactions with GenAI to preserve core skill development during their foundational academic years?"* This framework accentuates GenAI's benefits and potential as a pedagogical instrument as well as demonstrating its failures. The framework equips instructors with tools for evaluating students' progress and understanding and seeks to prevent the Junior-Year Wall. This paper serves

as the foundational framework for future studies in which we are performing rigorous educational studies into the effectiveness of the AI-Lab in answering this and other research questions.

Given the recent emergence of readily accessible GenAI tools, the exploration of their application within academic environments is still in its early stages. A pivotal question surfaces within the context of integrating Generative AI into programming courses: *"When instructing students in utilizing GenAI, what facets of pedagogy and subject matter remain within the purview of educators?"* We contend that, especially in courses emphasizing the foundational aspects of Computer Science, educators should prioritize cultivating mathematical reasoning and problem-solving skills over mere coding proficiency. Echoing the sentiments of Leslie Lamport, coding is akin to the mechanical act of typing in the realm of programming, a sentiment well-expressed in his observation, *"coding is to programming what typing is to writing"* [50]. This insight highlights the necessity of emphasizing broader cognitive skills that transcend specific programming languages or tools.

Appendix

GenAI Lab Framework

This lab/lecture aims to introduce generative AI to students. It is a lab that focuses on teaching students some of the strengths of ChatGPT[8] and emphasizes that one cannot mindlessly trust the responses from ChatGPT. It ensures students understand when to use ChatGPT in their studies.

The lab/lecture will consist of three main parts:

1. **Pre-lab**: The students must complete this part of the lab before they attend the lecture/lab. It focuses on students having a basic exposure to ChatGPT to ensure that they are ready for the lecture/lab and do not interact with ChatGPT for the first time during the lab/lecture.
2. **Lab**: This interactive lab/lecture covers some aspects of ChatGPT. It uses ChatGPT as a learning tool where the students will learn a topic using it.
3. **Post-lab**: This worksheet is geared towards students using ChatGPT for various tasks. It may or may not be CS-specific. However, one part of this worksheet should focus on problems related to the topic covered in class (solving questions, generating new questions, evaluating ChatGPT's topical responses, etc.).

[8] Or any chosen chat-based generative AI.

Pre-lab

Note: At some point, this pre-lab introduces a topic that will be taught in the lab using ChatGPT. Choosing this topic carefully is imperative as ChatGPT can perform extremely well, averagely, or poorly, depending on the topic. Hence, the topic will set the tone of the entire lab.

Perspective: The following section is written as if the student is reading this as a worksheet. (This is an outline/example of what a pre-lab should be like. There are parts of this lab that refer to topics in computer science. The outline will largely be the same for other disciplines, with the computer science topics replaced with topics specific to the class/course this lab is being offered in.)

Please make sure that you do the pre-lab survey before starting this pre-lab.

Please ensure that you have completed the following steps before you attend the lab/lecture:

Step 1: You must have an account to use ChatGPT. If you don't have one already, please sign up for one at https://chat.openai.com/

Step 2: Once you have signed up for an account, familiarize yourself with the tool by getting comfortable with the user interface and by asking the tool a few questions.

For example, start with a general question: *"Tell me about yourself."* Once the tool responds, continue asking the tool some basic questions. You can talk to it as you would to a human. Some example questions are:

- What places are there to visit in Indiana? Or, does West Lafayette/Lafayette, Indiana, have anything to do?
- What is a good recipe for chocolate chip cookies?
- Help me plan a trip to <a place you want to visit>

Once you are comfortable with these general questions, move towards questions more in line with your studies in computer science. You can ask questions about topics that you have learned in this class and ask them to revisit the topic with you or reteach you a topic. Some example questions are:

- What is Big-O?
- What is a linked list? Help me visualize a linked list.
- What is a left-leaning red-black tree? Give me an example of a left leaning red black tree.

Note: If you have a question about one of its responses or it uses terminology you are uncomfortable with, try asking it follow-up questions. A general tip to any question you ask is to be specific. Tell it exactly what you want. Give it context. Give as many details as you can. For example, for the question - Help me plan a trip to <a place you want to visit>, you can tell it details like:

- How much time do you have?
- What time of the year do you want to visit that place, or are you open to any time during the year?
- Is there a specific thing/activity you are interested in at that place?
- Other relevant details that you might have.

Treat your prompt as if you were giving instructions to a personal assistant. They need to know every detail you can give them to make the best trip possible for you. The same is true of ChatGPT.

Step 3: Finally, let's introduce the topic that will be covered in the lab/lecture. Use (copy) the prompt that we give you for this task. We have provided you with an exact prompt that you will use for this lab and a version with the basic structure of the prompt written out. You only need to introduce yourself to the topic. We will be going into the details in the lab/lecture.

Prompt to copy: *"Act as a professor of computer science at a top 10 engineering university who has been trained to teach well. I want you to give me a short introduction to Huffman Coding. I already know String pattern matching and Tries. The learning objectives for this section are as follows: 1. Analyze and illustrate the process by which Huffman coding compresses data. 2. Decompose and evaluate the structure and information presented in a Huffman tree. 3. Compute the bit requirement for a specific set of data using Huffman coding and validate its efficiency."*

Note the very specific task we assigned: briefly introducing a particular topic.

Fill-in-the-blank Prompt: *"Act as a [insert top-level expert in the field you are trying to learn about, with more qualifications or titles if necessary]. I want you to introduce me to [whatever the topic is] briefly. I already know [list all relevant previous topics here]. The learning objectives for this section are as follows: [do some work to figure these out (see below), or ask the professor]."*

Feel free to ask questions or clarifications on the response ChatGPT generates. Chat with it like you would to a professor and ask for further explanations and examples. You can also ask it to help you visualize the topic you are trying to learn. Note that ChatGPT3.5 is not very good at giving visual examples for Huffman Encoding and some other graphical topics, in particular.

Note: You may want to try the sample prompt with a topic of your choice. A pro-tip is to use Bloom's Taxonomy for defining the learning objectives. You can define the learning objectives in your own language and ask ChatGPT to rephrase/rewrite them using Bloom's Taxonomy. More about Bloom's Taxonomy here: https://cft.vanderbilt.edu/guides-sub-pages/blooms-taxonomy/

You can use the following prompt to rewrite learning objectives using Bloom's Taxonomy: *"Act as a professor of engineering education to rephrase the learning objectives using Bloom's taxonomy. The learning objectives are: <List the learning objectives you want>."*

You can use this prompt to get learning objectives for introducing a topic: *"Act as a professor of engineering education and computer science. Give me three learning objectives using Bloom's taxonomy for the topic of Huffman Encoding for an in-class lecture."*

Lab/Lecture

This part of the lab will focus towards familiarizing students with uses of ChatGPT, when should it be used, when should it not be used, what it is good at, what might it not be good at, etc. This will also have an interactive part where the students will teach themselves a new topic. Students should have laptops to use ChatGPT in class, as the instructor does on the projector.

Start with: what is GenAI? How is it useful in daily life? What are some of the more straightforward tasks it can and is good at?

- Try using GPT.
- Make sure to include examples - Some of these have been covered in the pre-lab worksheet. A quick recap or live demonstration should be good to get everyone on the same page.

Then, proceed towards using GenAI in education. When should you use it, and when should you not use it?

- Have students brainstorm what GenAI is useful for as a class for a few minutes. Write down their responses.
- Personal tutor for conceptual questions:
  - It is good to teach oneself one topic.
  - Good to ask questions on a concept you don't understand.
  - It is good to generate practice questions on a topic you want to strengthen.
  - Feel free to give students this list of prompt phrases they can use to get better educational benefit out of the tools: "We would strongly encourage you to use the tools with phrases as follows:

    * Help me with X, *don't give me all the answers but help lead me to it*
    * *Provide suggestions for how to start* without solving my problem
    * *Find my bug but don't tell me exactly where it is*, help me discover it for myself
    * *Only give me a partial solution* to my problem, don't give it all away but guide me to the answer

> * Here's my problem, *help me find the issue by asking me guiding questions only*

- It is good to reword sentences to make them more precise and clear. It is also quite creative to add narrative to a story or a motivating example for a question or in a different scenario, which requires adding narrative to a story or a motivating example.
- Asking homework questions or project questions without understanding the topics is something that should not be done.
- A very important point to get across to the students is to make sure they know that not only will it (ChatGPT) make mistakes, but that it sounds very confident in them. Give a specific example here and say we'll play with this idea of mistakes in output to help teach you topics.
- It shouldn't be mindlessly trusted as it makes mistakes when solving some of the easiest questions. (something we will see later in the lab).

Do the students know how to use GenAI well? Go over how to use GenAI. Using the framework saw a drastic improvement in the answers provided by ChatGPT.

GenAI Framework: How Can We Use GenAI Well?

Understanding how to interact with GenAI:

- The single most important thing you can get from this section is that you must be clear, concise, and specific with what you want.
  - Imagine you are giving instructions to a 4–6 year-old child. They are very good at executing exactly what you tell them to do, assuming they know how to do it, but are not very good at understanding what you meant for them to do if it does not align with what you told them to do exactly. ChatGPT is like a 4–6 year-old child trained on the internet instead of just 4–6 years of human experiences.
  - Furthermore, you can converse with it ("Chat") just like you would a human. You can say things like "the part about X wasn't very clear, could you explain it differently, please?" or any other way you would communicate with a human being. It is a language model, so understanding what you say to it is what it is best at.
- Another thing we want to emphasize here is do not expect it always to be perfect, and do not give up because it didn't give you what you wanted the first time. Much like a Google search, you may have to tailor your search to get exactly what you want.
  - For example, if you ask it to give you 10 things that match your requirements, you may get 2–4 decent or good responses. From there, you can tell it which ones you liked and work with each one individually to make it better.

Crafting Your Prompt

There are three parts to crafting your prompt:

1. Set your context. For example, *"act as a collegiate professor with a PhD in computer Science for a sophomore level data structures and algorithms course. Students are familiar with Java."*
   - Always specify at least one expert to act as. You could specify multiple (e.g., "act as a computer science professor and experienced statistics researcher") or add qualifications as we did in the prelab if those qualifications aren't obvious (e.g., *"Act as a professor of computer science at Purdue University who has been trained how to teach well."*)
   - Tell it what setting that expert is acting in. These give it a starting point for what level of help you are asking for. For example, *"Act as a political science professor teaching a summer school for gifted high school students"* vs. *"Act as a political science professor teaching a summer school for academically advanced high school students"* tells two slightly different messages.
     - The first states that the students are smart but likely don't have an introduction to the material yet and have likely only experienced high school level education before.
     - The second states that students are smart and likely have taken collegiate-level dual-enrollment-style courses, and thus are more accustomed with the speed and density of collegiate-level material.
     - In our prompt, we specified *"sophomore level"* class because data structures and algorithms courses can be offered at many different levels, so you could get anything from beginner to advanced without being more specific.
   - Lastly, tell it what your audience is specifically familiar with when asking it to teach you. This tells it what technical lingo is OK to use and what things it can assume you know/skip over in explanations.
2. Ask it your specific request. For example, *"Give me 10 problems to practice Huffman encoding."*

- It won't be good on most of them. You may get 2–4 good problems in a set of 10. Don't expect it to be perfect.
- In our request, we did several things of note. Let's break down the prompt:
  - *"give me"*: a command, a request, specific and directed at it.
  - *"a set of 10 problems"*: a specific large number so that you have a variation to choose from on a specific, actionable thing (a set of problems).
  - *"to practice"*: part of your goals, it gives problems that are more tailored to learning and practicing the different components of the subject.
  - *"Huffman encoding"*: words that clearly identify the specific topic you want to practice. You may get more or less specific, with varying success (likely higher success the more specific you get). For example, you could ask for problems about "compression trees" in general, which will give you more theoretical questions, or about "using frequency analysis to create a Huffman Tree," which will give you a set of situations to practice that specific task.

3. Tell it your goals. This is an optional part of the prompt. If your task is simple and short enough, there is likely no need to tell it your goals. If you want to craft a long set of questions and examples or a detailed itinerary or give you ideas or any number of complex tasks, it is best to tell it where you generally want to go.
   - For example, in the prelab, we told it what our learning objectives were for introducing this topic. From there, it knew generally what we wanted you to learn and could give a more tailored response.
   - I asked ChatGPT4.0 what tasks would be best for me to specify my goals in the prompt.

Interacting and Dealing with the Response

It's a chatbot! Use the chat feature! You can tell it to be more specific, work with a particular example, give you answers to a particular problem it created, or do a large variety of other things that tailor responses to exactly what you want. Some important things to note:

1. When you are asking it to refine a previous response (instead of asking for new content related to previous responses), make sure you **tell it what you liked or didn't like from its response that you want it to do more of or change**, respectively, before asking it to reword. Simply asking it to reword could give you many even worse responses. Don't stop being as specific as you can during your conversation.
2. When refining particular subsets of lists or specific parts of a response, try to **separate your queries into one per list element or part of a response**. It doesn't do as well when you ask it to refine all the questions simultaneously. If you think about it, your goals for refining are likely a little different for each question.
3. If there have been many prompts since you asked it to act a particular way or since a response that you keep referencing, **try reminding it what you are talking about**. The LLMs can get confused when the conversation is too long without direct reference to certain things. It can, for example, forget that you modified code a long time ago, so it may be helpful to remind it what code you liked by telling it that and copying and pasting the code.
4. **It can also be helpful just to start a new chat**. Especially when it has gotten a lot of things wrong, it can be helpful to give it a fresh start, just like you do when you come back to debugging your code after a break (except it has to be retaught the context, etc).

Let's Try it Out!

Proceed towards having students use AI: Use ChatGPT to teach students about a topic. Preferably a topic on the easier side, something they have heard of but still something the students haven't learned about. E.g., stacks and queues, hash tables, or Huffman Coding in CS 251.

Based on the lab's goal - Choose a topic that ChatGPT performs well on. Choose a topic on which ChatGPT does not perform well. Or maybe choose two topics to show the contrast of the performance of GenAI based on different topics. An instructor should prepare for these things by using GenAI and preparing for the topic they decide to teach.

This will continue the pre-lab prompt in which the students were introduced to the topic to be taught using ChatGPT (e.g., Huffman Coding in the pre-lab above).

Fill-in-the-blank Prompt: *"Act as a [insert top-level expert in the field you are trying to learn about, with more qualifications or titles if necessary]. I want you to introduce me to [whatever the topic is] briefly. I already know [list all relevant previous topics here]. The learning objectives for this section are as follows: [do some work to figure these out (see below), or ask the professor]."*

Note the very specific task we assigned: briefly introducing a particular topic.

This lab part should be about the student interacting with ChatGPT. Make groups of 2 students to encourage discussion while learning the topic (Think, Pair, Share educational framework).

- Help students ask questions to AI and give them enough time to go over a topic.
- Ideally, a topic should not take more than 20 mins to be taught. Provide sample questions or a start like the prompt above.

Instructors should consider the following questions in preparation for the lab.

- What should be the prompts provided? What should the students do if they have a followup question (in-class) to a response from ChatGPT?
- Something to think about is whether ChatGPT is good with examples of the topic or solving problems related to the topic or not. Can ChatGPT provide visual representation to make explanation/problem solving easier to understand?
- Is solving an example by hand necessary? If it is, it is recommended to have the students go over the topic using ChatGPT and then solve an example by hand on the class board. Once an example is done by hand, ask students to generate more examples using ChatGPT.

At this stage, students should be able to solve the example generated by ChatGPT by themselves. Students should feel comfortable answering questions and solving examples about the topic discussed.

- Ask them to evaluate the completeness and correctness of the example solved by ChatGPT. Ask two groups to discuss and display in front of the class.
- Depending on the topic chosen and ChatGPT's expected performance, ask the class if anyone got a correct/wrong example solution. Call on groups from both sides to discuss and display the examples they got from ChatGPT.
- Time permits solving the incorrect examples from ChatGPT with the help of the class.
- While it will be ideal to have the students experience chatGPT making mistakes on questions (Topic dependent), having a backup of pictures of instances where ChatGPT gave wrong answers (questions on the topics they have already learned) is a good idea to share with students. You should have run across incorrect examples in your preparation for the course.

Post-Lab

This will be a worksheet geared towards students using ChatGPT for various tasks. It may or may not be CS-specific. However, one part of this worksheet should focus on problems related to the topic covered in class (solving questions, generating new questions, evaluating ChatGPT's topical responses, etc.).

If the instructor chooses to give tasks unrelated to the topic covered in class, they can choose to give tasks related to something that may be important or relevant to the class but cannot be covered in a lecture setting. For example, a task that asks students to teach themselves a topic that will benefit them but is not on the list of canonical topics (e.g., Web Development, JUnit testing, etc.)

An important task related to the topic that was covered in class is to have students generate problem questions or examples about the topic they learned using GenAI:

- Have them generate questions about a topic (a topic that they should be comfortable with and was taught recently - the topic they just learned about may be used, however having students come up with good learning objectives may be hard. Learning objectives should be provided however are not necessary if the topic is relatively straightforward and the instructors think that the students have a good grasp of it.)
- Once they generate questions about a topic they were comfortable with (to get them comfortable with generating and evaluating questions), ask them to generate questions about the topic they learned in the lab. Learning objectives must be provided for this task.
- Tell them to give us 2 questions they liked and why they liked it. Ask them to solve those two questions using ChatGPT and also ask them to solve them on their own. Ensure you emphasize why they should solve it using ChatGPT after they solve it independently. Some reasons are:
  - It ensures the student knows if they learned the topic.
  - Gives them a different perspective.
  - The answer the student gets may or may not match what ChatGPT gave, etc.
- Ask the students to evaluate the answers that ChatGPT gave using these questions:
  - Is the answer technically correct in everything it said?
  - Does it answer the question asked?
  - Is it a complete answer? i.e., does it cover everything it should about the question?
- Based on how much the instructor plans to discuss in class, the instructor may or may not provide 1–5 instructor questions on the topic taught. The task will be again to solve it on their own using ChatGPT and analyze the answers.

Note: While the students are solving questions using ChatGPT, remind them that the tool will likely make mistakes.

This will help students realize that they cannot mindlessly trust ChatGPT.

Author Contributions Noted in title page.

Funding The authors declare that they did not receive funding for this study.

Data Availability Not Applicable.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Research Involving Human and/or Animals IRB exempt due to class data.

Informed Consent Informed consent was obtained from all individual participants included in the study.

References

1. Kalles D. Artificial intelligence meets software engineering in computing education. In: Proceedings of the 9th Hellenic Conference on Artificial Intelligence. SETN '16. Association for Computing Machinery, New York, NY, USA; 2016. https://doi.org/10.1145/2903220.2903223.
2. Chassignol M, Khoroshavin A, Klimova A, Bilyatdinova A. Artificial intelligence trends in education: a narrative overview. Procedia Computer Science. 2018;136:16–24. https://doi.org/10.1016/j.procs.2018.08.233. 7th International Young Scientists Conference on Computational Science, YSC2018, 02-06 July2018, Heraklion, Greece.
3. Chen L, Chen P, Lin Z. Artificial intelligence in education: A review. IEEE Access. 2020;8:75264–78. https://doi.org/10.1109/ACCESS.2020.2988510.
4. Holmes W, Tuomi I. State of the art and practice in ai in education. Eur J Educ. 2022;57(4):542–70.
5. Terry OK. I'm a Student. You Have No Idea How Much We're Using ChatGPT. Accessed on: August 14, 2023. https://www.chronicle.com/article/im-a-student-you-have-no-idea-how-much-were-using-chatgpt. 2023.
6. Nolan B. College professors are considering creative ways to stop students from using AI to cheat. https://www.businessinsider.com/ai-chatgpt-college-professors-students-cheating-2023-1. 2023.
7. Weekend PN. Educators worry about students using artificial intelligence to cheat. https://www.pbs.org/newshour/show/educators-worry-about-students-using-artificial-intelligence-to-cheat. 2023.
8. Puryear B, Sprint G. Github copilot in the classroom: learning to code with ai assistance. J Comput Sci Coll. 2022;38(1):37–47.
9. Wermelinger M. Using github copilot to solve simple programming problems. In: Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1. SIGCSE 2023, pp. 172–178. Association for Computing Machinery, New York, NY, USA; 2023. https://doi.org/10.1145/3545945.3569830.
10. Finnie-Ansley J, Denny P, Becker BA, Luxton-Reilly A, Prather J. The robots are coming: Exploring the implications of openai codex on introductory programming. In: Proceedings of the 24th Australasian Computing Education Conference. ACE '22, pp. 10–19. Association for Computing Machinery, New York, NY, USA; 2022. https://doi.org/10.1145/3511861.3511863.
11. Moradi Dakhel A, Majdinasab V, Nikanjam A, Khomh F, Desmarais MC, Jiang ZMJ. Github copilot ai pair programmer: Asset or liability? J Syst Softw. 2023;203: 111734. https://doi.org/10.1016/j.jss.2023.111734.
12. Becker BA, Denny P, Finnie-Ansley J, Luxton-Reilly A, Prather J, Santos EA. Programming is hard - or at least it used to be: Educational opportunities and challenges of ai code generation. In: Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1. SIGCSE 2023, pp. 500–506. Association for Computing Machinery, New York, NY, USA; 2023. https://doi.org/10.1145/3545945.3569759.
13. Qadir J. Engineering education in the era of chatgpt: Promise and pitfalls of generative ai for education. In: 2023 IEEE Global Engineering Education Conference (EDUCON); 2023. pp. 1–9. https://doi.org/10.1109/EDUCON54358.2023.10125121.
14. Denny P, Prather J, Becker BA, Finnie-Ansley J, Hellas A, Leinonen J, Luxton-Reilly A, Reeves BN, Santos EA, Sarsa S. Computing Education in the Era of Generative AI. arXiv:2306.02608. 2023.
15. Yilmaz R, Karaoglan Yilmaz FG. Augmented intelligence in programming learning: Examining student views on the use of chatgpt for programming learning. Computers in Human Behavior: Artificial Humans. 2023;1(2): 100005. https://doi.org/10.1016/j.chbah.2023.100005.
16. Yilmaz R, Karaoglan Yilmaz FG. The effect of generative artificial intelligence (ai)-based tool use on students' computational thinking skills, programming self-efficacy and motivation. Computers and Education: Artificial Intelligence. 2023;4: 100147. https://doi.org/10.1016/j.caeai.2023.100147.
17. Ouh EL, Gan BKS, Shim KJ, Wlodkowski S. Chatgpt, can you generate solutions for my coding exercises? an evaluation on its effectiveness in an undergraduate java programming course. In: Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1. ITiCSE 2023, pp. 54–60. Association for Computing Machinery, New York, NY, USA; 2023. https://doi.org/10.1145/3587102.3588794.
18. Lau S, Guo P. From "ban it till we understand it" to "resistance is futile": How university programming instructors plan to adapt as more students use ai code generation and explanation tools such as chatgpt and github copilot. In: Proceedings of the The 19th ACM Conference on International Computing Education Research. ICER '23, vol. 1. https://icer2023.acm.org/details/icer-2023-papers/8/From-Ban-It-Till-We-Understand-It-to-Resistance-is-Futile-How-University-Program. 2023.
19. Savelka J, Agarwal A, An M, Bogart C, Sakr M. Thrilled by your progress! large language models (gpt-4) no longer struggle to pass assessments in higher education programming courses. In: Proceedings of the The 19th ACM Conference on International Computing Education Research. ICER '23, vol. 1. https://icer2023.acm.org/details/icer-2023-papers/6/Thrilled-by-Your-Progress-Large-Language-Models-GPT-4-No-Longer-Struggle-to-Pass-A. 2023.
20. Perkel J. Six tips for better coding with chatgpt. Nature. 2023;618(7964):422–3. https://doi.org/10.1038/d41586-023-01833-0.
21. Hellas A, Leinonen J, Sarsa S, Koutcheme C, Kujanpää L, Sorva J. Exploring the responses of large language models to beginner programmers' help requests. In: Proceedings of the The 19th ACM Conference on International Computing Education Research. ICER '23, vol. 1, pp. 422–431. Chicago, Illinois; 2023. https://icer2023.acm.org/details/icer-2023-papers/7/Exploring-the-Responses-of-Large-Language-Models-to-Beginner-Programmers-Help-Reques.

22. Giray L. Prompt engineering with chatgpt: A guide for academic writers. Ann Biomed Eng. 2023. https://doi.org/10.1007/s10439-023-03272-4.
23. OpenAI: GPT-4 Technical Report. 2023.
24. Dobslaw F, Bergh P. Experiences with remote examination formats in light of gpt-4. In: Proceedings of the 5th European Conference on Software Engineering Education. ECSEE '23, pp. 220–225. Association for Computing Machinery, New York, NY, USA; 2023. https://doi.org/10.1145/3593663.3593695.
25. Liffiton M, Sheese B, Savelka J, Denny P. CodeHelp: Using Large Language Models with Guardrails for Scalable Support in Programming Classes. 2023.
26. Gutiérrez JD. Guidelines for the use of artificial intelligence in university courses. Version 4.3.1, Universidad del Rosario. License C.C. BY 4.0. February 28 2023. https://forogpp.files.wordpress.com/2023/02/guidelines-for-the-use-of-artificial-intelligence-in-university-courses-v4.3.1.pdf.
27. Guerriero SA. 4 ways teachers can harness the power of chatgpt; 2023. Accessed on: August 14, 2023.
28. Su J, Yang W. Unlocking the power of chatgpt: A framework for applying generative ai in education. ECNU Review of Education. 2023;20965311231168424. https://doi.org/10.1177/20965311231168423.
29. Bull C, Kharrufa A. Generative ai assistants in software development education: A vision for integrating generative ai into educational practice, not instinctively defending against it. IEEE Software. 2023;1–9. https://doi.org/10.1109/MS.2023.3300574.
30. Ahmad SF, Han H, Alam MM, Rehmat MK, Irshad M, Arraño-Muñoz M, Ariza-Montes A. Impact of artificial intelligence on human loss in decision making, laziness and safety in education. Humanities and Social Sciences Communications. 2023;10(1). https://doi.org/10.1057/s41599-023-01787-8.
31. Prasad P, Sane A. A self-regulated learning framework using generative ai and its application in cs educational intervention design. In: Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1. SIGCSE 2024, pp. 1070–1076. Association for Computing Machinery, New York, NY, USA; 2024. https://doi.org/10.1145/3626252.3630828.
32. Liu R, Zenke C, Liu C, Holmes A, Thornton P, Malan DJ. 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, pp. 750–756. Association for Computing Machinery, New York, NY, USA; 2024. https://doi.org/10.1145/3626252.3630938.
33. Chen M, Tworek J, Jun H, Yuan Q, Oliveira Pinto HP, Kaplan J, Edwards H, Burda Y, Joseph N, Brockman G, Ray A, Puri R, Krueger G, Petrov M, Khlaaf H, Sastry G, Mishkin P, Chan B, Gray S, Ryder N, Pavlov M, Power A, Kaiser L, Bavarian M, Winter C, Tillet P, Such FP, Cummings D, Plappert M, Chantzis F, Barnes E, Herbert-Voss A, Guss WH, Nichol A, Paino A, Tezak N, Tang J, Babuschkin I, Balaji S, Jain S, Saunders W, Hesse C, Carr AN, Leike J, Achiam J, Misra V, Morikawa E, Radford A, Knight M, Brundage M, Murati M, Mayer K, Welinder P, McGrew B, Amodei D, McCandlish S, Sutskever I, Zaremba W. Evaluating Large Language Models Trained on Code. 2021.
34. Jonsson M, Tholander J. Cracking the code: Co-coding with ai in creative programming education. In: Proceedings of the 14th Conference on Creativity and Cognition. CC '22, pp. 5–14. Association for Computing Machinery, New York, NY, USA; 2022. https://doi.org/10.1145/3527927.3532801.
35. Sue Sentance JW, Kallia M. Teaching computer programming with primm: a sociocultural perspective. Comput Sci Educ. 2019;29(2–3):136–76. https://doi.org/10.1080/08993408.2019.1608781.
36. Philbin CA. Exploring the potential of artificial intelligence program generators in computer programming education for students. ACM Inroads. 2023;14(3):30–8.
37. Howard JL, Bureau J, Guay F, Chong JXY, Ryan RM. Student motivation and associated outcomes: A meta-analysis from self-determination theory. Perspect Psychol Sci. 2021;16(6):1300–23. https://doi.org/10.1177/1745691620966789. (**PMID: 33593153**).
38. Sciences NA, Engineering Medicine. How People Learn II: Learners, Contexts, and Cultures. The National Academies Press, Washington, DC; 2018. https://doi.org/10.17226/24783. https://nap.nationalacademies.org/catalog/24783/how-people-learn-ii-learners-contexts-and-cultures.
39. Ryan RM, Deci EL. Intrinsic and extrinsic motivation from a self-determination theory perspective: Definitions, theory, practices, and future directions. Contemp Educ Psychol. 2020;61: 101860. https://doi.org/10.1016/j.cedpsych.2020.101860.
40. Dickey E, Bejarano A. A Model for Integrating Generative AI into Course Content Development. arXiv:2308.12276. 2023.
41. Kabir S, Udo-Imeh DN, Kou B, Zhang T. Who Answers It Better? An In-Depth Analysis of ChatGPT and Stack Overflow Answers to Software Engineering Questions. 2023.
42. White J, Fu Q, Hays S, Sandborn M, Olea C, Gilbert H, Elnashar A, Spencer-Smith J, Schmidt DC. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. 2023.
43. Anderson L, Krathwohl D, Airasian P, Cruikshank K, Mayer R, Pintrich P, Raths J, Wittrock M. A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. 1st ed. New York: Pearson; 2000.
44. Zamfirescu-Pereira JD, Wong RY, Hartmann B, Yang Q. Why johnny can't prompt: How non-ai experts try (and fail) to design llm prompts. In: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. CHI '23. Association for Computing Machinery, New York, NY, USA; 2023. https://doi.org/10.1145/3544548.3581388.
45. Felder RM, Brent R. Teaching and Learning STEM: A Practical Guide. Amsterdam: Wiley; 2016.
46. Xu W, Ouyang F. The application of ai technologies in stem education: a systematic review from 2011 to 2021. International Journal of STEM Education. 2022;9(59):2196–7822. https://doi.org/10.1186/s40594-022-00377-5.
47. Zhai X, Haudek K, Shi L, Nehm R, Urban-Lurain M. From substitution to redefinition: A framework of machine learning-based science assessment. J Res Sci Teach. 2020;57(9):1430–59. https://doi.org/10.1002/tea.21658.
48. Jones C, Shao B. The net generation and digital natives: implications for higher education. Higher Education Academy, York. 2011. https://oro.open.ac.uk/30014/.
49. Prensky M. Digital natives, digital immigrants. Gifted. 2005;135:29–31.
50. Magazine Q. The Man Who Revolutionized Computer Science With Math. Accessed on: August 14, 2023. https://www.youtube.com/watch?v=rkZzg7Vowao. 2022.