Jiang, B. et al. (Eds.) (2025). Proceedings of the 33rd International Conference on Computers in Education. Asia-Pacific Society for Computers in Education

An Exploration of the Impact of Using ChatGPT on Students' On-Task Performance and Task Success

Arnel OCAY [ab*] & Maria Mercedes RODRIGO [a]
[a]*Ateneo de Manila University, Philippines*
[b]*Urdaneta City University*
*arnelocay@ucu.edu.ph

Abstract: Learning programming can be difficult, especially for beginners. This study examines the potential of ChatGPT to influence students' programming performance by analyzing self-reported on-task behaviors. Sixty-two first-year IT and CS students were randomly assigned to a control group and an experimental group. Participants of the study completed a hands-on GUI-based programming task using Java Swing. Findings showed significant differences in help-seeking behaviors between groups, with ChatGPT users seeking less help from instructors. While self-reported task success was significantly higher among ChatGPT users, on-task performance metrics like error count and code execution attempts were not associated with task success. These findings suggest that while ChatGPT may support performance on programming tasks, its pedagogical implications require careful consideration to scaffold learning effectively and avoid over-reliance on the tool.

Keywords: ChatGPT, On-Task Performance metrics, Experimental study, Programming performance

1. Introduction

Many students in introductory programming courses, particularly those without prior coding experience, face significant challenges in learning and understanding programming concepts (Javier, 2021; Muntanga et al., 2023). These difficulties extend beyond writing code to include reading, interpreting, and debugging programs (Msane et al., 2020; Thuné & Eckerdal, 2018; Robins, Rountree, & Rountree, 2003). For example, McCracken et al. (2001) found that most first-year students were unable to produce syntactically and logically correct code, while Lister et al. (2004) observed that novice programmers struggled with code comprehension. These recurring issues have been widely recognized (Lahtinen, Ala-Mutka, & Järvinen, 2005; Kaila et al., 2010; Höök & Eckerdal, 2015), often leading students to develop negative perceptions of programming courses (Lai et al., 2022).

To address these challenges, various tools have emerged to support students in acquiring programming skills. One of these is the use of large language models (LLMs), such as ChatGPT, which have shown promise in automating code generation and offering real-time support to novice programmers (White et al., 2023; Takerngsaksiri et al., 2023). ChatGPT, a neural network-based generative AI model, can generate complex responses and has been widely applied in domains like programming and software development (Petrovska et al., 2024; Kadir et al., 2023). Several studies have documented the usefulness of ChatGPT and similar tools in improving productivity and workflow among professional developers (Sauvola et al., 2024; Deniz et al., 2023; Benaich & Hogarth, 2021). These tools are now being adopted in educational settings, particularly in computer science education (Silva et al., 2024), where they have been utilized for code generation (Denny et al., 2024), error correction (Sobania et al., 2023), and syntax assistance.

Despite growing integration of these, there is still limited empirical evidence on how tools like ChatGPT influence novice programmers' problem-solving behaviors, particularly during hands-on programming activities. While several existing studies have focused on

student perceptions (Yilmaz & Yilmaz, 2023; Sun et al., 2024), performance comparisons (Xue et al., 2024; Kozar et al., 2024), or theoretical overviews (Kasneci et al., 2023; Lo, 2023), few have investigated the nature of students' on-task behaviors while using AI assistance in programming. In particular, the relationship between tool usage, task success, and behavioral indicators such as help-seeking, debugging efforts, and persistence remains underexplored.

This study contributes to the discourse on generative AI in computing education by analyzing self-reported on-task performance metrics during a programming task with or without ChatGPT assistance. Focusing on performance behaviors, it examines engagement and interaction strategies through indicators such as completion time, code executions, and error frequencies (Robins et al., 2003). While not direct measures of cognitive gains, these metrics offer valuable proxies for understanding how students navigate programming challenges in real time. This study is part of a threefold research initiative exploring the role of Gen AI in computing education. An overview of these interrelated studies was published in LCNS Augmented Cognition HCII 2025 (Ocay and Rodrigo, 2025).

Specifically, the study aims to answer the following research questions:

RQ1: What is the difference in self-reported on-task performance between students who used ChatGPT and those who did not?

RQ2: What is the relationship between the self-reported on-task performance metrics and the overall self-reported task success of the students?

RQ3: Is there a significant difference in overall self-reported task success between students who used ChatGPT and those who did not?

2. Methods and Procedures

2.1 Participants and Experimental Design

Sixty-four first-year students from two Philippine universities participated in the study. After excluding two students due to incomplete data, 62 participants remained (Male=46, 74.2%; Female=16, 25.8%). All were enrolled in an introductory Java programming course and had no exposure to GUI programming using Swing components.

Participants were randomly assigned to a control group (n=31), which used PDFs, slides, and videos, or an experimental group (n=31), which used ChatGPT. Both groups received a 22-minute video tutorial on Java Swing, then completed a 40-minute programming task, that is to build a basic calculator app with GUI components. The experimental group used the same native ChatGPT version throughout the activity for assistance.

2.2 Measure: On-Task Performance Metrics

To evaluate ChatGPT's impact on students' engagement and outcomes in programming, we adopted on-task performance metrics grounded in prior studies on novice programmers (Robins, Rountree, & Rountree, 2003). Participants self-reported these metrics using a web-based interface that logged task-specific actions during the hands-on session (see Figure 1).

2.2.1 Error Count (ErrorCount)

Error count is an indicator of problem-solving strategy in programming, where a high frequency suggests ineffective or underdeveloped approaches (Lahtinen et al., 2005; Robins et al., 2003; Winslow, 1996). This refers to the total number of errors a student records during the coding task, incremented with each occurrence using the logging system. Only positive values are accepted.

2.2.2 Number of Attempts to Execute the Code (NumAttemptsExecCode)

A common strategy in problem-solving is trial and error (Iyagba, 2020). Often used by novice programmers through repeated code execution attempts (Lister et al., 2004), which reflects iterative refinement leading to solutions (Lahtinen et al., 2005; Jadud, 2006). This metric tracks the frequency a student executes code during the task, with each attempt logged as a single increment.

2.2.3 Number of Times Sought Help from Teachers (NumTimesHelp)

Help-seeking is widely recognized as a sign of engagement and effective problem-solving (Nelson-Le Gall, 1985; Newman & Schwager, 1995). This metric tracks how often a student requested help during the task, with each instance logged as a single increment.

2.2.4 Task Success (TaskSuccess)

A binary metric that captures whether the student completed or solved the programming problem. Students report this by indicating “Yes”, which means the task was completed, or “No”, which means the task was not completed at the end of the programming task.

2.3 Data Collection Procedure

To ensure ethical compliance, the study obtained clearance from the Ateneo Research Ethics Office. Participation was strictly voluntary.

Data on students’ on-task performance was collected using a custom-built web application called the Study User Interface (SUI, see Figure 1), developed with JavaScript, HTML, and CSS, and using Firebase as the backend. This platform allowed students to record performance logs during and after the hands-on task. This includes the on-task performance metrics mentioned in section 2.3.

Figure 1. User Interface for Students during Hands-on Task.

During the 40-minute task, participants interacted with the Student User Interface (SUI) to record task-related actions in real time. Selecting START initiated the session timer, while each error and debugging event was incrementally logged at the moment of occurrence. Upon selecting FINISH, participants provided self-assessment data, including the number of help requests, perceived code quality, and task success. Built-in input validation ensured data consistency and accuracy, thereby minimizing memory bias.

2.4 Statistical Analysis

An independent non-parametric test was used to compare on-task performance metrics between groups. Descriptive statistics, including mean and frequency analysis, were employed to identify group differences. A point-biserial correlation analysis was conducted to assess relationships between task success and each performance metric. Fisher’s exact test was used to examine whether task success rates significantly differed between the experimental and control groups.

3. Discussion of Findings

3.1 (RQ1) Comparison of Self-Reported On-Task Student Performance

A Mann-Whitney U test was conducted to compare self-reported on-task performance between groups. Among the three indicators, only the number of help-seeking instances differed significantly (U=606.50, p=.024), with the control group (M=1.29, SD=2.75) requesting help more frequently than the experimental group (M=0.16, SD=0.37). This suggests that students without AI support relied more on external instructional help, aligning with prior work highlighting novice learners' reliance on external scaffolds (Lister et al., 2004; Vygotsky, 1978).

No significant differences were observed for ErrorCount or NumAttemptsExecCode. However, mean comparisons showed that students using ChatGPT had slightly more executions (M=5.65 vs. M=4.71) and more errors (M=3.81 vs. M=3.16), possibly reflecting more trial-and-error attempts encouraged by immediate AI feedback. While this may indicate iterative engagement, it does not confirm improved understanding. Prior research suggests novice programmers often iterate without grasping underlying logic (Lahtinen et al., 2005; Lister et al., 2004). These findings suggest that ChatGPT may support sustained engagement and reduced instructor dependence but do not confirm cognitive gains or deeper learning.

3.2 (RQ2) Relationship between Self-Reported On-Task Metrics and Task Success

A point-biserial correlation showed that self-reported task success was not significantly related to ErrorCount (rpb=.099, p=.445), NumAttemptsExecCode (rpb=.004, p=.976), or NumTimesHelp (rpb=.056, p=.664). This indicates that the amount of effort or frequency of activity did not directly translate to task success.

However, correlations were found among the behavioral metrics themselves. Students who executed code more often tended to encounter more errors (rpb=.465, p<.001), and those who executed more also sought help more often (rpb=.265, p=.037). These patterns reflect typical novice strategies that use trial-and-error and help-seeking (Lahtinen et al., 2005; Puustinen & Rouet, 2009; Vygotsky, 1978; Lister et al., 2004) in navigating problems and tasks.

These results suggest that while students may demonstrate high activity levels, these do not reliably indicate success. Educational designers may need to design interventions that promote more reflective, strategic problem-solving behaviors rather than relying solely on behavioral persistence, such as trial-and-error strategy.

3.3 (RQ3) Differences in Self-Reported Task Success Rates

A significant difference in task success was observed between groups. Fisher's exact test (p<.001) and chi-square analysis ($\chi^2(1)=18.645$, p<.001) showed that students with ChatGPT reported higher success rates than those without. While encouraging, these are self-reported success outcomes and should not be interpreted as evidence of improved programming competence.

These findings are consistent with prior work highlighting the potential of generative AI tools to support novice learners (Kasneci et al., 2023). However, the increased success may reflect perceived progress or reliance on AI-generated solutions rather than a deeper understanding of the programming concepts.

4. Conclusions

With the growing interest in AI's impact on programming education, this study examined ChatGPT's influence on students' performance behaviors through self-reports. Findings show that ChatGPT users sought less human assistance and executed their code more often, suggesting that AI use may encourage more independent, iterative problem-solving. However, behaviors such as error count, code execution, and help-seeking showed no significant link to

reported task success. This indicates that high engagement does not necessarily translate to deeper conceptual learning.

Students in the experimental group reported higher success rates, which may reflect greater confidence in task completion. While this aligns with industry observations of AI tools improving productivity, it raises important pedagogical questions about overreliance and surface-level learning. As generative AI tools become more accessible in educational contexts, thoughtful integration into programming pedagogy is essential.

5. Limitations and Future Work

This study used self-reported data, which remain prone to bias despite real-time entry. Future work should triangulate these with ChatGPT logs, including chat transcript analysis to analyze student behaviors further, and use pretest and posttest analyses to determine learning gains.

Acknowledgements

We thank Ateneo de Manila University, Ateneo Laboratory for the Learning Sciences, and the Ateneo University Research Office for their support. We also extend our gratitude to all participants and their schools, Pangasinan State University-Urdaneta Campus, and Saint Louis University-Baguio City. Also, our gratitude to Dr. Jennifer Ramos-Miguel for her guidance in the statistical design of this work.

References

Benaich, I., Hogarth, N. (2021) State of AI Report. Accessed at https://www.stateof.ai/2021-report-Launch

Deniz, B. K., Gnanasambandam, C., Harrysson, M., Hussin, A., & Srivastava, S. (2023). Unleashing developer productivity with generative AI. McKinsey Digital.

Denny, P., Prather, J., Becker, B., Finnie-Ansley, J., Hellas, A., Leinonen, J., Luxton-Reilly, A., Reeves, B., Santos, E. A., Sarsa, S. (2024) Computing Education in the Era of Generative AI. Commun. ACM 67, 2, 56–67. https://doi.org/10.1145/3624720

Höök, J., & Eckerdal, A. (2015) On the bimodality in an introductory programming course. An analysis of student performance factors. *In Proceedings of the 2015 Learning and Teaching in Computing and Engineering Conference, Taipei, Taiwan pp. 79-86.* https://doi.org/doi:10.1109/LaTiCE.2015.25

Iyagba, P. W., (2020). Learning and Problem Solving: The Use of Problem-Solving Method To Achieve learning in Pupils. African Social and Educational Journal.

Jadud, M. C. (2006). Methods and tools for exploring novice compilation behavior. *ACM SIGCSE Bulletin, 38(3), 1–5.* https://doi.org/10.1145/1140123.1140124

Javier, B. (2021). Understanding their Voices from Within: Difficulties and Code Comprehension of Life-Long Novice Programmers. *1. 53-76*.

Kadir, E., Rahman, T., & Barman, S. (2023). Exploring the Competency of ChatGPT in Solving Competitive Programming Challenges. Int. J. Adv. Trends Comput. Sci. Eng., vol. 13, no. 1, pp. 13–23, 2023, https://doi.org/doi:10.30534/ijatcse/2024/031312024.

Kaila, E., Rajala, T., Laakso, M., & Salakoski, T. (2010). Effects of course-long use of a program visualization tool. Australasian Computing Education Conference, 97–106. https://doi.org/10.5555/1862219.1862234

Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., …Kasneci, G. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. Learning and Individual Differences, 103, Article 102274.

Kozar, T., Ostojic, D., Liu, Y., Mernik, M. (2024) Computer Science Education in ChatGPT Era: Experiences from Experiment in a Programming Course for Novice Programmers. *Mathematics. 12, 629*.

Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin*, 37(3), 14–18.

Lai, C., Chen, Y., Wang, Y., & Liao, H. (2022). The Study of Learning Computer Programming for Students with Medical Fields of Specification—An Analysis via Structural Equation Modeling. International Journal of Environmental Research and Public Health, 19(10), 6005. https://doi.org/10.3390/ijerph19106005

Lister, R., Adams, E., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, et al. (2004). A multi-national study of reading and tracing skills in novice programmers. *Inroads, 36(4),* 119–150.
Lo, C. K. (2023). What is the impact of ChatGPT on education? A rapid review of the literature. Education Sciences, 13(4), 410.
McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y.B., Laxer, C., Thomas, L., Utting, I., & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *Inroads, 33(4)*, 125–140. https://doi.org/10.1145/572133.572137
Msane, J., Mutanga, B., & Chani, T. (2020). Students' perception of the effect of cognitive factors in determining success in computer programming: a case study. *International Journal of Advanced Computer Science and Applications, 11(7).* https://doi.org/10.14569/ijacsa.2020.0110724M.
Mutanga, M. B., Piyose, P. X., & Ndovela, S. (2023). Factors Affecting Career Preferences and Pathways: Insights from IT Students. *Journal of Information Systems and Informatics, 5(3), 1111–1122.* https://doi.org/10.51519/journalisi.v5i3.556
Nelson-Le Gall, S. (1985). Help-seeking behavior in learning. Review of Research in Education, 12, 55-90.
Newman, R.S., & Schwager, M.T. (1995). Students' help seeking during problem solving: Effects of grade, goal, and prior achievement. American Educational Research Journal, 32, 352-376.
Ocay, A., & Rodrigo, M.M. (2025). A Field Study on the Use of Gen AI to Support Computing Education. In: Schmorrow, D.D., Fidopiastis, C.M. (eds) Augmented Cognition. HCII 2025. Lecture Notes in Computer Science, vol 15778. Springer, Cham. https://doi.org/10.1007/978-3-031-93724-8_17
Petrovska, O., Clift, L., Moller, F., & Pearsall, R. (2024). Incorporating Generative AI into Software Development Education. *ACM. United Kingdom.*
Puustinen, M., & Rouet, J. F. (2009). Learning with new technologies: Help seeking and information searching revisited. Computers & Education, 53(4), 1014–1019. https://doi.org/10.1016/j.compedu.2008.07.002
Robins, A., Rountree, J., & Rountree, N. (2003). Learning and Teaching Programming: A Review and Discussion. *Computer Science Education, 13(2), 137–172.* https://doi.org/10.1076/csed.13.2.137.14200
Sauvola, J., Tarkoma, S., Klemettinen, M., Riekki, J., Doermann, D. (2024). Future of software development with generative AI. *Automated Software Engineering 31:25.* https://doi.org/10.1007/s10515-024-00426-z
Silva, C.A.G.d., Ramos, F.N., de Moraes, R.V., & Santos, E.L.d. (2024). ChatGPT: Challenges and Benefits in Software Programming for Higher Education. *Sustainability 2024, 16,1245.* https://doi.org/10.3390/su16031245
Sobania, D., Briesch, M., Hanna, C., & Petke, J. (2023). An Analysis of the Automatic Bug Fixing Performance of ChatGPT. arXiv:2301.08653.
Sun, D., Boudouaia, A., Zhu, C., Li, Y. (2024) Would ChatGPT-facilitated programming mode impact college students' programming behaviors, performances, and perceptions? An empirical study. *Int J Educ Technol High Educ. 21:14.*
Takerngsaksiri, W., Warusavitarne, C., Yaacoub, C., Hou, M. H. K., & Tantithamthavorn, C. (2023, October 31). Students' perspective on AI code completion: benefits and challenges. arXiv.org. https://arxiv.org/abs/2311.00177
Thuné, M., & Eckerdal, A. (2018). Analysis of Students' learning of computer programming in a computer laboratory context. European Journal of Engineering Education, 44(5), 769–786. https://doi.org/10.1080/03043797.2018.1544609
Vygotsky, L. S. (1978). Mind in society: The development of higher psychological processes. Harvard University Press.
White, J., Hays, S., Fu, Q., Spencer-Smith, J., & Schmidt, D. C. (2023). ChatGPT prompt patterns for improving code quality, refactoring, requirements elicitation, and software design. arXiv.org. https://arxiv.org/abs/2303.07839
Winslow, L. E. (1996). Programming pedagogy—a psychological overview. *SIGCSE Bulletin, 28(3), 17–22.* Retrieved at https://dl.acm.org/doi/pdf/10.1145/234867.234872
Xue, Y., Chen, H., Bai, G., Tairas, R., Huang, Y. (2024). Does ChatGPT Help with Introductory Programming? An Experiment of Students Using ChatGPT in CS1. *Int Conf on Software Engineering. IEEE/ACM.*
Yilmaz, R., Yilmaz, F. G. (2023) The effect of generative artificial intelligence-based tool use on students' computational thinking skills, programming self-efficacy and motivation. *Computers and Education: Artificial Intelligence 4*