2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION
Paper ID #52336

Developing AI Literacy in First-Year Engineering: Student Prompting, Evaluation, and Implications for Instruction

Dr. Elizabeth Flanagan, Clemson University

Libby Flanagan is a lecturer in General Engineering at Clemson University. She received her B.S. in Biosystems Engineering from the Clemson University Honors Program in 2017, along with a minor in Spanish Language Studies. She completed a two-year teaching appointment with Teach for America in 2019, where she taught 6th-grade math and computer science in Tulsa, Oklahoma. She earned her M.S. in Biosystems Engineering at Clemson University in 2021, during which she was an NRT Resilient Infrastructure and Environmental Systems Fellow. Libby holds a Ph.D. in Engineering and Science Education with a research focus on empathy in engineering education.

Grace Fulmer Crocker, Clemson University

Grace F. Crocker is a Lecturer of General Engineering at Clemson University, Clemson, SC, where she received her BS, MS, and PhD in civil engineering in 2018, 2019, and 2022, respectively. Her primary research interests include resilient structural-architectural systems, as well as interdisciplinary education for undergraduate engineering and architecture students.

Developing AI Literacy in First-Year Engineering: Student Prompting, Evaluation, and Implications for Instruction

Abstract

This **complete research paper** investigates the current level of artificial intelligence (AI) literacy among first-year engineering students by examining how they formulate prompts, interpret AI responses, and apply those responses to engineering problem-solving tasks. In addition to characterizing students' AI use, the study aims to inform instructional practice by identifying specific challenges and opportunities for supporting productive and critical AI use in early engineering courses. While research has documented widespread student adoption of AI tools, less is known about how novice engineering students prompt these tools, evaluate AI-generated outputs, and demonstrate emerging forms of AI literacy. This study examines aspects of AI literacy among first-year engineering students through their interactions with a generative AI tool in a programming-based engineering task. Participants were drawn from an introductory engineering course at a large public R1 university. Students completed a survey describing their AI prompting practices and evaluating a sample AI-generated solution to a familiar engineering problem. In addition, student pairs completed a MATLAB-based programming activity in which use of an AI assistant was permitted and required to be documented through prompt logs and reflective reports. Qualitative analysis of prompt lists, reflections, and survey responses was conducted using an established AI literacy framework, with particular attention to effective prompting and critical evaluation of AI outputs. Results indicate that most first-year engineering students operate at or below a novice level of prompting, a component of AI literacy. Common prompting strategies included copying and pasting full problem statements, using AI primarily for debugging, requesting assistance one line of code at a time, or avoiding AI altogether. Students who employed more specific, constrained prompts or used AI to explain errors rather than generate complete solutions reported greater success and less frustration. However, a significant proportion of students were unable to identify errors in AI-generated solutions, highlighting limitations in critical evaluation skills. Together, these findings are synthesized into an actionable set of instructional recommendations for integrating AI tools into first-year engineering courses while maintaining student ownership, disciplinary reasoning, and critical thinking. These findings suggest that exposure to AI tools alone is insufficient for developing productive and ethical AI use in engineering education. Instead, explicit instruction in prompt construction, critical evaluation of AI outputs, and ethical boundaries is necessary, particularly in first-year courses where foundational problem-solving habits are formed. This study provides baseline evidence of first-year engineering students' AI literacy and offers guidance for integrating AI literacy instruction into early engineering curricula.

Introduction

Generative artificial intelligence (AI) tools are rapidly reshaping how students approach learning, problem-solving, and communication in engineering education. Tools such as large language models (LLMs) can generate explanations, draft algorithms, troubleshoot errors, and propose solutions to engineering problems within seconds. As a result, first-year engineering students are increasingly encountering AI not as an optional technology, but as a pervasive presence in both academic and professional contexts. This shift presents a critical opportunity to support novice learners; however, it also introduces new challenges related to how students interpret, evaluate, and appropriately integrate AI-generated information into their learning processes.

For first-year engineering students in particular, AI occupies a complex role. On one hand, AI tools can scaffold early learning by helping students translate word problems into structured approaches, articulate algorithms, or debug early mistakes [1]. On the other hand, effective use of AI requires skills that many novice learners have not yet developed, including the ability to clearly articulate goals through prompting, critically evaluate AI-generated outputs, and verify the correctness and relevance of suggested solutions [2]. Without these skills, students may experience frustration, overreliance on AI, or confusion about the boundaries between productive assistance and academic dishonesty.

These challenges highlight the growing importance of artificial intelligence literacy in engineering education. AI literacy extends beyond basic tool usage to include understanding AI limitations, recognizing potential errors or biases, and making informed judgments about when and how AI should support human reasoning. In the context of engineering education, AI literacy also intersects with disciplinary ways of thinking, including algorithmic reasoning, verification, and iterative problem-solving.

Despite the rapid adoption of generative AI by students [1], [3], [4], there remains limited empirical understanding of how first-year engineering students currently use these tools, particularly with respect to prompt formulation and evaluation of AI-generated outputs. Little research has systematically documented the specific strategies first-year engineering students use when prompting AI, how they interpret responses, or how these behaviors align with broader dimensions of AI literacy [5].

Understanding students' current AI practices is especially important at the first-year level, where foundational habits related to problem-solving, computational thinking, and ethical tool use are being formed. If students enter engineering programs already relying on AI tools without the skills to evaluate or refine AI output, these practices may persist and potentially undermine deeper learning. Conversely, identifying students' existing strategies and misconceptions can inform targeted instructional interventions that integrate AI use with core engineering learning objectives.

This study seeks to address this gap by examining some aspects of AI literacy of first-year engineering students through their interactions with generative AI tools. Specifically, the research investigates how students craft prompts, how they interpret and evaluate AI-generated responses to an engineering word problem, and how these practices reflect broader dimensions of AI literacy. By documenting baseline AI literacy and prompting competence, this work provides empirical grounding for future instructional design and longitudinal research aimed at supporting ethical, effective, and critical AI use in early engineering education.

The study is guided by the following research questions:

1. What strategies do first-year engineering students use when crafting prompts for generative AI tools?
2. How do first-year engineering students' evaluations of AI-generated outputs align with criteria in an AI literacy framework?
3. How do these practices relate to broader dimensions of AI literacy in engineering education?

Literature Review

A report based on June 15, 2025 data indicates that "Americans are much more concerned than excited about the increased use of AI in daily life, with a majority saying they want more control over how AI is used in their lives" [4, p. 5]. The report also indicates that "Far larger shares say AI will erode than improve people's ability to think creatively and form meaningful relationships" [4, p. 5]. This result is even stronger in adults under 30, where "Concern about people's ability to do things getting worse because of AI use is somewhat higher …. Roughly six-in-ten adults under 30 (57%) say they are extremely or very concerned about this, compared with 46% of adults ages 65 and older" [4, p. 21]. With a vast majority of first-year engineering students falling into this age group at the study institution, the concern among students is likely to be high.

A study out of Saudi Arabia from March to June 2024 on all first-year students (not limited to engineering), "indicates that 78.7% of students frequently use GenAI tools, while 21.3% do not, often due to a lack of knowledge or interest" [6, p. 1]. Students reported that "AI might negatively affect their learning outcomes, [and] these students also believed that overreliance on AI tools could hinder the development of critical thinking and problem-solving abilities" [6, p. 12]. Scores for "provide unreliable information" and "provide inaccurate or false references" (2.28) were particularly high, suggesting that students are aware of potential issues with AI use [6, p. 13]. This once again demonstrates that students all over the world are concerned about how AI is going to affect their ability to think critically.

At a university in the US in 2024, "the percentage of students who had tried ChatGPT was 69%, which was nearly double that of Spring 2023, revealing a rapid increase in use in the engineering student population" [3, p. 11]. Among first-year students, "experience was correlated with students' ethical opinions. Those with little to no experience felt that it would be unethical to use ChatGPT in many contexts, and the few students who reported themselves to have had expert-level experience (used ChatGPT regularly with specific purposes in mind) felt that it was ethical" [3, p. 12]. Students currently make decisions about AI use based on their own perceptions of what is ethical; however, rather than leaving this complex and nuanced judgment entirely to students, educators must begin to define and articulate what constitutes ethical use [7].

A recent literature review found that "the number of published articles on ChatGPT in the education domain has increased dramatically" and "research on ChatGPT in the education domain has been conducted almost all over the world" [1, p. 5]. They identified five frameworks that were frequently used, including the Unified Theory of Acceptance and Use of Technology (UTAUT), task-technology fit (TTF), Technology Acceptance Model (TAM), Social Cognitive Theory (SCT), and Information Systems Success Model (ISSM) [1]. The review included many ways that students use AI, specifically ChatGPT, including coding and programming assistance and debugging assistance [1]. While there are many frameworks which we might use to study students' adaptation of the technology, we need a unified framework for what AI literacy looks like for engineering, and particularly first-year engineering students.

Studies focused on first-year engineering students in the United States have examined how they use LLMs to aid in writing research papers. Instructors teamed up with the librarians to design a module that would "train students on how to use an AI large language model generative chatbot, and second, to train the students on AI Literacy so they can analyze and interpret the synthetically generated outputs" [8, p. 3]. The students in this study increased their ability to

recognize text generated by ChatGPT, but students also reported that AI "is not yet advanced or trustworthy enough to perform engineering capabilities independently" [8, p. 8]. The question remains: when it comes to engineering specific tasks, like writing code, how are students interacting with and experiencing AI?

A survey conducted at the University of New Haven looked at how AI technologies are currently used within the engineering curriculum. The qualitative portions of the survey reflected that students have " a desire for AI applications that are directly relevant to their engineering disciplines and that enhance practical skills" and that "AI should not replace fundamental engineering skills" [7], indicating there is a fine balance that must be struck in teaching students how to use AI but not allowing it to replace their engineering skills.

Some work has been done in creating AI literacy courses, separate from the first-year engineering courses students already take. For example, "The World of AI" course in India examines AI and the Planet, AI and the Society, and AI and the Workplace. From this course "students' comprehension of AI challenges improved across diverse metrics like (a) increased awareness of AI's environmental impact, and (b) efficient corrective solutions for AI fairness" [9]. In being a separate course, it could be possible that students are not seeing the connection to their engineering disciplines.

In contrast, a self-learning and project-based learning course in Switzerland intertwined writing and programming in their AI literacy course. Students in the course successfully combined practical experience with AI theory in self-directed projects, such as deep fakes. Feedback for this course was very positive, although attendance was not required and was low (30-60 percent) [10].

Together, these studies reveal a consistent narrative: first-year students are simultaneously enthusiastic about and wary of AI technologies. Their concerns, ranging from misinformation and overreliance to ethical uncertainty, mirror broader societal debates about AI's role in learning and creativity. Yet, despite the challenges, these findings underscore a critical opportunity for engineering educators. By embedding AI literacy directly into the curriculum, universities can help first-year students develop both competence and confidence to engage responsibly with AI in their academic and professional futures.

AI Literacy Framework

AI literacy frameworks are constantly evolving as the educational community continues to grapple with determining what AI skills we want students to have [11]. For this study we used the working framework developed by the Clemson University library [2], which was adapted from Lo, L. S. [12] and EDUCAUSE [13] and AAC&U Value Rubrics for "Ethical Reasoning" and for "Information Literacy.*"* The framework includes aspects for Understanding of AI Concepts and Terminology, Critical Evaluation of AI Outputs and Tools, Effective Prompting and Interaction, Societal Impact, Ethical Awareness and Responsibility, and Information Literacy. While all of these dimensions are sure to be critical in the age of AI, our study focused on evaluation of outputs and effective prompting. All student work we encountered fell under the novice or intermediate categories, although the rubric extends to advanced and expert. A snippet of the rubric is included below in Table 1 (please see [2] for the full rubric).

Table 1. Portion of AI Literacy Framework from [2]

Dimension / Skill	Novice	Intermediate
Effective Prompting & Interaction	Uses simple, direct prompts (basic questions or requests) to generate content; recognizes variability in outputs.	Applies prompt generation guidelines (clarity, context, specificity) to refine prompts; applies prompts across different modalities (e.g., text-to-image, text-to-video); begins experimenting with adding constraints and role-based instructions.
Critical Evaluation of AI Outputs / Tools	Identifies ways to evaluate appropriate uses and outputs. Identifies obvious errors or obvious hallucinations in AI-generated content; Understands and can point out that there are different AI tools for various purposes.	Evaluates and questions sources, flags possible misinformation or bias in results, begins to compare and contrast strengths/weaknesses of the outputs from multiple tools.

Methods

Participants and Course Context

Participants were drawn from an introductory engineering course at a large public R1 university in the fall of 2025. All participants are enrolled in a Programming and Problem-Solving Skills course, in which students learn the fundamentals of programming (MATLAB) and apply them to engineering problems. The course is heavily coordinated, and during the semester of study, two instructors taught four total sections. During the last several weeks of the course, the instructors spent about thirty minutes of class discussing a very brief introduction to the basics of AI use in engineering. This discussion included AI vocabulary, the differences between model types, how AI works at a high level, and some ethical concerns.

Out of 192 students enrolled in the course, 35 are included in this analysis. Participants were selected from all four sections if they consented to participate in the study and submitted the assignment individually, or if they were a part of a group and the entire group consented to participate. Additionally, participants were removed if they did not answer the qualitative portions of the survey, as the dataset was incomplete. When asked, “Choose one or more races that you consider yourself to be,” 24 selected White, 2 selected Black or African American, 3 selected Asian, 3 selected multiple, and 3 selected other or preferred not to say. 12 participants describe themselves as female, 21 male, and 2 preferred not to say. All students in this course are still considered “General Engineering” students, but their intended majors include Automotive 1, Biosystems 1, Civil 7, Computer 3, Electrical 5, Environmental 1, Industrial 6, Materials Science 1, and Mechanical 10 Engineering.

Data Collection Instruments and Timeline

The data for this study comes from two sources. The first was the last lab of the course, where students had to integrate their code with an Arduino for the first time and were allowed to use MATLAB Copilot for the first time. The second was a survey offered as a completion grade for the course, in which students did not have to participate in the research to receive full credit for the assignment (IRB approval #2025-1243). The lab was due two weeks before the final exam, and the survey was due on the last day of class, which was 10 days apart.

Programming Activity and AI Prompt Logs

Each student worked individually or within a pair to complete an engineering-related programming activity in MATLAB over the course of one week (four hours of class time, unlimited work time outside of class), during which the use of MATLAB Copilot was unrestricted. The activity required students to integrate an Arduino into their MATLAB program, a task that was completely unfamiliar to them. The ultimate goal of the activity was to design a greenhouse monitoring system that measured temperature and turned on a warning light and, if applicable, a fan when the temperature was out of range. The system should work regardless of the time of day, even though the acceptable temperature range for the greenhouse was higher during the day, based on a measurement of the light intensity.

Deliverables for this activity included the following:

- Code functionality (graded using an Autograder system and physical testing in class)
- A properly documented PDF of the program
- A short report including an algorithm, a list of prompts used with MATLAB Copilot, a summary of any changes made to Copilot output, and a short reflection on their Copilot experience

Throughout the semester, students in this course were asked to refrain from using certain coding practices. These practices were intentionally excluded from the curriculum to improve coding comprehension. For example, rather than using “while true-break" syntax to control a while loop, students were expected to control their while loops with appropriate conditions. It was determined that continuing to prohibit the use of these practices for this activity would maintain consistency in the course, as well as add an additional challenge by requiring the students to more critically evaluate their output from Copilot. The following were specified as restricted coding syntax for the activity discussed above:

- while true, while ~false, while(1)
- break, return, continue
- try, catch
- switch, case
- disp
- %i, %d

Students were also warned that the following actions would constitute academic dishonesty on this activity:

- Failure to produce their own MATLAB code and Copilot prompt log.
- Sharing code or prompts with other students.

- Failure to submit work that reflected their own understanding. Assistance was limited to the AI-based tool MATLAB Copilot and was permitted only if the students were able to explain and justify every line of code. If the instructors suspected that code was copied, pasted, or generated without comprehension, students could be asked to meet and verbally describe their implementation. Inability to do so would be treated as an academic integrity violation.

Survey

Students completed a survey on their AI use and literacy in which they described their process for crafting prompts. The survey began by asking if they used AI before this semester, during this semester, how often they use it, what tool they use most frequently, and on what types of academic tasks. Then it asked students to rate on a Likert-like scale how confident they were that the AI response was accurate, ethical, and unbiased. Then it asked students, "How do you create AI prompts?", "How skilled do you feel at writing effective AI prompts?", and "When the AI response is wrong or confusing, what do you usually do?". Next, "Describe a time when you used an AI tool to help with an engineering-related task. When did it do well, and what did it not do well?", "What challenges or concerns do you have about using AI in your coursework?", and "What would you like to learn about using AI more effectively or responsibly in your classes?". It then asked them to created a sample prompt and were provided with an AI-generated response to a familiar engineering word problem, which they were asked to evaluate for accuracy. The solution used the correct equations but contained a mathematical error (a common feature of previous AI iterations no longer seen). Students were asked if they thought the solution was accurate, what mistakes (if any) they noticed, and how they would revise or clarify their prompt. Lastly, they were asked how confident they were in their ability to evaluate the accuracy of AI-generated solutions.

Data Analysis

For each participant, their lab report, MATLAB program, and survey responses were combined into one document for analysis. We began by reading the entire data set to immerse ourselves in it. We split the participants so we would each code half of the reports. We used the software Gradescope to create inductive codes through the first round of analysis. After the first round, we met to review the codes, merge similar ones, and discuss their meaning for a similar application. Before beginning the second round, we added a set of a priori codes using the AI literacy framework described above, each corresponding to every level and component of the framework. After the second round, we met to discuss the final themes in the data. Specific focus was given to the prompt lists in their lab report to determine themes in the strategies students used to draft prompts.

Results

Prompt Strategies

To address research question 1 ("What strategies do first-year engineering students use when crafting prompts for generative AI tools?"), the prompt lists included in the reports submitted by the student pairs were analyzed to determine if there were any commonly used strategies by the student participants. Several common strategies emerged, which were grouped into the following categories for discussion: copying and pasting instructions from the problem statement, pasting

in draft code and asking for debugging help, asking for help one line of code at a time, and using Copilot as little as possible.

One of the most prevalent student strategies used on the programming activity was to begin by **copying and pasting the entire problem statement into MATLAB Copilot**. Forty percent (40%) of the student participants discussed trying to copy and paste the whole lab activity into Copilot, or uploading the lab document, even though it was discouraged by the instructor in class. Students who attempted this method expressed frustration in their reflections, as they did not understand the output from Copilot and therefore struggled to make corrections. These students also struggled not to use the specified restricted syntax in their programs. One such student wrote in their reflection,

> *"I found Copilot to be very distracting at most times. It added lines that were doing things I had to research and lines that weren't allowed. It really hindered my progress throughout the week."*

The above student reflection is a clear example of the frustration experienced by some of the participants who tried to generate a large portion of the assignment all at once. It is evident that many of the students lacked AI experience, so their prompts were not specific enough to generate code within the assignment's parameters. Despite having covered the entire curriculum by that point in the semester, it was inevitable that students would encounter unfamiliar coding strategies Copilot used. One of the characteristics of an intermediate level of Effective Prompting & Interaction from the AI Literacy framework is to make prompts more specific by "experimenting with adding constraints." Working within the scope of their coding knowledge would likely have been much easier for these students if they were more comfortable applying this principle. This theme of prompt specificity emerged throughout the analysis of the student reports.

Some students began the assignment using this copy-paste strategy, but gave up and switched to one of the other strategies out of frustration. For example, one pair wrote,

> *"The idea of having an AI companion to provide assistance is amazing, however, the actual experience had a few hiccups. At times Copilot would give out a lot of code, but most of it was unusable. To fix this, we made sure to ask very specific questions to ensure that we got the code and ideas that we needed, instead of a bunch of unusable code."*

The above reflection is an example of a pair of students who began by copying and pasting all the program requirements into Copilot, but changed course due to the "unusable" output from Copilot. Here we see that some of the more successful students realized that if they "made sure to ask very specific questions" the assignment would be made easier.

Similarly, another pair's reflection indicated the same realizations about prompt efficacy after attempting the copy-paste method,

> *"One negative we noticed about Copilot itself is that if you ask it to generate code on its own it is bound to make several mistakes, and only creates more work for you to correct... In order to use Copilot to its full capacity, it is best to pair it with knowledge already known, and to be as descriptive as possible with any question asked. It is best to treat it as an assistant to double check your code, rather than as a program that can do the work for you."*

This reflection not only highlights the value of specifically worded prompts but also addresses another theme that emerged during analysis. Many students discussed the importance of understanding the fundamental coding structures required in the program before attempting to utilize an AI tool.

The second major prompting strategy utilized by this group of students was to use **Copilot as a debugging tool**. This strategy was discussed by twenty-nine percent (29%) of the student participants. These students worked on their code independently, then used Copilot to review their work and explain any errors they encountered. This is the strategy many of the copy-paste students switched to after experiencing frustration, as well as the one implemented by many of the high-performing students in the course. Most of these students expressed satisfaction with their Copilot experience in their reflections. For example, one high-performing student in the course wrote the following reflection:

> *"I thought using Copilot was pretty great. I wanted to try and make the code myself so that I could make sure we didn't use any of the program restrictions, which made the process much easier. Having Copilot create the algorithm helped me to stay on track and keep the loops organized so that the program would run properly. It also made the debugging phase so much faster by telling me exactly what was wrong and where to fix it. Some of the suggestions from AI didn't make sense to me, or they used the restricted features, so it did take some time to ensure that the responses were actually helpful. However, overall I thought the process was made much easier and faster through the use of AI."*

This reflection demonstrates that frustration was seemingly reduced when students attempted to use their own coding skills to complete the assignment, only resorting to Copilot to explain errors. As noted in the reflection, challenges such as avoiding restricted code syntax were still encountered, but were more manageable.

Another strong student pair wrote the following:

> *"Copilot was a lot more helpful at the end of the lab than at the beginning. When giving Copilot the prompts, it was difficult to achieve the result we wanted from it. We found it more useful to write out the code ourselves and then utilize Copilot to explain why we had errors when they popped up. For the most part, we did not use Copilot in writing our actual MATLAB code. Copilot was a useful tool, but there was definitely more value in using it after we had code written rather than trying to write the code with Copilot."*

Again, we see students finding more success using Copilot's debugging power than its large-scale code-generation capabilities. These reflections also revisit the idea that students are more successful in writing the program when they have a solid understanding of coding fundamentals before turning to Copilot.

A third strategy utilized by some of the more successful students was to ask for **help from Copilot one line of code at a time**. Twenty-three (23%) of the participants demonstrated this approach. These students relied more on Copilot than those who simply used it as a debugger, but by breaking the prompts down into smaller segments, they were better able to control the structure of the code and thereby their understanding of the overall program. One such pair wrote,

> *"Copilot...did not work well when given a lot of data and instructions at once. We found it was better to ask step by step what we wanted, in order to minimize errors in the code so we would also then be able to catch mistakes more easily. We also needed to be very specific in the instructions we gave Copilot or else it would get confused and give code that didn't fit the exact parameters..."*

Again, the theme of specific prompts is demonstrated in this reflection. The practice of drafting detailed prompts was especially crucial for students using this method, as they were attempting to write a very specific program one line at a time. Vague prompts would likely result in output that could lead them in the wrong direction. Similarly, another pair whose prompt list reflected the one-line-at-a-time method wrote,

> *"We thought that the Copilot experience was generally helpful as it was able to teach us some necessary code that we otherwise would not have known... Furthermore, on specific questions it answered with multiple versions which made it much easier to build code off. While it did struggle with some code restrictions and keeping a general flow with the existing code, this was not a major problem. All of this could be easily fixed by hand and most likely could have been avoided with more carefully worded questions."*

Students were less frustrated because errors could be corrected more easily one line at a time. Additionally, students were able to use more of the helpful Copilot features (such as several alternative suggestions for a given prompt) because their prompts were more granular. Again, this assignment clearly helped the students realize that more specific prompts produce better results.

The last major category of students was comprised of those who attempted to **use Copilot as little as possible or avoided it altogether**, since the amount of use was not dictated by the assignment. Twenty percent (20%) of the participants fell into this category. These students either felt that they did not need the help of the tool or expressed a resistance to using AI in general after encountering initial frustration. For example, one group reflected,

> *"...we didn't use it too much, and it didn't play a big role in creating our code. It was useful for little suggestions on which function was able to perform which duty or finding a function that was able to perform a specific action we needed. We think Copilot is a very useful tool and could be utilized a lot more in other scenarios, but we did not really need it here."*

The students who wrote the above reflection are strong students who felt the tool was unnecessary, except for getting the lines of code associated with the Arduino. It should be noted that several of the students in this category were also coded as copying and pasting the problem statement into Copilot, but these students abandoned the AI tool when it was not immediately helpful. One such pair wrote,

> *"I don't think we used Copilot for any of our lines of code. We tried to use Copilot a little bit, but it didn't help too much. We ended up figuring it out by ourselves."*

The students who wrote the above reflection submitted a program that did not meet the functionality requirements.

It is interesting to note that students falling in this category seemed to be on the extreme ends of the spectrum; either their coding skills were very strong, and they found the AI tool to be more trouble than it was worth, or their skills were weak, and they likely did not understand much of the Copilot output. This result suggests that a single, uniform approach to AI integration is unlikely to benefit all students equally. Instead, educators may need to adopt differentiated strategies that account for students' prior programming knowledge.

The student participants used a variety of prompting strategies, but based on the prompt lists included in their reflections, their responses to the survey question "How do you create AI prompts?", and the sample prompt they wrote in the survey, very few have surpassed the novice level of Effective Prompting & Interaction in the AI literacy framework. Forty-six percent (46%) were coded as reaching this novice level. Recall that the novice level is defined by "using simple, direct prompts (basic questions or requests) to generate content; recognizing variability in outputs." Many of these students have not yet attained this novice level, as they are still learning that copying and pasting a large problem statement into an AI tool is not an effective approach, or they are avoiding using AI tools altogether. The few students who might be considered to have reached the intermediate level (9%; defined by "applying prompt generation guidelines to refine prompts; applying prompts across different modalities; experimenting with adding constraints and role-based instructions"), broke the problem statement down into many parts and had notably more specific prompts in their prompt list than their peers. These students also made an effort to include any necessary constraints in their prompts, which reduced much of the frustration the other participants were expressing.

Two epiphanies seemed to emerge among students attempting to use the AI tool, regardless of their prompting strategy: more specific prompts yielded better results, and understanding the basics of the required coding structures made the process significantly easier. The activity seems to have been effective in helping students start to notice which prompting practices are more successful.

Evaluation of AI-Generated Output

The second research question ("How do first-year engineering students' evaluations of AI-generated outputs align with criteria in an AI literacy framework?") was examined by analyzing the results of the survey taken by the participants. As discussed above, students were asked to write a sample prompt, and the relevant resulting AI sample output was provided on the student survey. The students were asked to prompt the AI tool to solve a simple engineering problem they were familiar with and could solve by hand. Students were then asked to evaluate the sample output and indicate any errors in the solution.

A significant portion (22%) of the participants were unable to identify an error in the solution or did not identify the error correctly. This indicates that these students are not yet meeting the novice level of Critical Evaluation of AI Outputs/Tools in the AI literacy framework [2]. Given the simplicity of the included mathematical error, this large a portion raises concerns for students ability to evaluate AI output.

Broader Dimensions of AI Literacy in Engineering Education

Research question 3 ("How do these practices relate to broader dimensions of AI literacy in engineering education?") was addressed by looking for common weaknesses and points of frustration that students self-identified in their Copilot experience reflections. Additionally, one of the questions from the survey specifically asked the student participants to describe aspects of AI that they would like to learn more about in their engineering courses. Their responses were analyzed for commonality.

As discussed to some extent above, the results of the programming activity indicate a need to teach effective prompt engineering strategies in early engineering education. The repeated realization by students that prompts need to be more detailed, one of the characteristics of prompts that meet the criteria for intermediate level prompting based on the AI literacy framework, makes it clear that assignments like these should be implemented in more introductory engineering courses.

Additionally, for one survey free response question, "What would you like to learn about using AI more effectively or responsibly in your classes?", seventeen percent of students (17%) responded that they would like to learn to write more effective prompts. The demand for exploring prompt engineering in these early courses is certainly present.

An additional dimension of AI literacy, evident in both the class activity and the survey results, should be further implemented in the classroom: critical AI evaluation. In many of the activity reflections, students reported that having an initial understanding of the fundamental programming skills needed for the assignment made creating the program much easier. They were able to critically evaluate Copilot's output for inaccuracies, restricted code syntax, and inconsistencies if they had this foundational knowledge.

As discussed above, a fairly large number of students were unable to identify the simple error in the sample AI output on the student survey. Some students (31%) expressed a concern about AI tools producing inaccurate output in the survey. An even larger share of students (51%) expressed strong concern that AI would impact their ability to think critically. It is evident that a discussion of strategies for critically analyzing AI output, grounded in the engineering fundamentals these students have been learning in their other coursework, would be valuable for improving their AI literacy. Introductory courses that incorporate AI tools into their curricula should emphasize a healthy balance between AI as a helpful tool and the engineer's responsibility to understand all of their work at a fundamental, theoretical level.

An additional topic that a significant portion of students indicated an interest in learning more about on the student survey was the ethical use of AI tools in their coursework. On the free response question inquiring what students would like to learn more about, approximately ren percent (10%) of students responded that they would like more clarity on when it is appropriate to use AI tools in school and what constitutes cheating with AI. This issue falls squarely into the "Ethical Awareness and Responsibility" dimension of the AI literacy framework. As AI continues to develop and become increasingly used by undergraduate students, it would be valuable to include discussions of ethical use in both academic and professional environments in these early engineering courses.

Conclusions

This study provides empirical insight into how first-year engineering students currently interact with generative AI tools and how those interactions reflect emerging dimensions of AI literacy. By examining students' prompting strategies, their evaluation of AI-generated outputs, and their reflections on AI-supported problem solving, the findings reveal that most first-year students are still operating at or below a novice level in both effective prompting and critical evaluation. While students widely recognize the potential value of AI tools, many lack the skills necessary to use them productively, critically, and confidently within an engineering context.

Across prompting strategies, a consistent pattern emerged: students who attempted to offload large portions of the problem to AI experienced the greatest frustration, confusion, and difficulty complying with assignment constraints. In contrast, students who either used AI primarily for debugging or broke problems into smaller, more specific prompts reported more positive experiences and demonstrated greater control over their learning. These patterns suggest that effective AI use in engineering is tightly coupled with foundational disciplinary knowledge and the ability to articulate clear, well-constrained goals.

Findings related to critical evaluation further underscore the need for explicit instruction. A substantial portion of students were unable to identify a clear error in a familiar engineering problem generated by AI, indicating limited ability to verify correctness even when the task was within their competence. This gap is particularly concerning given students' expressed awareness of AI inaccuracies and their concern about overreliance on AI undermining critical thinking. Together, these results suggest that awareness of AI limitations does not automatically translate into the ability to detect or correct errors without targeted practice and guidance.

The results point to a clear instructional opportunity. Rather than banning AI tools or assuming students will naturally develop effective practices, introductory engineering courses should intentionally integrate AI literacy alongside core technical content. Specifically, instruction should emphasize (1) prompt construction strategies that align with engineering problem decomposition, (2) systematic approaches to evaluating AI output using disciplinary fundamentals, and (3) clear discussions of ethical and responsible AI use. To maintain a reasonable level of content coverage each semester, it will be imperative to redesign our courses so that AI is not "added" as an additional topic, but rather incorporated as a tool to supplement the existing content. Embedding these elements within authentic engineering tasks, rather than in isolated AI-focused courses, may help students see AI as a support for, rather than a replacement of, engineering thinking. Additionally, the introduction of responsibly used AI may make students more open to instructors' use of AI to aid in the grading of this additional, more qualitative coursework.

This study establishes a baseline understanding of first-year engineering students' AI literacy and highlights areas where instructional interventions are most needed. Future work should examine how targeted scaffolding, repeated exposure, and longitudinal integration of AI literacy across the curriculum influence students' prompting competence, evaluative skills, and ethical reasoning over time. As generative AI becomes increasingly embedded in engineering practice, helping students develop critical, reflective, and disciplined approaches to AI use will be essential for supporting both learning and professional formation.

Implications for Instructional Practice

The findings of this study suggest several implications for first-year engineering courses that incorporate or permit the use of generative AI tools. Most notably, the results indicate that students do not naturally develop effective prompting or critical evaluation skills through exposure alone. Instead, these competencies must be explicitly taught, practiced, and reinforced within the context of engineering problem solving.

1. Instruction should explicitly address how to prompt AI tools, rather than assuming students will intuitively learn effective strategies. Many students began by copying and pasting entire problem statements into AI tools, which often resulted in unusable or misleading output and heightened frustration. Instructional interventions should therefore model prompt decomposition as an extension of engineering problem-solving practices. Structured activities in which students iteratively refine prompts and compare AI outputs can help students recognize the relationship between prompt specificity and output quality.
2. The results indicate that students benefit most from using AI as a debugging and explanation tool. Instructors can leverage this finding by explicitly framing AI tools as "assistive partners" for checking, debugging, and explaining, rather than as solution generators. Assignments can be designed to require students to submit both their original work and annotated AI interactions that justify how AI feedback was evaluated, modified, or rejected. This approach reinforces student ownership of solutions while legitimizing productive AI use.
3. The difficulty many students had in identifying errors in AI-generated solutions highlights the need to intentionally teach critical evaluation strategies. Instruction should connect AI evaluation directly to engineering fundamentals already being taught, such as dimensional analysis, boundary checking, algorithm verification, and logical consistency. Embedding short, low-stakes exercises where students critique incorrect or partially correct AI-generated solutions can help students practice identifying errors in a controlled setting. These activities reinforce the idea that AI outputs must always be verified using engineering reasoning, not accepted at face value.
4. The results suggest that AI literacy instruction is most effective when integrated directly into existing coursework rather than treated as a separate or abstract topic. Students' reflections indicated that foundational knowledge in programming and problem structure was essential for meaningful AI use. This suggests that AI instruction should be tightly coupled with technical content, introduced after students have sufficient conceptual grounding to evaluate outputs. For first-year courses, this may mean delaying unrestricted AI use until students have practiced core skills manually, followed by guided AI-supported activities that emphasize comparison, evaluation, and refinement.
5. Student responses reveal a clear need for greater clarity around ethical and responsible AI use. Uncertainty about what constitutes acceptable versus dishonest AI use persists, even among students who frequently use AI tools. Instructors should provide explicit, contextualized guidelines that distinguish between permissible assistance (e.g., debugging explanations, conceptual clarification) and inappropriate substitution of AI-generated work. Incorporating reflective prompts that ask students to justify their AI use decisions can further support ethical awareness and align AI practices with professional engineering standards.

References

[1] M. I. Baig and E. Yadegaridehkordi, "ChatGPT in the higher education: A systematic literature review and research challenges," *Int. J. Educ. Res.*, vol. 127, p. 102411, 2024, doi: 10.1016/j.ijer.2024.102411.

[2] "AI Literacy Framework." Clemson Libraries. [Online]. Available: https://libraries.clemson.edu/ai-tools-literacy/

[3] C. Bego *et al.*, "Working Towards GenAI Literacy: Assessing First-Year Engineering Students' Attitudes towards, Trust in, and Ethical Opinions of ChatGPT," in *2024 ASEE Annual Conference & Exposition Proceedings*, Portland, Oregon: ASEE Conferences, Jun. 2024, p. 48555. doi: 10.18260/1-2--48555.

[4] B. Kennedy, E. Yam, E. Kikuchi, I. Pula, and J. Fuentes, "How Americans View AI and Its Impact on People and Society," Pew Research Center, Washington, D.C., Sep. 2025. [Online]. Available: https://www.pewresearch.org/science/2025/09/17/how-americans-view-ai-and-its-impact-on-people-and-society/

[5] L. Mohandas and N. Mentzer, "Students' Perception and Use of AI Tools in a First-Year Design Thinking Course," in *2024 ASEE Annual Conference & Exposition Proceedings*, Portland, Oregon: ASEE Conferences, Jun. 2024, p. 48025. doi: 10.18260/1-2--48025.

[6] A. Almassaad, H. Alajlan, and R. Alebaikan, "Student Perceptions of Generative Artificial Intelligence: Investigating Utilization, Benefits, and Challenges in Higher Education," *Systems*, vol. 12, no. 385, Sep. 2024, doi: 10.3390/systems12100385.

[7] S. E. Kamali and R. Samsami, "AI in Engineering Curricula: Adoption, Challenges, and Ethical Considerations Among Engineering Students," in *2025 Northeast Section Conference Proceedings*, University of Bridgeport, Bridgeport, CT: ASEE Conferences, Mar. 2025, p. 54997. doi: 10.18260/1-2-1127.1139-54997.

[8] U. Feldman and C. Cherry, "Equipping First-Year Engineering Students with Artificial Intelligence Literacy (AI-L): Implementation, Assessment, and Impact," in *2024 ASEE Annual Conference & Exposition Proceedings*, Portland, Oregon: ASEE Conferences, Jun. 2024, p. 47327. doi: 10.18260/1-2--47327.

[9] S. Siddharth, B. Prince, A. Harsh, and S. Ramachandran, "The World of AI: A Novel Approach to AI Literacy for First-year Engineering Students," *Commun. Comput. Inf. Sci.*, vol. 2591, Jun. 2025, doi: https://doi.org/10.1007/978-3-031-99264-3_31.

[10] F. Benites and C. Battegay, "AI Literacy: Evaluation of an AI Literacy Course for Engineers," in *2025 IEEE Global Engineering Education Conference (EDUCON)*, London, United Kingdom: IEEE, Apr. 2025, pp. 1–10. doi: 10.1109/EDUCON62633.2025.11016470.

[11] J. Delcker, J. Heil, D. Ifenthaler, S. Seufert, and L. Spirgi, "First-year students AI-competence as a predictor for intended and de facto use of AI-tools for supporting learning processes in higher education," *Int. J. Educ. Technol. High. Educ.*, vol. 21, no. 1, p. 18, Mar. 2024, doi: 10.1186/s41239-024-00452-7.

[12] L. S. Lo, "AI Literacy for All: A Universal Framework," 2025, *University of New Mexico Digital Repository*.

[13] M. Hibbert, E. Altman, T. Shippen, and M. Wright, "A Framework for AI Literacy," EDUCAUSE Review - The Voice of the Higher Education Technology Community. [Online]. Available: https://er.educause.edu/articles/2024/6/a-framework-for-ai-literacy