ISSN: 2473-4901
v12 n6692

Evaluating a Socratic AI-Based Learning Activity in Introductory Python Programming

Mark Frydenberg
mfrydenberg@bentley.edu
Bentley University
Waltham, MA, 02452, USA

Sen Li
senli@bentley.edu
Bentley University
Waltham, MA, 02452, USA

Abstract

Generative artificial intelligence (AI) tools are increasingly used in introductory programming courses, yet questions remain about their perceived effectiveness relative to traditional learning resources. This study examines student perceptions of a structured Learn with AI activity designed to guide students in an introductory Python course through programming concepts using AI-generated explanations, code practice, and feedback. Survey data were collected from students regarding their use of course resources and perceptions of their effectiveness. Results indicate that students generally perceived the activity as a valuable support for understanding course material and making progress when encountering difficulties. Comparative analyses of learning resources suggest that the activity was rated as more helpful than several traditional supports, including textbook readings and using search engines, while complementing instructor and peer assistance. Topic-level results also indicate that the activity was particularly helpful for topics such as loops, lists, and dictionaries. These findings suggest that structured AI-based learning activities can play a meaningful role in supporting student learning in introductory programming courses. Rather than replacing existing resources, such tools may be most effective when integrated alongside instructor guidance and opportunities for practice.

Keywords: Socratic Learning, AI-based learning, artificial intelligence, programming, Python

Evaluating a Socratic AI-Based Learning Activity in Introductory Python Programming

Mark Frydenberg and Sen Li

1. INTRODUCTION

Generative artificial intelligence (AI) tools are increasingly used in introductory programming courses for explanation, debugging, code generation, and on-demand support. While these tools offer new opportunities for learning, many uses of AI in programming education remain informal and unstructured, often emphasizing obtaining answers rather than guided reasoning (Amiri & Islam, 2025; Prather et al., 2024). This creates both an opportunity and a concern for educators, who must find new ways to use AI in the programming classroom that extend beyond code generation and model thoughtful, ethical use.

To address this challenge, this study examines Learn with AI, a Socratic learning activity for students in an introductory Python course. In the Learn with AI activity, students provide a generative AI chatbot of their choice with a customized, structured prompt to generate an interactive learning activity related to a course topic. Students complete eight weekly activities throughout the course, each of which presents an overview of a topic using examples based on a student's self-identified interests, followed by three problems that increase in difficulty assessing student understanding of the topic. Rather than providing direct answers, an AI chatbot provides helpful hints and scaffolding to guide students as they work through each question. This interactive Socratic approach invites a new way to interact with a chatbot that is intended to support critical thinking and logical reasoning.

This paper presents Learn with AI as a structured, Socratic AI learning activity in an introductory programming course. This study examines student perceptions of the activity and its helpfulness relative to other learning resources.

2. LITERATURE REVIEW

To understand how AI can function as more than a tool for generating answers, the literature review presents pedagogical approaches that emphasize guided inquiry and active learning.

2.1 Socratic Learning in Introductory Programming Education

The Socratic method, which has its origins in Plato's works, emphasizes guided questioning as a means toward deeper understanding through dialogue (Paul & Elder, 2007). Socratic approaches have also been applied in programming education, where dialogue and guided questioning help students develop a deeper understanding of programming concepts rather than treating code as a "black box." This approach also supports conceptual understanding by using natural language concepts to help learners grasp fundamental programming ideas (El-Zakhem, 2016).

2.2 AI-Supported Learning and Tutoring

The rise of web-based and AI-supported learning tools has enabled several approaches to implementing Socratic learning in programming education. Collaborative coding using tools such as repl.it classroom has shown increased motivation and engagement (Rahman et al., 2020). More recently, adaptive tutorial systems have begun to incorporate Socratic interactions to support learners in completing programming tasks. For example, Lucas et al. (2026) designed an iterative system design in which K-12 students learning Python transitioned from flexible tutorial support to Socratic dialogue, resulting in increased engagement. Additional studies have found that students using Socratic tutoring approaches demonstrate improved understanding of fundamental coding concepts (Alshaikh et al., 2020).

Cornelison (2025) examined an AI-enhanced Socratic teaching approach in AP Computer Science Principles courses, demonstrating that structured, AI-supported inquiry can maintain student

performance while enabling guided exploration. Central to this study is the use of a structured implementation framework that supports inquiry while keeping students focused on core coding concepts and supporting the development of correct reasoning during problem solving.

Other approaches emphasize more direct, on-demand support. For example, Amiri and Islam (2025) describe an AI-based chatbot designed to assist students in Python programming by providing explanations, debugging support, and solutions to common problems. While effective for immediate assistance, this approach emphasizes responsiveness and providing solutions over structured questioning. Similarly, instructional designs that encourage students to use generative AI for exploration and code generation highlight the importance of prompt construction and critical evaluation of outputs but often lack a formalized Socratic interaction model that systematically guides student reasoning.

Several uses of generative AI in programming instruction support interacting with and evaluating AI-generated code, rather than engaging students in a Socratic learning process. For example, some studies use AI tools to generate, interpret, or compare code (Bekkering & Harrington, 2025; McCulloh et al., 2025). Frydenberg et al. (2025) extend this work by asking students to solve a coding problem on their own without AI assistance, and then prompt AI to do the same and compare the results. Results showed that evaluating AI-generated code exposed many novice programming students to programming constructs that were new to them and supported learning through critique and comparison.

Dedicated AI tutoring systems, including Khanmigo and Google Socratic, reflect growing interest in AI-supported, personalized learning (Slijepcevic & Yaylali, 2025; Stefanova & Georgiev, 2026). These learning tools demonstrate the potential for scalable tutoring support, but they often depend on specific platforms and pre-authored content, institutional adoption, or require individual subscriptions. In contrast, prompt-based tutoring approaches can adapt general-purpose chatbots to instructor-defined learning goals and offer flexibility when integrating AI-supported learning activities into coursework.

Even with these advances, much of the existing work in AI-supported programming education still focuses on generating, explaining, or evaluating solutions, rather than structured, inquiry-based interactions that guide student reasoning.

2.3 Structured vs Unstructured Use of AI

Despite the growing use of AI-supported tools in programming education, their effectiveness depends heavily on how students engage with them. Many current implementations emphasize unstructured interaction, where students use AI systems to ask for explanation or code on demand, focusing on how to solve a coding problem rather than how to develop that solution and why it works. While these approaches provide immediate support, they can encourage surface-level engagement and reduce opportunities for deeper reasoning.

In contrast, structured approaches guide students through inquiry-based interactions, prompting them to explain their thinking, reflect on their reasoning, and work through problems step by step. Research suggests that structured prompting can lead to more complete and in-depth responses compared to unstructured interactions (Wang et al., 2024), highlighting the importance of intentional prompt design in shaping learning outcomes.

This distinction highlights a key gap in existing approaches. While AI tools are widely available and used in programming education, less attention has been given to how structured interaction design can shape student learning. This study builds on this distinction by examining a Socratic prompting framework that positions AI as a partner in guiding reasoning rather than simply providing answers.

3. DESIGNING AI AS A SOCRATIC LEARNING PARTNER

While AI is often used as a tool to obtain answers, it can also be used as a cognitive partner that supports the learning process. AI can contribute to scaffolding and adaptive instruction, aligning with themes of constructivism and self-regulated learning (Beishuizen & Steffens, 2011; Fosnot, 2005; Mattar, 2017) and Socratic learning approaches that emphasize guided questioning and inquiry. This section describes considerations when designing a prompt that positions AI as a tool that supports the learner's thinking processes rather than as a tool that provides students with answers. See Figure 1.

ISSN: 2473-4901
v12 n6692

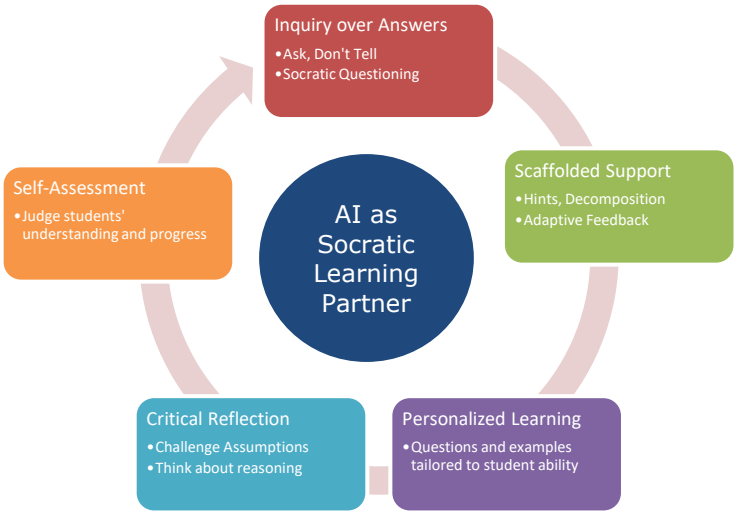


Figure 1. AI as a Socratic Learning Partner (Source: Author).

The activity is designed around five principles. In this sequence, the chatbot first adapts examples to students' interests and learning needs, then uses guided questioning and feedback to support reflection and self-assessment.

Inquiry over Answers. Prompts should encourage learners to interact with the material and discover solutions by asking structured questions rather than offering direct solutions. This approach reflects principles of Socratic learning, where guided questioning supports deeper understanding through dialogue and reflection. Socratic prompting has been shown to develop critical thinking skills and place the learner in a role of creating as well as consuming knowledge (Cornelison, 2025; Sun et al., 2026).

Scaffolded Support. AI can provide hints, explanations, and detailed instructions to learners, dynamically adapting to the learner's needs. Studies have shown that scaffolding improves learning performance in adaptive learning environments (Herold & Seufert, 2025).

Personalized Learning. AI enables personalized learning by tailoring the difficulty, pacing, examples, and responses to the learner's interests and performance (Hang et al., 2024).

Critical Reflection. AI interactions should encourage learners to reflect on their learning, challenge assumptions, and consider how they arrive at their solutions. Encouraging learners to reflect on their work and thought processes promotes critical thinking (Hasanah & Malik, 2019).

Self-Assessment. Instructional designs that include self-assessment encourage learners to identify gaps in their understanding and take ownership of their learning (Simkin, 2015). When paired with reflection, these practices help learners better understand their thinking.

These design principles position AI as an interactive partner in the learning process, supporting inquiry, reflection, and the development of higher-order thinking skills within a technology-assisted environment.

4. METHODOLOGY

4.1 Course Context

The Learn with AI activity was implemented in two sections of CS 230, an introductory Python programming course offered at a business-focused university in New England, taught by two different instructors. A total of 47 students enrolled across both sections; one section was offered in an in-person synchronous format, and the other in an online asynchronous format. The course is standardized, so that despite the difference in instructors and delivery modality, the textbook and major assignments were consistent across all sections. Most students were Finance, Finance and technology, AI, or Data Analytics majors taking the course as a requirement for their degree programs.

The Learn with AI activity was offered to supplement the first eight introductory topics, which included variables, conditional statements, loops, strings, functions, files, lists, and dictionaries. Each Learn with AI activity was assigned as a weekly homework assignment along with two short "practice problems" that developed student Python coding skills in mastering the week's topic.

4.2 Learn with AI Activity

The Learn with AI activity uses the model described in Figure 1 to provide a structured, generalized prompt template for a Socratic lesson. Figure 2 shows a sample prompt.

I am _____, a student in an introductory Python programming course at a business university. Use examples about ____ throughout our lesson, help me to master repetition structures, specifically the while loop, the for loop, and the range() function and using Boolean variables to control while loops, and nested for loops.

Please act as my personal coding tutor. Follow these instructions (in order):

1. The Lesson:
   - Provide a bullet-list summary of the three most important concepts related to for and while loops
   - In a table, show the difference between basic for and while loops.
2. Interactive Practice:
   - Ask me three coding questions, one at a time.
   - The questions should increase in difficulty; the last one should also incorporate an if statement as part of a loop.
   - Use scenarios based on my stated interest.
3. The Socratic Method:
   - If I make a mistake, do not give me the answer.
   - Provide a 'Hint' and a 'Why it matters' explanation, then let me try again.
4. Grade Report and Debrief
   - After I finish all three questions, show me a 'Grade Report' with my name and a score for each question out of 10 points.
   - Show me a 'Pro Tip' for writing cleaner code
5. Constraints:
   - Only use concepts that would appear in a typical introductory Python textbook at this point in the course.
   - Explain any new concepts in a way that is accessible to a beginner.

Please start with the Lesson.

Figure 2. Sample Structured Prompt

To distinguish the Learn with AI activity from unstructured chatbot use, the Socratic tutor is designed as a guided learning interaction that emphasizes adaptive questioning, scaffolded support, and deeper understanding rather than answer delivery. The prompt identifies the role ("I am a student"), context ("in an introductory programming course"), command ("provide three questions in increasing difficulty, do not provide answers, but hints using the Socratic method), and format ("show me a report and a tip") for AI to follow.

The instructor customizes the prompt with information about the week's topics and learning goals and specifies types of questions to ask. Students copy and paste the customized prompt from their course learning management system assignment to the message window of the AI chatbot of their choice (often ChatGPT, CoPilot, Gemini, Grok), personalizing it with their name and a hobby or interest (baseball, flowers, marketing, travel, etc.) from which the chatbot generates examples and questions using themes related to their personal interests. This ensures each student's content is equivalent but different.

Instructions to students warn that AI may ask for a response using a future topic that has not yet been covered in class. If that happens, students may look it up, ask AI for help, or respond that the topic has not been covered in class.

Students were also advised to draft multi-line code responses in a text editor before pasting them into a chatbot, or to upload them as an attached file, because indentation and spacing can affect how Python code is interpreted.

To demonstrate completion, students submit the link to their conversation with the AI chatbot, and a screenshot showing their score on the three coding problems presented. See Appendix B for excerpts from a session using ChatGPT to complete this activity.

4.3 Research Questions

Based on the literature review and the design of the Learn with AI activity, this study addresses the following research questions:

RQ1. How do students perceive the effectiveness of the Learn with AI activity?
RQ2. How do students perceive the helpfulness of the Learn with AI activity compared to that of other learning resources?
RQ3. For which introductory programming topics do students report that the Learn with AI activity is most helpful?

4.4 Survey Data

At the end of the semester, students enrolled in the two course sections voluntarily completed an online survey. The survey was aligned with the study's research questions and asked about perceptions of the Learn with AI activity related to its helpfulness compared with other course resources, frequency of AI chatbot use, and topics for which the activity was most helpful. Additional items addressed student perceptions of ease of use and reliability of AI.

Open-ended questions invited students to describe what the activity did well and how to improve the activity in future offerings. These responses were reviewed to identify representative comments that illustrated students' perceptions of the activity's usefulness, personalization, and support for programming practice. See Appendix A for the complete survey.

5. ANALYSIS AND RESULTS

CS 230 introduced the Learn with AI activity to guide students toward more-structured, inquiry-based engagement with AI tools. The Learn with AI activity was assigned weekly (for eight weeks), and participation rates were high, with 85% of students completing seven or more activities.

Each activity included three questions scored at ten points each; the lowest observed score across all eight weeks was 26 out of 30. Performance was consistently high. Most students received perfect scores,

ISSN: 2473-4901
v12 n6692

with occasional scores of 28 or 29. This pattern suggests that students were able to complete the problems and answer the questions successfully.

5.1 Student Use of AI Resources for Learning

Students were already using AI chatbots for class-related support. Figure 3 provides context for this study by showing that the Learn with AI activity was introduced in an environment where students used AI chatbots multiple times per day.

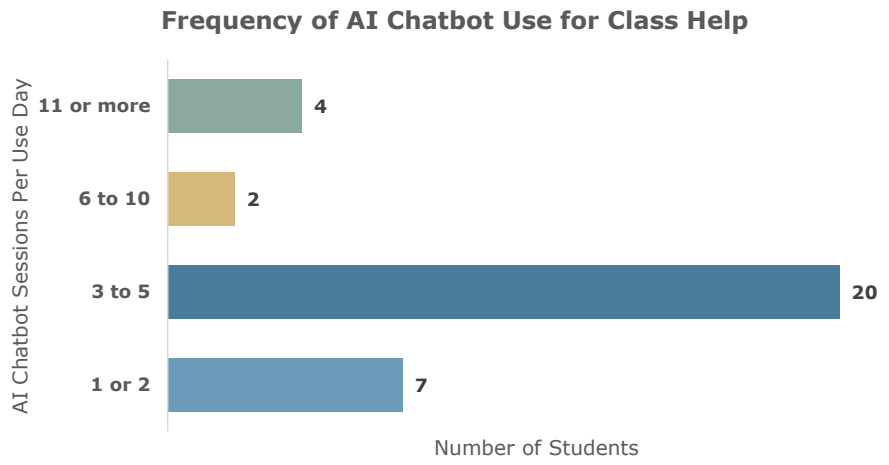


Figure 3. Student Frequency of Daily AI Chatbot Use for Coursework Support (n=33).

Student use of AI Chatbots is frequent but not extreme, with most students using them 3 to 5 times per day. Students were not encountering AI for the first time through the Learn with AI activity; instead, the activity gave structure to a type of support many students were already using informally. The contribution of the Learn with AI activity is not simply to introduce another way to use AI in a programming course, but to do so in a more intentional way that supports critical thinking and problem solving.

5.2 Perceived Effectiveness of Learn with AI Activity (RQ1)

For RQ1, the study uses a 5-item perceived usefulness and acceptance scale adapted from the Technology Acceptance Model (Davis, 1989; Marangunić & Granić, 2015). These items were selected because they focused on students' perceived usefulness, satisfaction, value, and intention to reuse the Learn with AI activity. Specifically, the items addressed whether students perceived that the activity improved their understanding, helped them make progress when stuck, was a valuable use of time, produced a satisfying experience, and would be used again in future courses.

Four additional perception items were analyzed descriptively but not included in this scale because they addressed related but distinct constructs: ease of understanding, fit within the asynchronous course structure, trust in AI feedback, and uncertainty when feedback seemed incorrect or confusing.

The internal consistency of this 5-item perceived usefulness and acceptance scale was assessed using Cronbach's alpha. The analysis was conducted on 34 valid responses with complete data across all items. The scale demonstrated good internal consistency ($\alpha = 0.841$), exceeding the commonly accepted threshold of 0.70 for research purposes (Nunnally & Bernstein, 1994). See Table 1.

ISSN: 2473-4901
v12 n6692

Metric	Value
Cronbach's Alpha	0.841
Valid Responses (n)	34
Number of Items (k)	5
Mean Total Score	21.38 (out of 25 max)
Std. Deviation	3.51
Mean Per Item	4.28 (out of 5)

Table 1. Statistical Metrics on Perceived Usefulness and Acceptance Scale

Figure 4 shows that students reported positive perceptions of the Learn with AI activity, particularly in supporting their understanding and helping them make progress when stuck.

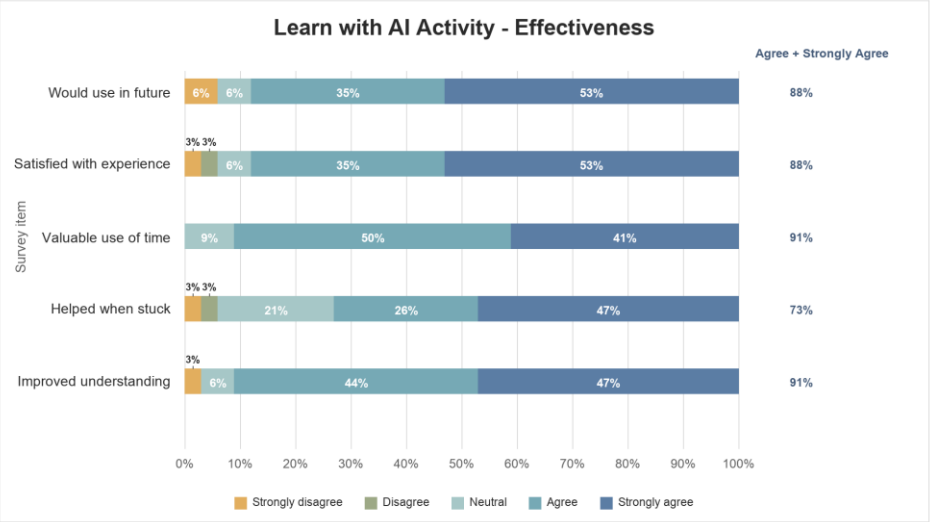


Figure 4. Learn with AI Activity Effectiveness

The strongest results were for improving understanding of weekly topics and being a valuable use of time, with about 91% of respondents selecting *Agree* or *Strongly Agree* for these items. About 88% of the respondents agreed or strongly agreed that they were satisfied with the experience and were willing to participate in a similar activity if offered in other future courses.

Approximately 73% agreed or strongly agreed that the Learn with AI activity helped them make progress when they were stuck, which is favorable. These results suggest that students viewed the activity as an effective and worthwhile support for reinforcing understanding and providing a useful complement to other course resources.

One student commented, "It is giving me a tutor that allows me to personalize my learning but also forces me to practice the code. As the prompt tells it to not give you the answer but hints, it helps you gain that experience and understand the format of that aspect of Python." Another student noted that using a chatbot "allows me to ask questions in a context I best understand."

5.3 Learn with AI Activity Helpfulness Compared with Other Learning Resources (RQ2)

ISSN: 2473-4901
v12 n6692

Figure 5 compares helpfulness ratings of the various learning resources available to CS 230 students. Students rated the Learn with AI activity similarly to other interactive practice activities, including practice coding problems and asking questions of an AI chatbot. These closely grouped ratings suggest that students valued interactive learning resources more highly than textbooks and other passive resources, but the small differences among the top-rated resources should be interpreted with caution. Students in this course rated interactive or practice-oriented resources more highly than textbooks and other passive learning resources, suggesting that they value interactive learning and immediate feedback.

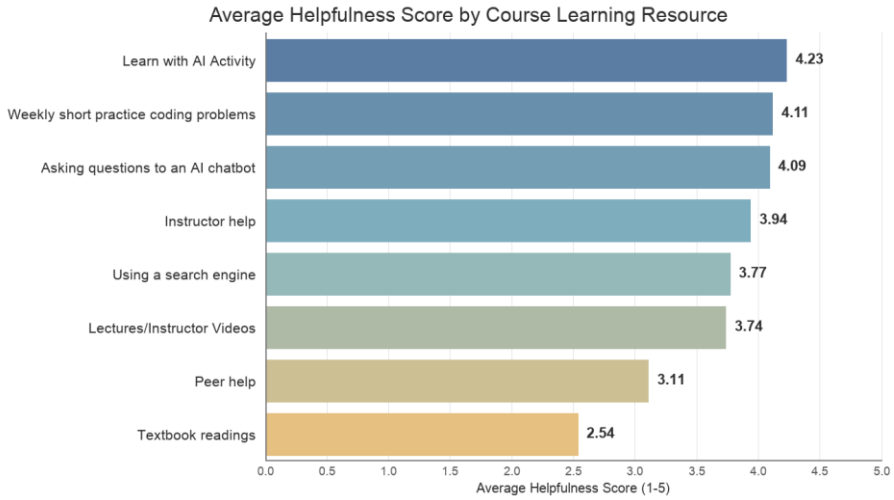


Figure 5. Helpfulness of Learning Resources

5.4 Where AI Helped Most (RQ3)

Figure 6 shows the major course topics and where students found the Learn with AI activity to be most helpful. The activity was most helpful for more advanced topics related to data structures (lists and dictionaries) and less helpful for more basic topics (variables, conditions, strings).

ISSN: 2473-4901
v12 n6692

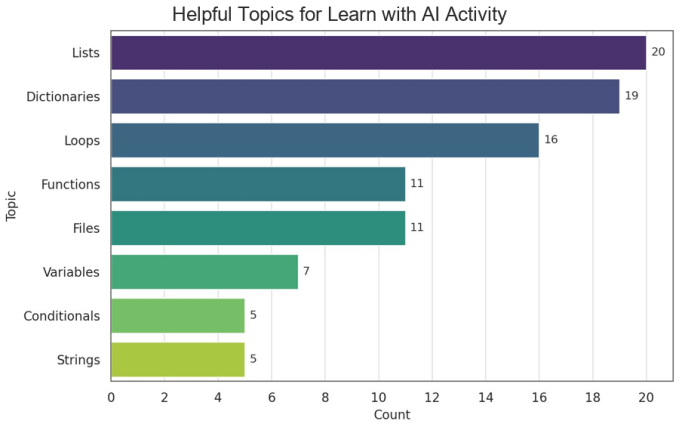


Figure 6. Topics for which Students Reported the Learn with AI Activity to be Most Helpful

Students reported that the Learn with AI activity was most helpful for lists, dictionaries, and loops. These topics typically require students to coordinate multiple programming constructs while reasoning about a problem space: what data must be represented, how it should be organized, and how repeated actions should be applied. In this sense, the activity may have been especially useful when students needed assistance translating a problem scenario or representing data into appropriate Python data structures and control flow. The lower counts for variables, conditionals, and strings may indicate that students found the activity most useful once the course concepts became more complex and practice-intensive.

Student comments also suggest that the activity was of value for the guidance provided to students as they worked through the coding problems. One student said, "It helps you correct small details and is able to explain in very simple terms. Makes it fun and easy to understand."

6. DISCUSSION

The results of this study suggest that a structured, Socratic AI-based learning activity can support students in an introductory programming course. Findings indicate that students had positive perceptions of the activity (RQ1), rated it as one of the most helpful course learning resources (RQ2), and found it especially helpful when learning topics such as lists, dictionaries, and loops (RQ3).

6.1 Why It Worked

One likely reason for the positive perceptions of the Learn with AI activity is that it provided students with a more structured way to interact with AI that they may not have used previously on their own. Rather than using generative AI primarily as a tool to obtain answers, the activity summarized course concepts in a manageable format and then guided students through interactive questions designed to support learning. One student remarked, "I loved how it would fix my mistakes by telling me what I could do to fix them but not giving me the answer. I felt like it was playing the role of a teacher in class."

In this way, AI complemented other learning resources, including human help from the instructor and peers, as well as traditional resources such as lectures and textbooks. Personalized content based on student interests provided different but equivalent questions to each student. The Learn with AI activity provided feedback and guided practice that helped students learn from mistakes and continue working when they were stuck. The activity was designed to encourage students to evaluate their own thinking processes.

6.2 Implications for Teaching

From an instructor's perspective, the activity was relatively easy to implement and grade. Students submitted a link to their chatbot conversation along with a screenshot showing their score on the three problems. Using a consistent prompt template also added familiarity and coherence to the course, while allowing the instructor to customize the topic each week and adapt the activity to other content areas. Repeating the Learn with AI activity across eight weeks may also have helped students recognize its value, especially as the course topics became more difficult. In this way, the activity also modeled ethical AI use by positioning AI as a source of targeted practice rather than answer substitution.

The Learn with AI activity offers a lightweight approach to AI-supported tutoring that does not require a dedicated tutoring platform. It shows students how the chatbots they already use can be tailored with a customized prompt template. This approach gives instructors control over learning goals and reinforces the expectation that AI should provide hints rather than answers, while also giving students content customized to their interests. It offers a scalable approach to individualized instruction while keeping the activity aligned with course topics.

6.3 Limitations

A limitation of this study is that only two sections of an introductory Python course participated in the pilot. As a result, the sample size for this study is small. In addition, because the two sections were taught by different instructors and delivered in different modalities, future analyses should also examine whether student perceptions or activity scores differed by section/instructor. The small sample size limits the strength of such comparisons in the present study.

Future studies can expand the number of sections, courses, or institutions that participate in this or similar Learn with AI activities in Python or other coding courses. Additional implementations across different institutional settings and academic disciplines would also help determine whether the structured Socratic prompting framework scales beyond this introductory programming context.

The study focused on self-reported student perceptions of the Learn with AI activity rather than its effects on learning. Comparing grades or other learning outcomes across sections that used the activity and those that did not would provide a more meaningful control group for assessing its impact on learning as well as student experience and could be an area for future work. Because the Learn with AI activity was a required component in the course, participation levels were very high, limiting the ability to examine differences in outcomes based on level of Learn with AI use.

In this implementation, because students can use the AI chatbot of their choice, no models are trained with data specific to the course materials. Rather, the activity relies on the accuracy and quality of the prompt and the behavior of the underlying large language model (LLM) to present accurate and relevant information.

As AI does not know what students have learned or what topics their classes have covered, the possibility exists that an AI chatbot could present questions or information about some topics include programming constructs which have not yet been introduced in class (e.g., terse if, lambda functions, f-strings). This can create a learning opportunity for students, who may ask an AI chatbot for explanations of unfamiliar constructs.

Finally, the message window in most AI chatbots does not support entering multiline code displayed in monospaced fonts. This is a usability limitation of the current tool environment. To reduce potential formatting errors because spacing and indentation make a difference when entering code, the instructors recommend that students complete the coding questions in a text editor or integrated development environment and then paste their code into the chatbot's message window or upload a text file containing the code for further evaluation.

7. CONCLUSION

This study indicates that students perceived a structured generative AI activity as useful in an introductory programming course. The Learn with AI activity guides students through a process of interactive reading, guided questioning, adaptive scaffolding, and thoughtful self-reflection.

These findings suggest that the effectiveness of AI for learning programming concepts depends less on the tool itself and more on how it is designed and integrated into coursework. Future work could explore how using AI as an interactive learning partner scales across disciplines and how different patterns of AI use influence long-term learning outcomes.

8. REFERENCES

Alshaikh, Z., Tamang, L., & Rus, V. (2020). A Socratic Tutor for Source Code Comprehension. In I. I. Bittencourt, M. Cukurova, K. Muldner, R. Luckin, & E. Millán (Eds.), Artificial Intelligence in Education (pp. 15–19). Springer International Publishing. https://doi.org/10.1007/978-3-030-52240-7_3

Amiri, S. M. H., & Islam, M. M. (2025). Enhancing Python Programming Education with an AI-Powered Code Helper: Design, Implementation, and Impact. Software Engineering, 11(1), 1–17. https://doi.org/10.11648/j.se.20251101.11

Beishuizen, J., & Steffens, K. (2011). A Conceptual Framework for Research on Self-Regulated Learning (pp. 3–19). https://doi.org/10.1007/978-94-6091-654-0_1

Bekkering, T. J., & Harrington, P. (2025). A Comparison of Generative AI Solutions and Textbook Solutions in an Introductory Programming Course. ISEDJ, 23(1), 4. https://isedj.org/2025-23/n1/ISEDJv23n1p4.html

Cornelison, K. (2025, April 8). AI-Enhanced Socratic Method in Computer Science Education: An Implementation Study of Traditional and AI-Guided Learning Approaches. Sciety. https://sciety.org/articles/activity/10.35542/osf.io/uqhe2_v1

Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, 13(3), 319–340. https://doi.org/10.2307/249008

El-Zakhem, I. (2016). Socratic Programming: An Innovative Programming Learning Method. International Journal of Information and Education Technology. https://doi.org/10.7763/IJIET.2016.V6.694

Fosnot, C. T. (Ed.). (2005). Constructivism: Theory, perspectives, and practice (2nd ed). Teachers College Press.

Frydenberg, M., Xu, A., & Xu, J. (2025). Student Perceptions of Learning through Original and AI-Generated Python Programs from a Software Quality Perspective. ISEDJ, 25(4), 34. https://doi.org/https://doi.org/10.62273/DVNQ9288

Hang, C. N., Wei Tan, C., & Yu, P.-D. (2024). MCQGen: A Large Language Model-Driven MCQ Generator for Personalized Learning. IEEE Access, 12, 102261–102273. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3420709

Hasanah, & Malik, M. N. (2019). Implementation of Problem-Based Learning to Improve Critical Thinking Skills in Entrepreneurs Learning. Journal of Physics: Conference Series, 1387, 012037. https://doi.org/10.1088/1742-6596/1387/1/012037

Herold, A., & Seufert, T. (2025). Who regulates? Effects of scaffolding in system- and self-regulated learning. Frontiers in Psychology, 16, 1543024. https://doi.org/10.3389/fpsyg.2025.1543024

Lucas, C., Bihani, A., Kukka, R., Tsai, C.-H., Sarker, J., & Imran, M. M. (2026, May 18). Towards SocratiCode: Designing a Generative AI-Based Programming Tutor for K-12 Students through a 4-Week Participatory Design Study. arXiv.Org. https://arxiv.org/abs/2605.17857v1

Marangunić, N., & Granić, A. (2015). Technology acceptance model: A literature review from 1986 to 2013. Universal Access in the Information Society, 14(1), 81–95. https://doi.org/10.1007/s10209-014-0348-1

Mattar, J. (2017). Constructivism and connectivism in education technology: Active, situated, authentic experiential, and anchored learning. RIED. Revista Iberoamericana de Educación a Distancia, 21(2), 201–217. https://doi.org/10.5944/ried.21.2.20055

McCulloh, I., Rodriguez, P., Kumar, S., Gupta, M., Sharma, V. R., Johnson, B., & Johnson, A. N. (2025, May 24). Generative AI in Computer Science Education: Accelerating Python Learning with ChatGPT. arXiv.Org. https://arxiv.org/abs/2505.20329v1

Nunnally, J. C., & Bernstein, I. H. (1994). Psychometric Theory (3rd Edition). McGraw-Hill. https://archive.org/details/dli.scoerat.1556psychometrictheorysecondedition

Paul, R., & Elder, L. (2007). Critical Thinking: The Art of Socratic - ProQuest. Journal of Developmental Education, 31(1), 36–37. https://www.proquest.com/docview/228487383?sourcetype=Scholarly%20Journals

Prather, J., Denny, P., Leinonen, J., Smith, D. H., Reeves, B. N., MacNeil, S., Becker, B. A., Luxton-Reilly, A., Amarouche, T., & Kimmel, B. (2024). Interactions with Prompt Problems: A New Way to Teach Programming with Large Language Models (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2401.10759

Rahman, M. M., Sharker, M. H., & Paudel, R. (2020). Active and Collaborative Learning Based Dynamic Instructional Approach in Teaching Introductory Computer Science Course with Python Programming. 2020 IEEE Integrated STEM Education Conference (ISEC), 1–7. https://doi.org/10.1109/ISEC49744.2020.9280598

Simkin, M. G. (2015). Should you Allow your Students to Grade their own Homework? Journal of Information Systems Education, 26(2), 147. https://jise.org/Volume26/n2/JISEv26n2p147.html

Slijepcevic, N., & Yaylali, A. (2025). Leveraging "Khanmigo" Generative AI-Powered Tool for Personalized Tutoring to Learn Scientific Concepts. Journal of Teaching and Learning, 19(4). https://doi.org/10.22329/jtl.v19i4.10052

Stefanova, T., & Georgiev, S. (2026). The expanding scope of artificial intelligence integration in mathematics education: A comprehensive review. Mathematics and Education in Mathematics, 55, 242–254. https://doi.org/10.55630/mem.2026.55.242-254

Sun, D., Zheng, Y., Xu, J., & Yang, Z. (2026). When Generative AI Meets Socratic Method: Investigating Programming Learning Dynamics Through Behaviours, Interaction Qualities and Perceptions. Journal of Computer Assisted Learning, 42(2), e70210. https://doi.org/10.1002/jcal.70210

Wang, M., Wang, M., Xu, X., Yang, L., Cai, D., & Yin, M. (2024). Unleashing ChatGPT's Power: A Case Study on Optimizing Information Retrieval in Flipped Classrooms via Prompt Engineering. *IEEE Transactions on Learning Technologies*, *17*, 629–641. https://doi.org/10.1109/TLT.2023.3324714

APPENDIX A

Survey Instrument

Do you wish to participate?

- I agree to participate
- I do not agree to participate

About You

Which section are you in?

- Prof. A.
- Prof. B.

What is your major? (If you have more than one major, choose the one most related to this course.)

▼ Accounting (1) ... Undeclared/Other

What is your minor?

▼ Accounting (1) ... No minor declared

Helpfulness of Learning Resources

Likert Scale:
1 Not at all helpful 2 Slightly helpful 3 Moderately helpful
4 Very helpful 5 Extremely helpful N/A I didn't use this resource

How helpful has each resource been for your learning in this course so far?

- Lectures/Instructor Videos
- Textbook readings
- Learn with AI activity (AI summary + 3 code questions + feedback)
- Weekly short practice coding problems
- Instructor help (office hours, email, chat, etc.)
- Peer help (tutors, friends, etc.)
- Using a search engine (e.g., Google)
- Asking questions to an AI chatbot (e.g., ChatGPT)

Use of AI Tools

- How many of the eight Learn with AI activities did you complete?

None 1 or 2 3 or 4 5 or 6 7 or 8

On days when you used an AI chatbot (such as ChatGPT), about how many separate times did you ask for an explanation, debugging help, or other questions on topics related to this class? (Count a 'time' as a session where you asked one or more questions. If you came back later, that would be a separate time.)

Never used AI for help in this class 1 or 2 3 to 5 6 to 10 11 or more

Perceptions of Learn with AI Activity

Likert Scale:
1 Strongly Disagree 2 Disagree 3 Neutral
4 Agree 5 Strongly Agree N/A I didn't complete this activity

Please indicate how much you agree with the following statements about the Learn with AI activity.

- The Learn with AI activity improved my understanding of weekly topics.
- The activity helped me make progress when I was stuck.
- Overall, the activity was a valuable use of my time.
- The activity was easy to understand (instructions and what to do).
- The activity fit well with the structure of this asynchronous course.
- I generally trusted the activity's explanations and feedback.
- When the feedback seemed incorrect or confusing, it was hard to know what to believe.
- I was satisfied with my experience using the Learn with AI activity.
- If a similar activity was available in future courses, I would choose to use it.

For which topics was the Learn with AI activity most helpful (if you used it)? (Select up to 3)
- ▢ Variables
- ▢ Conditionals
- ▢ Loops
- ▢ Strings
- ▢ Functions
- ▢ Files
- ▢ Lists
- ▢ Dictionaries
- ▢ N/A (I didn't use it)

Rank these from most helpful to your learning (top) to least helpful (bottom):
- _____ Lectures/Instructor Videos
- _____ Textbook readings
- _____ Learn with AI Activity
- _____ Weekly short coding problems (non-AI)
- _____ Instructor help (office hours, email, etc.)
- _____ Peer help (tutors, friends, etc.)
- _____ Using a search engine (e.g., Google)
- _____ Asking questions to an AI chatbot (e.g., ChatGPT)

One thing the Learn with AI activity does well for my learning is… (1–2 sentences)

One change that would make the Learn with AI activity more helpful is… (1–2 sentences)

Which AI chatbot tools did you use to complete the Learn with AI activities?

▢ ChatGPT	▢ Gemini	▢ DeepSeek
▢ Claude	▢ Grok	▢ Meta AI
▢ CoPilot	▢ Perplexity	▢ Other

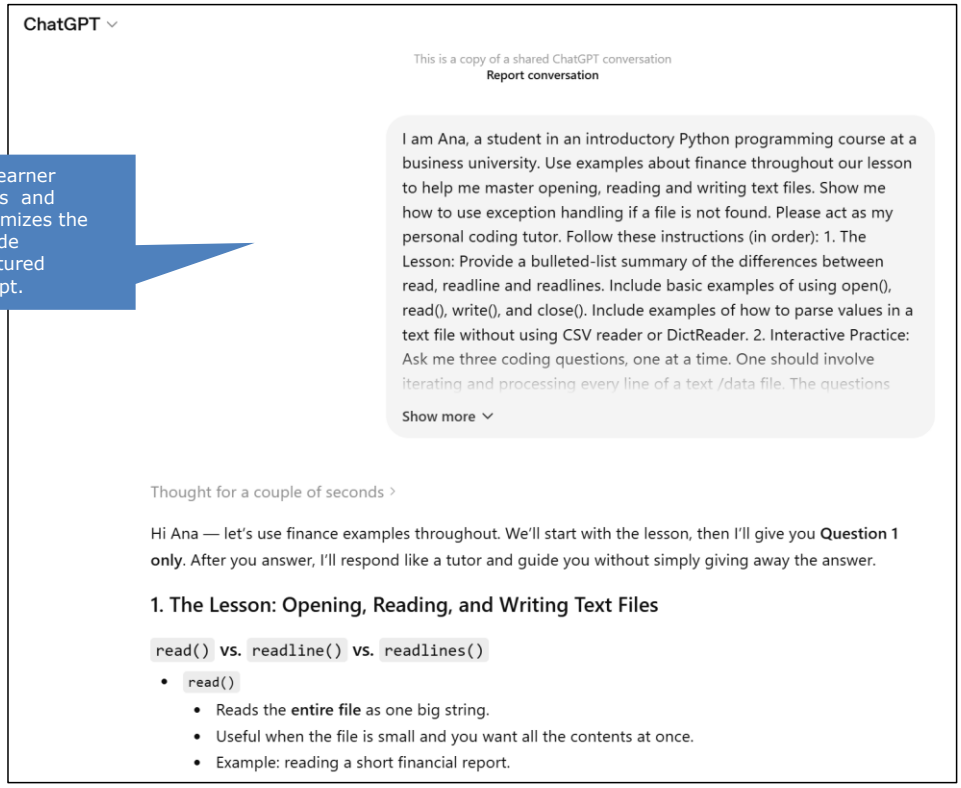
ISSN: 2473-4901
v12 n6692

APPENDIX B

Sample Student Chatbot Interaction

1. Initial Prompt

The learner enters the customized structured prompt for the lesson. AI replies with a welcome message and begins the lesson.

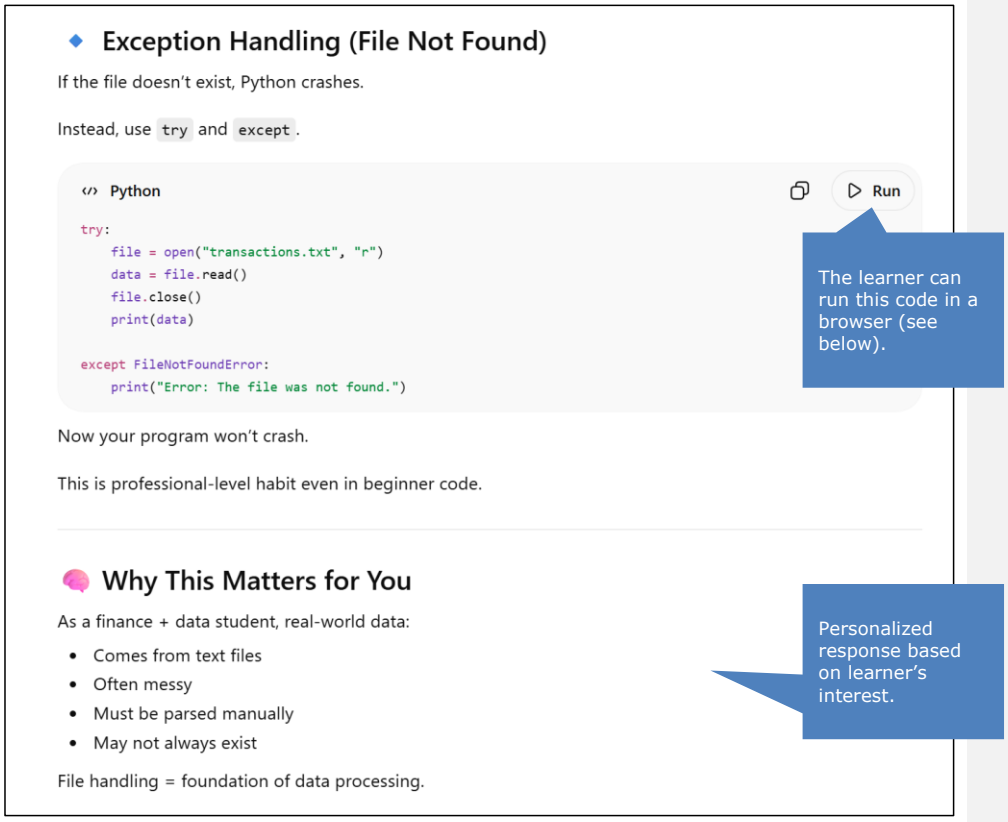


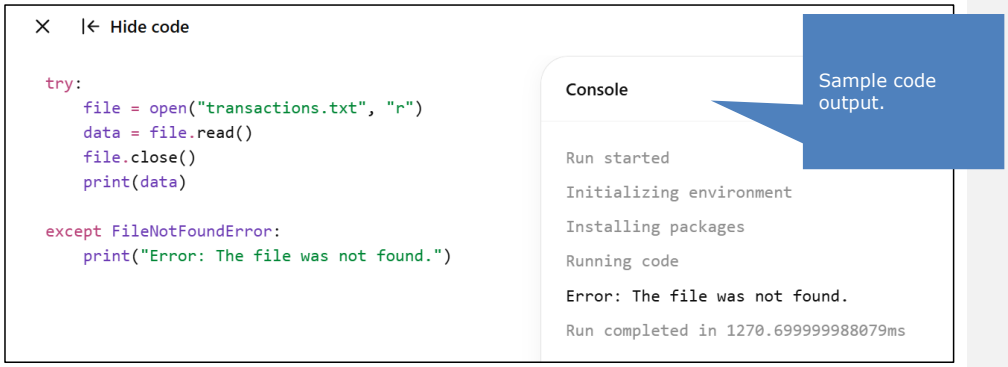
The learner enters and customizes the provide structured prompt.

2. Partial Response from "The Lesson"

This lesson includes several examples including code that the user can run within a browser window or copy and paste into a Python development environment. Since the learner mentioned finance as their interest, AI generates examples using words found in everyday finance discourse.

ISSN: 2473-4901
v12 n6692

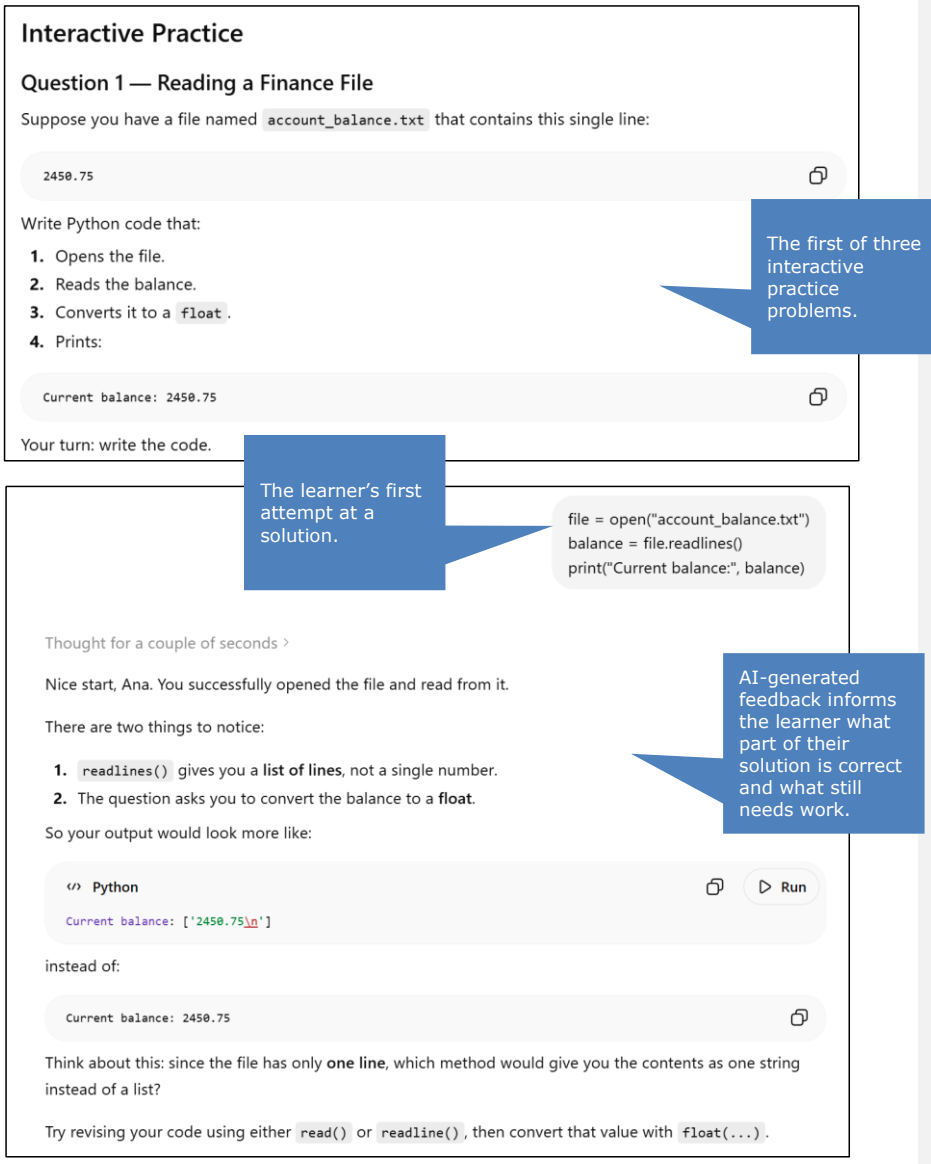




ISSN: 2473-4901
v12 n6692

3. Sample Question

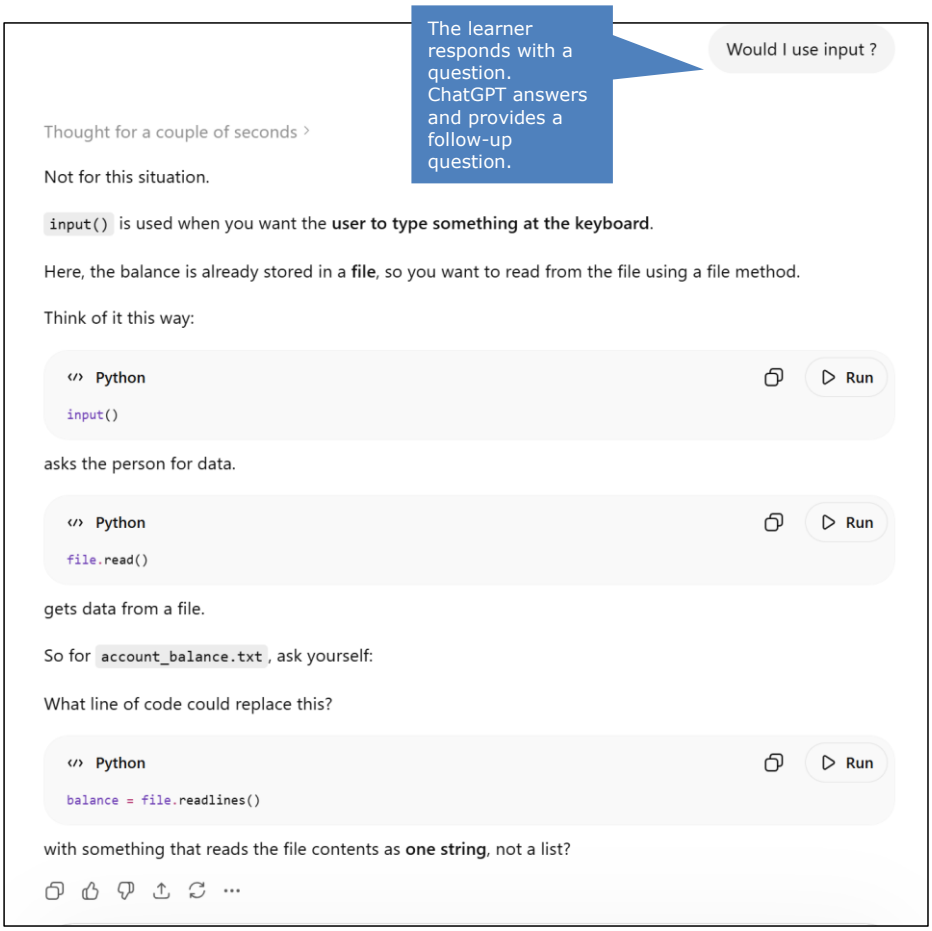
AI asks a question and provides the learner with guidance about incorrect responses.



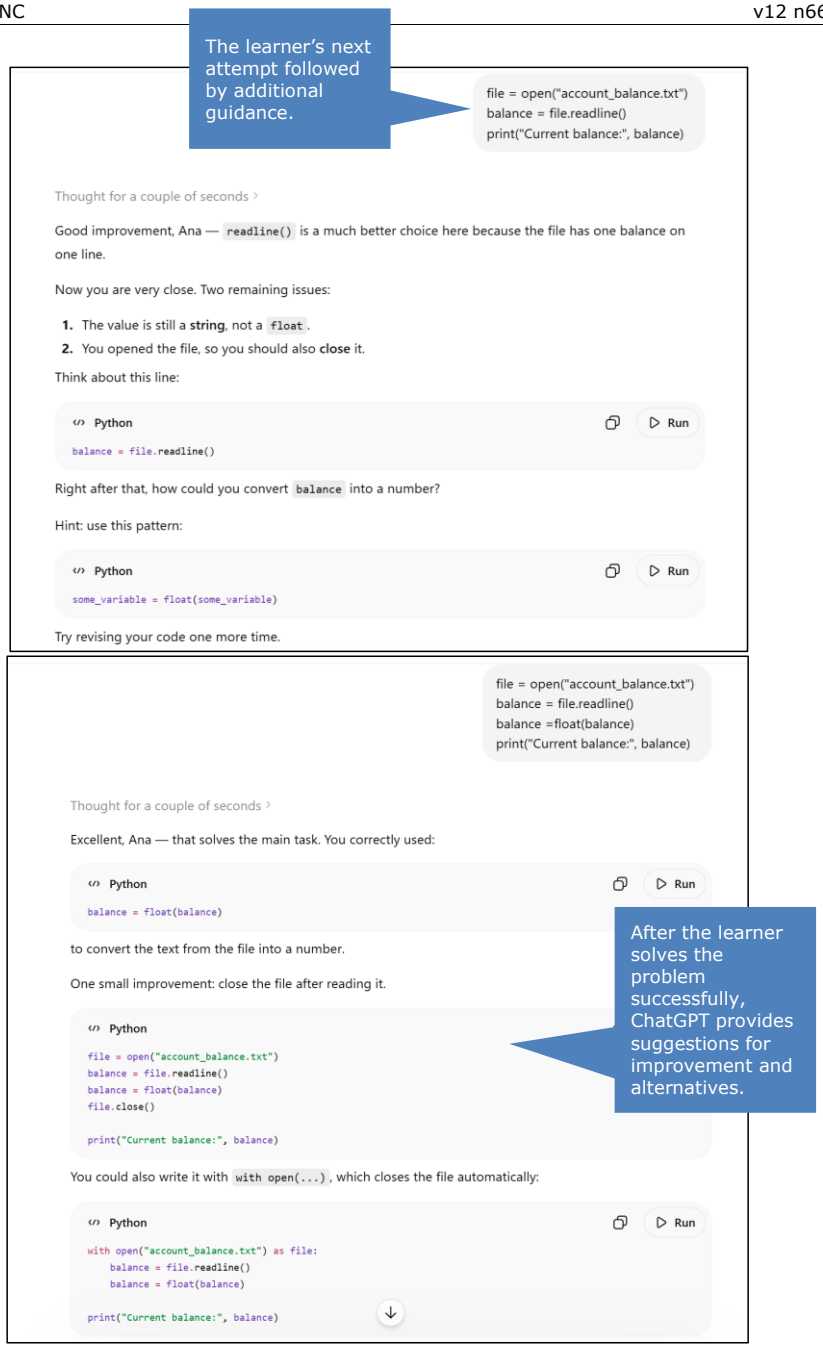
ISSN: 2473-4901
v12 n6692

4. Learner Response and Feedback

The learner interacts with AI and receives personalized feedback and coaching.

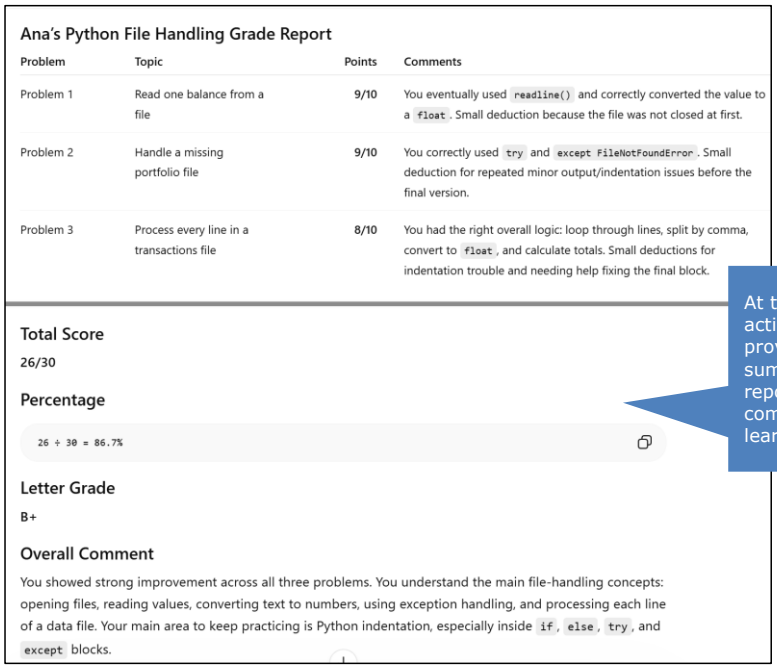


ISSN: 2473-4901
v12 n6692



ISSN: 2473-4901
v12 n6692

5. Report and Tips for Improvement



Ana's Python File Handling Grade Report

Problem	Topic	Points	Comments
Problem 1	Read one balance from a file	9/10	You eventually used `readline()` and correctly converted the value to a `float`. Small deduction because the file was not closed at first.
Problem 2	Handle a missing portfolio file	9/10	You correctly used `try` and `except FileNotFoundError`. Small deduction for repeated minor output/indentation issues before the final version.
Problem 3	Process every line in a transactions file	8/10	You had the right overall logic: loop through lines, split by comma, convert to `float`, and calculate totals. Small deductions for indentation trouble and needing help fixing the final block.

Total Score

26/30

Percentage

```
26 ÷ 30 = 86.7%
```

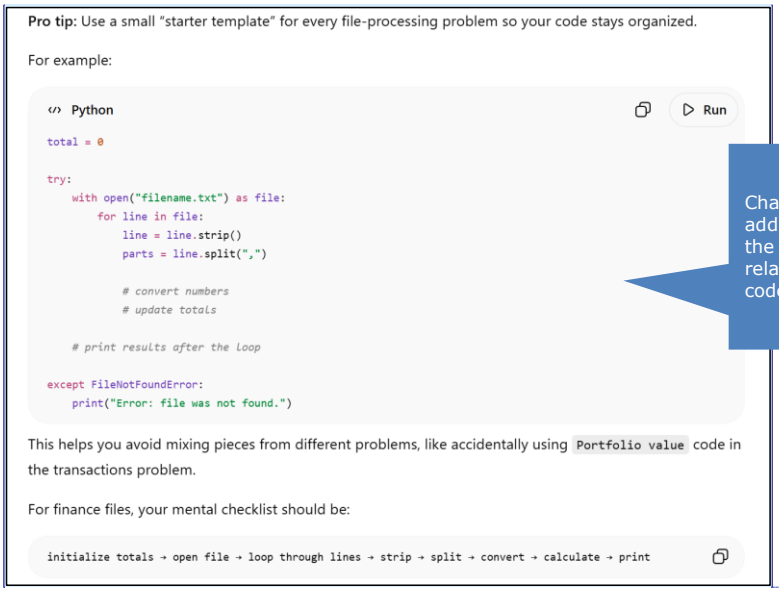
Letter Grade

B+

Overall Comment

You showed strong improvement across all three problems. You understand the main file-handling concepts: opening files, reading values, converting text to numbers, using exception handling, and processing each line of a data file. Your main area to keep practicing is Python indentation, especially inside `if`, `else`, `try`, and `except` blocks.

At the end of the activity, ChatGPT provides a summary grade report and comments to the learner.



Pro tip: Use a small "starter template" for every file-processing problem so your code stays organized.

For example:

Python Run

```
total = 0

try:
    with open("filename.txt") as file:
        for line in file:
            line = line.strip()
            parts = line.split(",")

            # convert numbers
            # update totals

    # print results after the loop

except FileNotFoundError:
    print("Error: file was not found.")
```

This helps you avoid mixing pieces from different problems, like accidentally using `Portfolio value` code in the transactions problem.

For finance files, your mental checklist should be:

```
initialize totals → open file → loop through lines → strip → split → convert → calculate → print
```

ChatGPT provides additional tips for the learner related to their code.

Commented [LS1]: Not sure if it's at my end, but the callout box is partially missing.

Commented [MF2R1]: Just checked I think it's fine.