SEFI Journal of Engineering Education Advancement
Vol. 1, No. 1, 2024, pp. 39-52, ISSN 3006-6301
DOI : 10.62492/sefijeea.v1i1.20

Exploring the Utilisation of Generative AI Tools by Undergraduate First-Year Mechanical Engineering Students in Programming Assessments

Lama Hamadeh[1]

University College London, London, UK

ABSTRACT

Integrating the fundamentals of computer science and programming skills into the undergraduate engineering curriculum has been a primary focus for many educational institutions worldwide. Learning the basics of programming from the beginning of undergraduate engineering education allows students to incorporate such skills into their future work easily. Therefore, an introductory programming course for first-year undergraduate students has been running since 2021 in the Mechanical Engineering Department at University College London intending to teach the fundamentals of Python programming language. However, it is well-known that generative artificial intelligence (Gen AI) tools in higher education are moving so fast that a wait-and-see approach cannot be taken. These applications have received much global attention from academics on their impact and proper use within the teaching-learning process. This paper investigates first-year undergraduate mechanical engineering students' use of Gen AI tools in their programming assessment. The results show that 60% of the cohort used tools that helped mainly to check their code, improve their English language, and understand error messages. However, 40% abstained from using any. Based on these findings, recommendations on how Gen AI tools can be utilised by undergraduate students in ways that support their learning and enhance their ability to achieve learning outcomes are made

KEYWORDS

Engineering Education, Generative Artificial Intelligence, Programming Assessment, Mechanical Engineering, Undergraduate Students, Python.

PUBLICATION

Submitted:
24th May 2024

Accepted after revision:
2nd August 2024

Introduction

Programming skills provide engineers with an opportunity to integrate innovative technologies into their everyday work and make it more efficient. Even the knowledge of one programming language or understanding of how to work with data gives engineers a substantial advantage in their work. Learning how to code allows students to start thinking like a programmer and improve their problem-solving abilities (Oliphant et al. 2007). After all, both engineering and programming include multiple tries and

[1] Corresponding Author : Lama Hamadeh - l.hamadeh@ucl.ac.uk, Mechanical Engineering Department, Roberts Engineering Building, University College London, London WC1E 7JE

ISSN 3006-6301, Vol. 1 (No. 1) 2024

attempts to develop high-quality results, so with experience, students will start finding solutions for their problems much easier. For the aim of giving rise to highly equipped engineering graduates, teaching these skills needs to be considered and implemented from the very beginning of their undergraduate university journey. For that reason, an introductory programming course has been running for first-year undergraduate students in the Mechanical Engineering Department at University College London since 2021. This course teaches the fundamentals of Python programming language due to its simple syntax that makes it easy to learn for all beginners, its adaptability to analyse and explain most engineering problems, and its popularity within the computational communities, industry, and academia, which greatly helps with students' employability (Hamadeh 2023).

The recent advancements of generative artificial intelligence (Gen AI) and large language models (LLM) in content creation and manipulation have brought significant challenges to higher education teaching and learning in various disciplines. These challenges have called for a transformative change in traditional teaching and assessment strategies to accommodate the latest technological advancements, without compromising the integrity of assessments and evaluations (Saeidlou et al. 2024). Some of the promising use cases of Gen AI tools include automated grading (González-Calatayud 2023; Zachari et al. 2022), personalised learning (Maghsudi et al. 2021; O'Connor 2022; Sallam 2023), generating vignettes as educational sources (Sallam 2023), and interacting with virtual learning assistant. Having said that, as of now, there are no conceptual frameworks that cover the use of Gen AI tools by first-year undergraduate engineering students in their programming assessments. This study is targeting this research gap.

Deciding whether to allow students to use these technologies in their assessments depends primarily on the assessment type and the intended learning outcomes. For instance, for an in-person assessment where students are expected to demonstrate foundational level skills such as remembering, understanding, and independently developing critical thinking skills, it would be impractical for Gen AI tools to be used by students. However, if the assessment aimed to support a particular process such as testing a code, then it would be impractical to completely ban the use of these tools by students in their assessments, as they can be useful in ways that support and optimise – rather than damage– their learning process. A key element of such an approach is communication with students so they are fully aware of what tools to use and in what capacity.

At University College London (UCL), and in the context of assessments, the goal is to ensure that students are using Gen AI tools in ways that support their learning, enhance their ability to achieve their program learning outcomes and prepare them to succeed in their future careers. If Gen AI were misused in assessments, this would be considered under the category of plagiarism and falsification, which ultimately damages students' learning process. For that purpose, UCL has designed a broad three-tiered categorization of AI use in an assessment (UCL 2024). The three categories are not defined rigidly; rather they are a tool for ensuring that for each piece of assessment, staff and students have a shared understanding of whether Gen AI tools can be used and, if they can, how, how much, and where in the assessment process. These categories are:

- Category 1: AI tools cannot be used. The purpose of the format of these assessments makes it inappropriate or impractical for AI tools to be used.
- Category 2: AI tools can be used in an assistive role. Students are permitted to use AI tools for specific defined processes within the assessment.
- Category 3: AI has an integral role. AI can be used as a primary tool throughout the assessment.

Starting this year, UCL policy on academic integrity has required all assessment briefs to provide details on what category the assessment falls into and how Gen AI tools are allowed to be used within the context of the assessment.

This paper investigates the use of Gen AI tools by first-year undergraduate mechanical engineering students in their programming assessment in an assistive role. It examines the different Gen AI tools students use in their work and in what capacity. Moreover, it explores the reasons behind some students' abstaining from using any Gen AI tools although they were permitted to do so. Based on these findings, recommendations on how Gen AI tools can be utilised by students in ways that support their programming learning and enhance their ability to achieve the course learning outcomes are made.

Literature Review

The need for computational thinking and improvements in the programming skills of engineers continues to receive widespread attention to better prepare future graduates (Vaca-Cárdenas et al. 2015; Ball and Zorn 2015). Because of this need, today, basic knowledge of programming is part of almost all engineering program curricula (Damla and Nergiz 2018). For first-year engineering students, assessing these skills could take the form of either an e-assessment using a fully integrated platform into Moodle (Souza et al. 2016), a manual assessment where the assessment is performed manually by the instructor (Dawson-Howe 1996), an automatic assessment which can assess students' solutions automatically (Blumenstein et al. 2008; Blumenstein et al. 2004; Blumenstein et al. 2008), or a Semi-automatic assessment which assesses student's assessments automatically but also require an additional manual inspection from the instructor (Jackson 2000). Although choosing the proper type of assessment is likely to be informed by a broad range of factors such as the intended learning outcomes, the level of study, the disciplinary context, and the delivery mode of the course but it is also a challenge for instructors as programming competencies widely vary amongst first-year engineering students. For instance, the average programming abilities score for first-year students is reported to be only about 23 out of 110 (as cited in Ma et al. 2011). Additionally, in early programming courses, the attrition rate is found to be between 30 and 40 percent, which also shows how students struggle with programming (Beaubouef and Mason 2005). For that reason, it has become important to consider introducing, within certain guidelines, additional assistive tools, such as generative artificial intelligence, that students can use to enhance and support the development of their programming skills.

Generative artificial intelligence (Gen AI) can be defined as "the field of science which studies the (fully) automated construction of intelligence" (Van der Zant et al. 2013). Gen AI involves machine learning and pre-trained large language models based on a large corpus of text data, learning grammar, vocabulary, and various linguistic elements to later generate coherent and contextually relevant human-like content in response to the complex prompts it receives (Salinas-Navarro et al. 2024). Among the specific application tools of Gen AI, ChatGPT 3.5 (Chat Generative Pre-Trained Transformer) is the most famous because it was one of the first tools that were made free and easily accessible online (Stokel-Walker et al. 2023). Nonetheless, an improved paid version (Chat GPT 4) is now available.

In higher education, Gen AI is revolutionising the teaching and learning process, shifting the paradigm towards a more student-centred approach (Grájeda et al. 2023). As the capabilities of Gen AI continue to expand, it is imperative that we critically assess the implications of its integration into higher education (Denecke et al. 2023). Due to its intuitive ease of use, the release of various Gen AI tools quickly ushered in an era of AI use across higher education institutions forcing faculty and administrators to respond to an increase in academic misconduct, especially the use of the application by students for assignments requiring text-based responses (Furze et al. 2024; McDonald et al. 2024).

For that reason, the parallel growth of AI detection software has been noticeable alongside the ongoing development of Gen AI tools. While Turnitin demonstrates a broadly accurate detection rate of Gen AI content, continuous improvements and updates to AI detection algorithms will be necessary to keep

pace with the evolving capabilities of AI models. Having said that, a study by (Perkins et al. 2024) shows that it would be impractical to ban the use of Gen AI tools in assessments and rely solely on detection software. It demonstrates that students will inevitably find ways around any such tools used to detect AI output, so it is likely to be more effective to focus on the adjustment of assessment strategies to discourage AI-generated submissions. Therefore, since the idea of blanket restriction on the use of such applications in higher education seems an unrealistic and impractical approach for certain types of assessments, many universities have reviewed and developed their Gen AI policies to guide more responsible use of the technology (Luo 2024). A notable example is the Gen AI policy guidelines co-designed by 24 Russel Group universities in the UK, which emphasises how universities should support students and staff to become AI-literate while attending to issues of privacy and plagiarism (Russell Group 2023).

Method

I. Structure of the Introductory Programming Course

Introductory courses aim to provide first-year mechanical engineering students with the engineering fundamentals and show how they are applied to basic real-life problems. When it comes to designing an introductory programming course and integrating it into the first-year curriculum, it must go in line with this aim (Sheth et al. 2016). The content, objectives, and the way everything is learned, taught, and assessed, need a lot of thought to be successfully implemented (Sobral 2021). For this introductory programming course, and since no prior programming experience is required or expected, the content and its associated activities should be progressive, i.e., starting from the most fundamental with simpler questions and advancing gradually toward more complex concepts (Gomes and Mendes 2009). To systemize the framework of a best practice, the structure of this element has two main components; teaching and learning classes followed by the assessment, as shown in Figure 1.

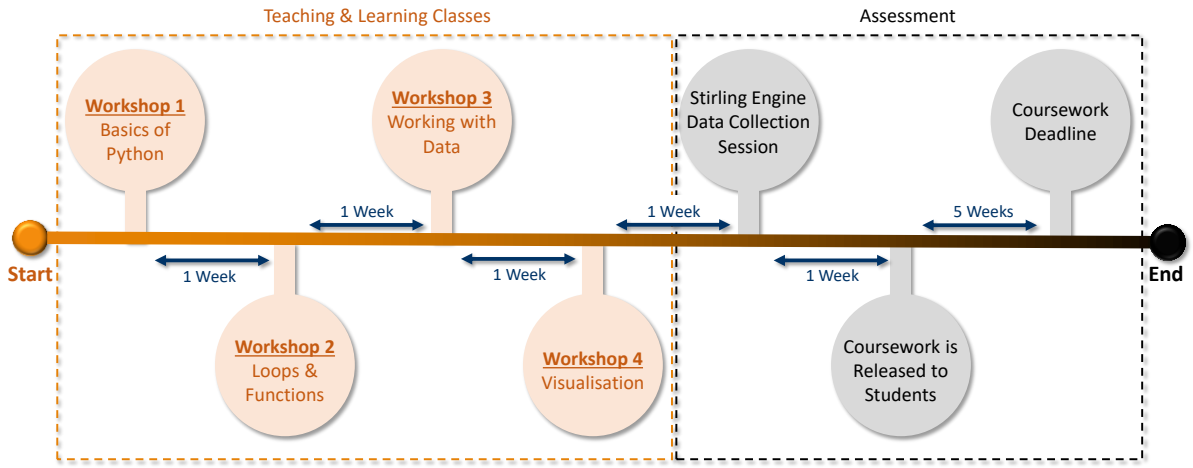

Figure 1: Programming course structure

The teaching and learning sessions of this introductory programming course are structured to be four, 2-hours, in-person, live coding workshops delivered for the entire cohort that has roughly 200 students and span throughout four weeks, i.e., one workshop per week. One major aspect that was considered when structuring these sessions was promoting active and engaging sessions among all students. Classroom climate is an important part of an encouraging experience, especially when students' feedback and questions are respected, students engage more in classes (Crombie et al. 2003; Dallimore et al. 2004). One way to increase classroom interaction in a positive and supportive way is to use postgraduate teaching assistants in large classes (Rissanen 2018). Graduate students are interested in

teaching, and although doctorate programs often do not focus on teaching skills or pedagogical knowledge, studies show (Golde and Dore 2001) that the majority of them seek faculty positions because of their passion for teaching. When postgraduate teaching assistants are present in the classroom with the professor, it helps everyone create a more inclusive and engaging learning environment (Allen and White 1999; Platt et al. 2003). This approach helps the professor to use more innovative teaching methods in a large class, helps the graduate students to gain valuable teaching experience, and helps the students to feel less pressured and engage more actively with the material (Allen and Tanner 2007). Therefore, four postgraduate teaching assistants were recruited for the course's teaching sessions to manage the large cohort number, to make sure students' questions were answered, and to ensure that their learning journey was on track. Another aspect that was considered is the programming abilities and skills variation between students. Some students will enter with a certain programming background, while the majority are novices in this field (Cawthorne 2021). Assuming that all students have no prior programming experience, and to provide programming material that is suitable for all students, the fundamental topics of Python were chosen to be delivered: Python basics, iterative loops, conditionals, functions, working with data, and visualization.

A typical introductory programming course will assess students by having them complete coursework so students are assessed in their ability to apply programming to solve problems (Cawthorn 2021). This course‘s assessment, and since it is directed toward first-year mechanical engineering students, is designed so that it examines and measures students' practical and programming skill sets. In this assessment, students are required to collect a dataset from a 3D-printed Stirling engine, which, as shown in (Figure 2), is a mechanical engineering system that uses cyclic compressions and expansions of a gas at different temperatures to convert heat energy into mechanical work at a certain frequency. The collected dataset, which has the format of a .csv file, includes parameters such as the engine's running time, the upper and lower temperatures, the number of revolutions, etc. Students are required to use this dataset to answer questions presented to them as coursework to evaluate and quantify the physics, dynamics, kinematics, and efficiency of the engine using Jupyter Notebook software and submit it into the online associated submission portal. Jupyter Notebook is an interactive web application for creating and sharing computational documents.

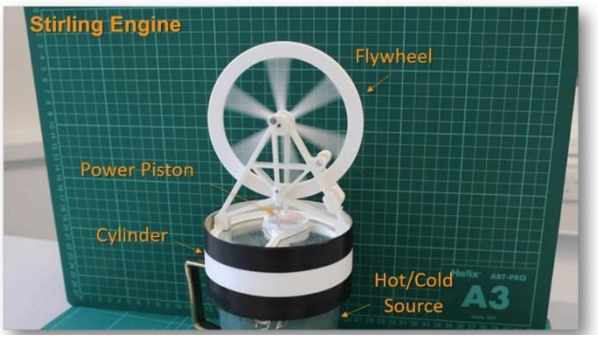

Figure 2: Stirling Engine.

II. Why Allow Students to Use GenAI Tools in their Assessment?

As mentioned before, UCL has developed a three-tiered categorization of AI use in assessments (UCL 2024). This categorization supports staff in designing, marking, and setting assessments, and students to complete assessments in ways that optimise – rather than damage – their learning. During the first two years since this introductory programming course was introduced, Gen AI tools were not allowed to be used by students, or at least it was not mentioned to them explicitly in the assessment brief not to use them. However, starting from this academic year; 2023/2024, and since UCL has required all assessment briefs to state plainly which category the assessment falls into, a decision was to be made

on choosing the proper category for this assessment. Category 2 – where Gen AI tools are used in an assistive role – is chosen based on the following reasons:

1. This course is directed towards beginners where it assumes that students have no prior coding knowledge. This might create a challenge for some students as learning how to code takes time and practice. Having some assistance with proofreading a written code or helping in understanding error messages is a supporting approach to be implemented.
2. The driver of any assessment should be achieving the module's learning outcomes, not the mode of the assessment. Therefore, permitting students to use new tools that would neither hinder the assessment quality nor devalue the course learning outcomes but rather support their knowledge is encouraged.
3. The ban on these tools would produce graduates with little or no knowledge of this ongoing technology, which is embedded now in most of the society sectors.
4. Students will always find a way around tools used to detect AI output no matter how good the used software is.

As a result, an explicit statement was added to the coursework brief informing the students that they can use Gen AI tools in an assistive role, i.e., not to generate code but to proofread it, understand the error messages generated by Python, and to check the grammar/spelling of their English language writing.

Results of Students' Use of Gen AI Tools in Their Programming Assessment

To capture and investigate students' use of Gen AI tools in their programming coursework, students were asked to respond to a 3-minute survey of four questions to be self-completed right after the coursework submission deadline. The survey was anonymous and was done within the scope of UCL Information Security policy (UCL 2024) and according to the ICO's Code of Conduct on Anonymisation (UCL 2024) where all data was saved securely in UCL OneDrive and students were aware of this. No personal data was collected therefore the collected data does not fall within the scope of the GDPR. The questions of the survey are:

1. Did you use any of the Gen AI tools in your coursework?
2. If you answered "Yes" to the previous question, which Gen AI tools did you use?
3. In what capacity did you use the previous tool(s)?
4. If you answered "No" to the previous question, why did not you use any tool?

The total number of students who submitted their coursework was 175, and 143 of them responded to the survey, i.e., 81% of the students completed the questions. Interestingly, 65% of these students used Gen AI tools whereas 34.9% of them abstained from using any. Five main Gen AI tools students used in their coursework, which are; 91.39% ChatGPT, 3.2% Gemini, 3.2% CoPilot, 1% DeepL, and 1% Grammarly. 57.3% of these students used the tools to understand the error message generated by Python language, 35.2% proofread their code, and only 7.3% checked their English writing. On the other hand, 64% of the students who abstained from using Gen AI tools said that these tools were not necessary and the material provided to them was enough, 16% used a simple search, such as Google, 10% felt like it was cheating, and 2% stated that they did not know how to use these tools. (Figure 3) shows a summary of the percentages for both answers.

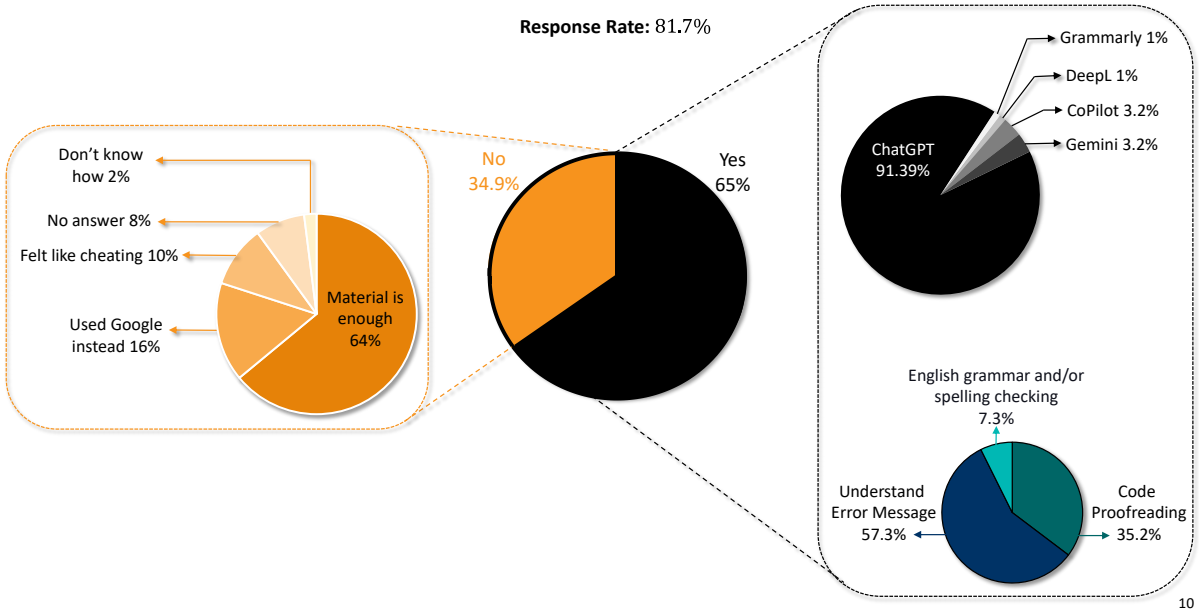

Figure 3 : The results of students' use of Gen AI tools in their programming assessment.

Results

This work qualitatively captured the utilisation of Gen AI tools by first-year undergraduate mechanical engineering students in their programming assessment. It shows that it is essential to listen to students' voices to fill in the gaps left by the fact that most of the conversations on the topic of using Gen AI tools in the teaching and learning process take place between instructors and/or researchers. It is clear from the findings of this work that students have the desire to use and develop their AI skills in their learning journey at university but it is essential to recognise that our job as educators is to regulate this desire by implementing proper guidelines on how and when to use these tools. This section analyses the results captured by the survey, suggests an indicator of whether students followed the permitted limits, identifies the work limitations, and sets several recommendations for future work.

I. Results Analysis

In this introductory programming assessment, it is clear from the results that permitting first-year undergraduate mechanical engineering students to use Gen AI tools has helped the majority of the cohort, 65%, to proofread their code, understand Python error messages, check their written English language, and ultimately submit their assessments satisfactorily and on time. Most of these students, ~57%, used Gen AI tools to understand error messages set off by Python programming language. Since this course delivers the fundamentals of Python language with no complex topics, this percentage indicates that these students had either no prior programming, or Python skills or had difficulties in the programming subject itself (Beaubouef and Mason 2005; Kinnunen 2010; Merriënboer 2003). Allowing them to use Gen AI tools to assist in interpreting these messages provided additional support that helped them rectify these errors and carry on with their assignment. The second highest percentage, ~35%, corresponds to students who needed assistance in code proofreading. This kind of support helped the students to fix semantic bugs and syntax errors and allowed them to focus more on the theoretical and problem-solving aspects of computational thinking (Wing 2006) instead of struggling with the syntax. Moreover, most of our undergraduate students are international students, i.e., non-English speakers. Several studies show that undergraduate international students find it difficult to be fluent in speaking and written English during their first year at university (Littlemore et

al. 2011; Liu 2021). This explains the ~7% of students who used these tools to check their grammar and write a better report.

On the other hand, nearly 35% of the students did not use any Gen AI tools to complete their assessment and the majority of them, 64%, relied only on the course material as they found it enough to understand the problems and write the code that solves them accordingly. It is worth noting here that even though these students felt that there was no need to use any of the Gen AI tools to complete the coursework, adopting Category 1 in this coursework, where Gen AI tools are not permitted, would not be a better approach. This is because these students form a small percentage of the full cohort who might have a higher level of coding competency and hence applying Category 1 would hinder the learning process for the rest of the students who have no prior programming skills.

II. Did students go beyond the permitted limits?

Surely, it would be tempting to know whether students followed the permitted practices given to them in the brief or if they went beyond the permitted limits. However, since there was no reliable method to check this, it would be difficult to know if each student who used Gen AI tools abided by the rules or not. Having said that, a simple comparison to the assessment marks during the past three academic years, i.e., since this programming course was introduced, shows a very similar marks distribution, as shown in (Figure 4). This indicates that using these tools did not change or skewed the marks pattern and hence it can be regarded as an acceptable proof that these tools were used by students as instructed and in an assistive manner.

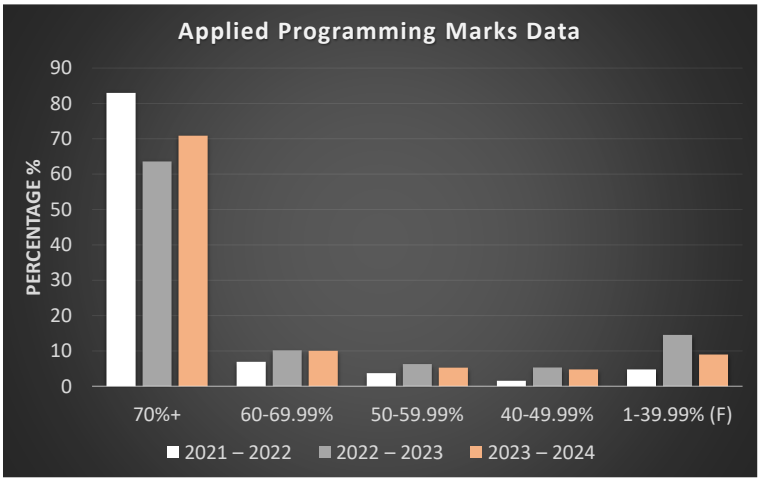

Figure 4: The distribution of the assessment marks during the three academic years of delivering the course, i.e., 2021/2022, 2022/2023, and 2023/2024.

III. Limitations

Although this study was carefully planned and designed, it has some limitations. One limitation is that this study overlooked key factors such as students' familiarity with AI technology, which could significantly affect the adoption and effectiveness of Gen AI tools implementation (Grajeda 2023). Additionally, the design of the questionnaire has some limitations: The questions belong mostly to "yes or no" or very short answers. This is advantageous since students do not find it a big task to do and hence a large response rate is received. However, due to the importance of this topic in higher education nowadays, detailed answers to some questions are needed. For instance, do you have previous experience using these tools? Roughly how much time did you spend on these tools getting responses? Are there any other areas not mentioned in the assessment brief that you would like to add

ISSN 3006-6301, Vol. 1 (No. 1) 2024

so these tools can support you? Testing students' backgrounds at the start of the study can inform the instructor of the students' abilities, and identify weaknesses, or areas needing development (Sieber 2009).

IV. Future Work

The integration of Gen AI in higher education presents immense potential in shaping learning experiences and empowering students and educators. However, harnessing this potential requires collective action and responsible decision-making to ensure the effective and ethical use of Gen AI technologies (Perera 2023). Therefore, it would be essential for future years to provide a foundational introduction and formal guidance either during lecturers or via assessment briefs about what Gen AI tools are, their limitations, possible inaccuracies (Gill et al. 2024), in what capacity students can use them, and the scope of the policy regarding the AI use in assessments imposed by the educational institution. This ensures a complete understanding between instructors and students on what is allowed and what is not (UCL 2024). Moreover, this study was conducted on first-year undergraduate mechanical engineering students. Extending the study to other various engineering fields would help identify the challenges and characteristics influencing the use of Gen AI tools in engineering education within programming assessments. This study can go even further by capturing responses from students from other educational institutions over a longer period of time. Finally yet importantly, monitoring students' utilisation of Gen AI tools in their assessments and confirming that they abide by the given instructions is essential to ensure that their learning process is right on track. Despite its difficulty, several steps can be considered to guarantee that. For instance, checking constantly for improvements and updates of AI detection algorithms to keep pace with the evolving capabilities of AI models used by students. Additionally, students should be asked to mention within their reports whether they used any of these tools, what they are, in which question, and in what capacity. This would form a clear student-instructor communication and an admissible indicator of students' commitment to the rules.

Conclusions

The integration of Gen AI tools in higher education has become a focal point of academic discourse in recent years. In this study, the usability of Gen AI tools by first-year undergraduate mechanical engineering students in their programming assessment was investigated. A qualitative study surveying 143 students on their usage of Gen AI tools was performed to better understand their usage practices and uncover important insights for future implementations. It has been found that the majority of students were motivated to use Gen AI tools as assistants to proofread their code, understand Python error messages, and write better English for their reports. This helped students with no prior knowledge of programming to overcome related difficulties. Additionally, the study shows that some students abstained from using any of these tools in their assessments although they were allowed to do so. This is an expected gap between students' needs, programming skills, and prior foundational knowledge. To facilitate replication of this study, the assessment brief is included in the supplemental materials for this work.

Ethical Statement

The data obtained from students was gathered anonymously and stored in securely. No personal information was obtained and therefore the collected data does not fall within the scope of the GDPR.

Declaration of Interest

The author confirms that this work has no conflict of interest.

Notes on Contributors

The sole contributor to this work is Dr Lama Hamadeh. She is a lecturer in Engineering Mathematics and Computing in the Department of Mechanical Engineering at University College London.

References

Allen D.E. and White H.B., (1999). “A few steps ahead on the same path”. Journal of College Science Teaching, 28, pp. 299–302.

Allen D., and Tanner K., (2007). “Transformations- Approaches to College Science Teaching”. New York, USA: W.H. Freeman and Company.

Ball T. and Zorn B., (2015). “Teach foundational language principles”. Communications of the ACM, 58(5), pp. 30-31 https://doi.org/10.1145/2663342

Beaubouef T., and Mason J., (2005). “Why the high attrition rate for computer science students: Some thoughts and observations”. SIGCSE Bulletin, 37 (2), pp. 103-106. https://doi.org/10.1145/1083431.1083474

Blumenstein M., Green S., Nguyen A., and Muthukkumarasamy V., (2004). “GAME: a generic automated marking environment for programming assessment”. Proceedings of the International Conference on Information Technology: Coding and Computing ser. ITCC 04, pp. 212-216. DOI : https://doi.org/10.1109/ITCC.2004.1286454

Blumenstein M., Green S., Nguyen A., and Muthukkumarasamy V., (2004). “An experimental analysis of GAME: a generic automated marking environment”. SIGCSE Bulletin, pp. 67-71. https://doi.org/10.1145/1007996.1008016

Blumenstein M., Green S., Fogelman S., Nguyen A., and Muthukkumarasamy V., (2008). “Performance analysis of game: A generic automated marking environment”. Computers Education, pp. 1203-1216. https://doi.org/10.1016/j.compedu.2006.11.006

Cawthorn L., (2021). “Invited Viewpoint: Teaching Programming to Students in Physical Sciences and Engineering”. Journal of Materials Science, vol. 56, pp. 16183-16194. https://doi.org/10.1007/s10853-021-06368-1

Crombie G., Pike S.W., Silverthorn N., Jones A., and Piccinin S., (2003). “Students’ perceptions of their classroom participation and instructor as a function of gender and context”. The Journal of Higher Education, 74(1), pp. 51–76. https://doi.org/10.1080/00221546.2003.11777187

Dallimore E. J., Hertenstein J. H., and Platt M. B., (2004). “Classroom participation and discussion effectiveness: Student-generated strategies”. Communication Education, 53, pp. 103–115. https://doi.org/10.1080/0363452032000135805

Topali D, Tagitay NC., (2018). “Improving Programming Skills in Engineering Education Through Problem-Based Game Projects with Scratch”. Computers & Education, Volume 120, pages 64-74. https://doi.org/10.1016/j.compedu.2018.01.011

Dawson-Howe K. M., (1996). "Automatic submission and administration of programming assignments", SIGCSE Bull., pp. 40-42. https://doi.org/10.1145/228296.228303

Denecke K., Glauser R., Reichenpfader D., (2023). “Assessing the Potential and Risks of AI-Based Tools in Higher Education: Results from an eSurvey and SWOT Analysis”. Trends High. Educ. 2, pp. 667-688. https://doi.org/10.3390/higheredu2040039

Furze L., Perkins M., Roe J., and MacVaugh J., (2024). “The AI Assessment Scale (AIAS) in Action: A Pilot Implementation of GenAI Supported Assessment”. Computer and Society, Cornell University, arXiv:2403.14692. https://doi.org/10.48550/arXiv.2403.14692

Gill S. S., Xu M., Patros P., Wu H., Kaur R., Kaur K., Fuller S., Singh M., Rora P., Parlikad A. K., Stankovski V., Abraham A., Ghosh S. K., Lutfiyya H., Kanhere S. S., Bahsoon R., Rana O., Dustdar S., Sakellariou R., and Buyya R., (2024). “Transformative effects of ChatGPT on modern education: Emerging era of A.I. Chatbots”. Internet of Things and Cyber-Physical Systems, 4, 19–23. https://doi.org/10.48550/arXiv.2203.04159

Golde C.M., and Dore T.M., (2001). “At cross purpose: what experiences of doctoral students reveal about doctoral education. Philadelphia: Pew Charitable Trusts”. Retrieved from http://www.phdcompletion.org/promising/Golde.pdf. Accessed on 1st August 2024.

Gomes A., and Mendes A., (2009). “Bloom’s Taxonomy based approach to learn basic programming”. In G. Siemens & C. Fulford (Eds), Proceedings of ED-MEDIA 2009-World Conference on Educational Multimedia, Hypermedia & Telecommunications (pp. 2547 – 2554).

González-Calatayud V., Prendes-Espinosa P., and Roig-Vila R., (2021). “Artificial intelligence for student assessment: A systematic review”. Appl. Sci., 11, 5467. https://doi.org/10.3390/app11125467

Grájeda A., Burgos J., Córdova P., and Sanjinés A. (2023). “Assessing student-perceived impact of using artificial intelligence tools: Construction of a synthetic index of application in higher education”. Cogent Education, 11(1). https://doi.org/10.1080/2331186X.2023.2287917

Hamadeh L., (2023). “Integrating Python into Mechanical Engineering Undergraduate Curriculum”. 9th International Research Symposium on Problem Based Learning (IRSPBL), Aalborg Universitestsforlag, pp. 191-198. https://vbn.aau.dk/en/publications/transforming-engineering-education.

Jackson D., (2000). “A semi-automated approach to online assessment”. SIGCSE Bull., pp. 164-167. https://doi.org/10.1145/353519.343160

Kinnunen P., and Simon B., (2010). “Experiencing programming assignments in CS1: the emotional toll”. In Proceedings of the Sixth international workshop on Computing education research, pp. 77–86. https://doi.org/10.1145/1839594.1839609

Littlemore J., Chen P. T., Koester A., and Barnden J., (2011). “Difficulties in Metaphor Comprehension Faced by International Students whose First Language is not English”. Applied Linguistics, 32(4), pp. 408–429. https://doi.org/10.1093/applin/amr009

Liu J., Hu S., and Pascarella E. T., (2021). "Are non-native English speaking students disadvantaged in college experiences and cognitive outcomes?" Journal of Diversity in Higher Education, 14(3), pp. 398–407. https://psycnet.apa.org/doi/10.1037/dhe0000164

Luo J., (2024). "A critical review of GenAI policies in higher education assessment: a call to reconsider the "originality" of students' work". Assessment Evaluation in Higher Education, pp. 1-14. https://doi.org/10.1080/02602938.2024.2309963

Ma L., Ferguson J., Roper M., and Wood M., (2011). "Investigating and improving the models of programming concepts held by the novice programmers". Computer Science Education, 21 (1), pp. 57-80. https://doi.org/10.1080/08993408.2011.554722

Maghsudi S., Lan A., Xu J., and van Der Schaar M., (2021). "Personalized education in the artificial intelligence era: What to expect next". IEEE Signal Process. Mag., 38, pp. 37–50. DOI:10.1109/MSP.2021.3055032

McDonald N., Johri A., Ali A., and Hingle A., (2024). "Generative Artificial Intelligence in Higher Education: Evidence from an Analysis of Institutional Policies and Guidelines". Computer and Society, Cornell University, https://doi.org/10.48550/arXiv.2402.01659

Merriënboer J. J. V., Kirschner P. A., and Kester L., (2003). "Taking the load off a learner's mind: Instructional design for complex learning". Educational psychologist 38, 1, pp. 5–13. https://doi.org/10.1207/S15326985EP3801_2

O'Connor S., (2022). "Open artificial intelligence platforms in nursing education: Tools for academic progress or abuse?" Nurse Educ. Pract., 66, 103537. DOI: 10.1016/j.nepr.2022.103537

Oliphant T. E., (2007). "Python for Scientific Computing". Computing in Science & Engineering, 9(3), pp. 10-20. DOI: 10.1109/MCSE.2007.58

Perera P., (2023). "Preparing to Revolutionize Education with the Multi-Model GenAI Tool Google Gemini? A Journey towards Effective Policy Making". CQUniversity. Journal contribution. DOI:10.36348/jaep.2023.v07i08.001

Perkins M., Roe J., Postma D., McGaughran J., and Hickerson D., (2024). "Detection of GPT-4 Generated Text in Higher Education: Combining Academic Judgement and Software to Identify Generative AI Tool Misuse". J Acad Ethics 22, pp. 89–113. https://doi.org/10.1007/s10805-023-09492-6

Platt T., Barber E., Yoshinaka A., and Roth V., (2003). "An innovative selection and training program for problem-based learning workshop leaders in biochemistry". Biochemistry Molecular Biology Education, 31, pp. 132–136. https://doi.org/10.1002/bmb.2003.494031020171

Rissanen A., (2018). "Student Engagement in Large Classroom: the Effect on Grades, Attendance and Student Experiences in an Undergraduate Biology Course". Can J Sci Math Techn 18, pp. 136–153. https://doi.org/10.1007/s42330-018-0015-2

Russell Group. (2023). New principles on use of AI in education. https://russellgroup.ac.uk/news/new-principles-on-use-of-ai-in-education/.

Saeidlou S., Nortcliffe A., Ghadiminia N., and Imam A., (2024). "Integrating AI into Engineering Education: Leveraging CDIO for Enhancing Assessment Strategies". Proceedings of the 20th

International CDIO Conference, hosted by Ecole Supérieure Privée d'Ingénierie et de Technologies (ESPRIT) Tunis, Tunisia.

Salinas-Navarro D.E., Vilalta-Perdomo E., Michel-Villarreal R., and Montesinos L., (2024). "Using Generative Artificial Intelligence Tools to Explain and Enhance Experiential Learning for Authentic Assessment". Education Sciences 14(1), 83. DOI : https://doi.org/10.3390/educsci14010083

Sallam M., (2023). "ChatGPT Utility in Healthcare Education, Research, and Practice: Systematic Review on the Promising Perspectives and Valid Concerns". Healthcare, 11, 887. DOI : 10.3390/healthcare11060887

Sheth S., Murphy C., Ross K., and Shasha D., (2016). "A course on programming and problem solving". Proceedings of the 47th ACM Technical Symposium on Computing Science Education – SIGCSE '16. https://doi.org/10.1145/2839509.2844594

Sieber V., (2009). "Diagnostic online assessment of basic IT skills in 1st-year undergraduates in the Medical Sciences Division, University of Oxford". British Journal of Educational Technology, 40(2), pp. 215-226. https://doi.org/10.1111/j.1467-8535.2008.00926.x

Sobral S. R., (2021). "Bloom's Taxonomy to improve teaching-leaning in introduction to programming". International Journal of Information and Education Technology, 11(3), 148-153. DOI: https://doi.org/10.18178/ijiet.2021.11.3.1504

Souza D. M., Felizardo K. R., and Barbosa E. F., (2016). "A Systematic Literature Review of Assessment Tools for Programming Assignments".IEEE 29th International Conference on Software Engineering Education and Training (CSEET), Dallas, TX, USA, pp. 147-156. DOI: 10.1109/CSEET.2016.48

Stokel-Walker C., and Van Noorden R., (2023). "What ChatGPT and Generative AI Mean for Science". Nature, 614, pp. 214–216. DOI: 10.1038/d41586-023-00340-6

UCL (2024), "Using AI tools in assessment", https://www.ucl.ac.uk/teaching-learning/generative-ai-hub/using-ai-tools-assessment. Accessed on 1st August 2024.

Information Commissioners Office (2024), "Anonymisation and Pseudonymisation of Personal Data: ICO's Code of Conduct on Anonymisation", https://ico.org.uk/media/1061/anonymisation-code.pdf. Accessed on 1st August 2024.

UCL (2024), "Policy on Information Security and Management", https://www.ucl.ac.uk/information-security. Accessed on 1st August 2024.

Vaca-Cárdenas L.A., Bertacchini F., Tavernise A., Gabriele L., Valenti A., Olmedo D.E., and Bilotta E., (2015). "Coding with Scratch: The design of an educational setting for Elementary pre-service teachers Interactive collaborative learning (ICL)". international conference on, IEEE (2015, September), pp. 1171-1177 DOI:10.1109/ICL.2015.7318200

Van der Zant T., Kouw M., and Schomaker L., (2013). "Generative Artificial Intelligence. In Philosophy and Theory of Artificial Intelligence; Studies in Applied Philosophy, Epistemology and Rational Ethics". Springer: Berlin/Heidelberg, Germany, pp. 107–120. http://dx.doi.org/10.1007/978-3-642-31674-6_8

Wing J. M., (2006). "Computational thinking". Commun. ACM 49, 3, pp. 33–35.

ISSN 3006-6301, Vol. 1 (No. 1) 2024

Zachari S., Hassan K., Guanliang C., Roberto M., Jason M. L., Sandra M., Neil S., and Dragan G., (2022). “Assessment in the age of artificial intelligence”. Computers and Education: Artificial Intelligence, 3(100075). https://doi.org/10.1016/j.caeai.2022.100075