ISSN: 2560-5429 Journal of Applied Technical and Educational Sciences http://jates.org/
Vol. 16, No. 1: 418 https://doi.org/10.24368/jates418 1-25

Comparing Gamified and Traditional Assessment Environments: A Quasi-Experimental Study in a University Python Course

József Cserkó [1,2*]

[1] *Assistant Professor – John von Neumann University, Izsáki str. 12, Kecskemét 6000, Hungary, cserko.jozsef@nje.hu*

[2*] *PhD student - Multidisciplinary Doctoral School of Engineering, Egyetem sqr. 1, Győr 9026, Hungary, j.cserko@gmail.com*

Abstract

This study examines student-performance outcomes by comparing two distinct assessment environments—a traditional paper-based exam and a complex gamified digital format—in a university-level introductory Python programming course. A quasi-experimental comparison with student self-selection was conducted at John von Neumann University (Hungary) with 63 first-year Information Technology students. Twenty-seven students took a conventional paper-based exam, while 36 completed the assessment in CodingUs, a custom-built "Among Us"-inspired web application. This gamified condition operated as a package intervention, incorporating not only game design elements but also individualized AI-generated tasks, disabled clipboard operations, and a distinct user interface. Isomorphic Python tasks were produced by an AI-assisted generation pipeline using GPT-4o-mini and GPT-4o. Performance was compared using the Mann–Whitney U test as the primary procedure, with an independent-samples t-test as a supplementary parametric analysis. The two groups did not differ significantly in mean performance (gamified: M = 63.06%, SD = 31.61; traditional: M = 68.89%, SD = 34.68; Mann–Whitney U = 423.50, p = .383; t(61) = −0.70, p = .490; Cohen's d = −0.18; 95% CI for the mean difference [−22.61, +10.94]). While no statistically significant difference in performance was detected in this sample, the wide confidence interval and the self-selection nature of the design preclude claims of equivalence. Informal classroom observations and unsolicited student feedback offered preliminary indications of elevated engagement and favourable perceptions of the anti-cheating provisions in the gamified cohort; because no validated self-report instrument was administered, these impressions are reported as exploratory rather than confirmatory. The study contributes a replicable AI-supported pipeline for generating isomorphic programming items and motivates further research employing randomised allocation and validated measurement instruments.

Keywords: Gamification; evaluation; exam; measurement; university.

1. Introduction

Higher education assessment is under pressure to keep pace with students who grew up in interactive, game-influenced digital environments. While gamification—the systematic use of game design elements, for example points, badges, and leaderboards, in non-game contexts (Deterding et al., 2011)—has been shown to boost motivation during learning activities (Sailer et al., 2017; Hamari et al., 2014; Lampropoulos & Sidiropoulos, 2024), its role in the evaluation phase (tests, quizzes, and exams) remains underexplored (Ulloa & Carcausto, 2024). Parallel developments in examination technology—such as adaptive scoring systems driven by fuzzy

logic and machine learning (Karl et al., 2025)—underscore the growing interest in digitally enhanced assessment, yet direct empirical comparisons between traditional and technology-augmented formats remain scarce. Most empirical studies stop at classroom engagement metrics and do not measure whether gamified assessments can produce grades that are as reliable and fair as traditional written exams.

This article addresses that gap by comparing learning outcomes and student experience across two fundamentally different assessment environments in a university-level programming course, rather than attempting to isolate gamification as a single variable. Sixty-three first-year Information Technology students at John von Neumann University (Hungary) completed Python programming tasks delivered either through a standard paper-and-pencil format or through CodingUs, a web-based game that awards coins, levels, and badges for correct solutions. Because the gamified condition also required students to navigate a different interface, manually type code without copy-paste functionality, and solve individualized AI-generated tasks, it is evaluated here as a complex package intervention rather than a pure test of gamification. Large-language-model tools (GPT-4o-mini and GPT-4o) were used to generate and automatically validate hundreds of isomorphic coding challenges (Shoaib et al., 2025; Sarsa et al., 2022) intended to provide comparable item difficulty across conditions, subject to the limitations of the task-equivalence procedure discussed in the Discussion section.

2. Background and Literature Review

2.1. Gamification in higher education

Gamification is most widely defined as the use of game design elements in non-game contexts (Deterding et al., 2011), a framing that has since been adopted across the broader educational-technology literature. Hamari (2019) broadens this conceptualisation by positioning gamification as a socio-technical phenomenon in which game-like feedback structures are intentionally grafted onto activities that are not themselves games. Indeed, the reach of gamification extends well beyond education; Lindsay (2011), for example, demonstrated its application to cultural preservation through documentary games. In educational settings, Kapp (2012) argues that the purposeful alignment of game mechanics, aesthetics, and narrative with instructional objectives is what distinguishes gamified learning from mere entertainment, and that poorly aligned implementations tend to produce short-lived novelty effects rather than sustained learning gains. Practitioner-oriented treatments such as Zichermann and Cunningham (2011) further emphasise that the selection and sequencing of mechanics must be tailored to the behavioural objectives of the target domain. Ruhi (2015) further proposes a descriptive

framework that separates structural gamification (the addition of point systems, levels, and leaderboards to an otherwise unchanged activity) from content gamification (the redesign of the activity itself around game-like goals and feedback loops); this distinction has proven useful in subsequent empirical work because the two approaches tend to yield different motivational and performance profiles.

Empirical syntheses generally report small-to-moderate positive effects of gamification on motivation and engagement in higher education, but with considerable heterogeneity across studies. The seminal systematic review by Hamari, Koivisto and Sarsa (2014) found mostly positive outcomes in the first generation of empirical studies, while also highlighting that effects depended heavily on context, target population, and the specific game elements used. More recent studies reinforce this picture: Lampropoulos and Sidiropoulos (2024) in a longitudinal study comparing online, traditional, and gamified learning found that gamified students achieved learning outcomes at least comparable to those of their peers in the conventional conditions. Zourmpakis, Kalogiannakis and Papadakis (2024) examined adaptive gamification in science learning and observed that the strongest effects emerged when difficulty adaptation matched learner profiles, suggesting that personalisation is a key moderator of gamification's impact.

Field-experimental and cross-context evidence adds nuance to these aggregate findings. Mirmotahari, Grun and Berg (2025) argued that gamification can act as a transferable pedagogical innovation for technology education, provided that it is designed to support collaborative and skill-oriented outcomes rather than isolated task completion. In a systematic review focused specifically on assessment, Ulloa and Carcausto (2024) noted that although terminology and implementations vary, a small set of elements (points, levels, challenges, and badges) recurs across studies, and their impact depends strongly on the alignment between gamified mechanics and the underlying learning objectives.

2.2. Gamified assessment and summative evaluation

Although the broader gamification literature is now substantial, empirical work specifically targeting graded, summative assessment remains comparatively thin. In their systematic review of gamified educational assessment, Ulloa and Carcausto (2024) observed that the majority of included studies concerned formative activities, quizzes, or low-stakes classroom exercises, and that high-stakes summative contexts were represented by only a handful of empirical reports. Mirmotahari, Grun and Berg (2025) explicitly warn that findings from formative gamification do not transfer unchanged to summative settings, because the constraints imposed by validity, fairness, and academic integrity requirements are qualitatively different.

Several methodological questions therefore remain open. First, it is unclear whether gamified assessment introduces construct-irrelevant variance through sources such as platform familiarity, user-interface overhead, or motivational self-selection. Second, the few published summative studies rarely report standard psychometric indicators such as reliability coefficients, item-level difficulty, or measurement invariance across formats, which makes cross-study comparison difficult. Third, the interaction between gamified formats and academic integrity provisions has been examined mainly in descriptive or design-oriented terms rather than in controlled comparisons. These gaps motivate comparative designs in which a gamified summative format is evaluated side-by-side with a conventional one within the same course and the same curriculum.

2.3. AI-assisted generation of programming assessment items

A parallel literature examines the use of large language models (LLMs) for generating programming exercises, solutions, and associated test suites. This line of work sits within the broader integration of artificial intelligence into educational practice (Molnár & Nagy, 2025), which has accelerated rapidly over the past half-decade. The systematic literature review by Shoaib et al. (2025) documents the rapid growth of this field and identifies recurring technical and demographic patterns: most reported systems rely on general-purpose commercial LLMs, most evaluations are short-term, and hallucinations or non-executable code remain a persistent source of error. Sarsa et al. (2022) were among the first to systematically evaluate LLM-generated programming exercises and explanations in computing education; they reported that while the quality of generated items was often acceptable, expert review was still required to guarantee correctness and pedagogical alignment, particularly for items intended to measure specific competencies. Fakhoury et al. (2024) investigated an interactive, test-driven loop in which an LLM iteratively generates and refines code against unit tests, and reported that the availability of a machine-checkable correctness oracle substantially improved the reliability of the generated artefacts.

Complementary work addresses feedback and prompt design in automated programming assessment systems. Frankford et al. (2025) surveyed the space of feedback types provided by such systems and identified a clear trend toward richer, adaptive, and LLM-driven feedback that goes beyond binary pass/fail signals. Serra and Oliveira (2025) examined how prompt engineering can transform existing educational digital resources into more engaging learning experiences, framing prompt design as an instructional-design activity in its own right. At a more practical level, earlier work on dataset construction for AI pipelines in education (Molnár, Cserkó, Nagy, & Balogh, 2023) has highlighted the non-trivial cost of preparing language-

specific training and validation material, a cost that persists even when modern LLMs are used as a backbone. Taken together, this literature establishes both the feasibility and the remaining limitations of AI-assisted item generation for programming assessment, and motivates pipelines that couple generation with automated correctness checks.

2.4. Academic integrity in digital and AI-era assessment

The widespread availability of generative AI has qualitatively changed the academic integrity landscape in higher education. Lancaster (2023) argues that commodity LLMs have blurred the traditional boundary between legitimate writing support and unattributed content generation, because paraphrased or freshly generated text leaves few of the textual fingerprints on which conventional similarity-based detection relies. Lodge (2024) extends this analysis by describing generative AI as an evolving rather than static risk, and by critically examining the turn toward surveillance-based proctoring technologies; he notes that webcam monitoring, screen recording, and behavioural analytics have attracted sustained criticism on grounds of intrusiveness, potential for bias, student anxiety, and limited demonstrated effectiveness. Earlier practitioner-oriented work on automated plagiarism checking by Molnár and Cserkó (2022) illustrates both the feasibility and the structural limits of similarity-based detection, which is ill-suited to content generated de novo by modern language models.

In response to these limitations, a growing strand of the literature advocates design-based rather than purely detection-based approaches to integrity. Rather than attempting to catch dishonest behaviour after the fact, this perspective emphasises individualised items, in-session constraints on tool use, and assessment environments in which honest engagement is the path of least resistance. Broader frameworks for behaviour change support this perspective: the capability–opportunity–motivation model proposed by Michie, van Stralen and West (2011) similarly suggests that restructuring environmental opportunities is more effective than relying solely on post-hoc detection. The design choices reported in the present study's Materials and Methods section are situated within this tradition.

2.5. Theoretical framework: Self-Determination Theory

The theoretical framework adopted here is Self-Determination Theory (SDT), developed by Ryan and Deci (2000). SDT posits that sustained, high-quality motivation depends on the satisfaction of three basic psychological needs: autonomy (a sense of volitional rather than coerced engagement), competence (a sense of progressive mastery and effective action), and relatedness (a sense of social connection to others). SDT has been extensively validated in

educational research, where the satisfaction of these needs predicts academic persistence, engagement, and well-being across a wide range of subject areas and learner populations.

Sailer, Hense, Mayr and Mandl (2017) extended SDT into the gamification literature through an experimental study that linked individual game elements to specific needs. They reported that elements such as points, performance graphs, and leaderboards primarily support the competence need, whereas elements such as avatars, meaningful stories, and shared cultural frames preferentially support relatedness; autonomy, in turn, is supported less by any single element and more by the overall framing of participation as a voluntary challenge. Complementing SDT, Nakamura and Csikszentmihalyi (2002) describe the concept of flow as an optimal experiential state that arises when task difficulty is well matched to current skill level and when feedback on progress is available in near-real time. Recent syntheses confirm that flow remains a central construct in gameful research: Oliveira and Hamari (2025) identify a steady increase in empirical studies linking flow to gameful environments, and Abrache and Oubahssi (2024) provide controlled experimental evidence that gamified conditions can elevate students' self-reported flow relative to non-gamified baselines. Flow theory and SDT are compatible rather than competing accounts, and together they provide the theoretical lens through which the design decisions and the observed performance patterns of the present study are interpreted in the Discussion.

2.6. Research gap and positioning of the present study

The preceding body of literature demonstrates that gamification in higher education has been studied extensively in terms of motivation, engagement, and perceived enjoyment, typically in the context of formative learning activities or low-stakes classroom exercises. Comparatively few studies address the use of gamified environments as the primary delivery format for summative, graded examinations, and even fewer combine such a comparison with an AI-assisted item-generation pipeline designed to preserve task equivalence across conditions. The present study is intended to contribute to this under-explored intersection by reporting a quasi-experimental comparison of a bespoke gamified Python examination platform (CodingUs) against a conventional paper-based examination within the same course, with explicit attention to both the statistical comparability of student performance and the methodological limitations (self-selection, absence of a validated psychometric analysis of task difficulty, and partial crossover between conditions) that such a comparison inevitably entails.

3. Research Goal and Research Questions

The study pursues two research questions:

1. RQ1: Does a gamified exam yield student performance (mean score, median, and variance) comparable to a traditional exam?

2. RQ2 (exploratory): What preliminary indications, based on informal classroom observations and unsolicited student feedback, emerge regarding student engagement, perceived fairness, and the deterrence of dishonest behaviour in the two assessment formats? RQ2 is framed as an exploratory rather than confirmatory research question because no validated self-report instrument (e.g., a Self-Determination Theory questionnaire, an academic-integrity perception scale, or a standardised engagement inventory) was administered in the present study. Any findings reported in response to RQ2 should therefore be understood as descriptive, hypothesis-generating observations intended to motivate subsequent, more rigorously instrumented investigations.

4. Materials and Methods

4.1. Gamification concept for exams

Adapting gamification to summative examination contexts is potentially valuable but imposes substantially stricter design requirements than its use in formative or low-stakes activities. Three principles guided the implementation described below.

The first rule is that participants must be comfortable with the measuring system before the exam. When we prepared the measurement, we ran tests on students who did not know the gamified system. Students complained about the game mechanisms, the menu system of the game, and other aspects because they did not have any experience with the application. Therefore, they had already used the gamified system during their labour. We practiced for 10 to 15 minutes in every classroom in the gamified exam system.

The second rule is freedom of choice. We allowed the students to choose between gamified and traditional classroom practices. Based on our experience, the students who participated in the research do not have any preconceptions or negative opinions against the gamified methods. The university age group is keen to try new things.

The third and most important design principle concerns the comparability of task difficulty across the two assessment formats. Because interpreting between-group performance differences requires items of similar difficulty, the study relied on three complementary strategies to approximate structural comparability, though true equivalence was assumed rather

than empirically demonstrated. First, all items in both formats were drawn from the same curricular content specification, which defined the set of Python language features, control structures, data-handling operations, and algorithmic patterns covered in the course. Second, in the gamified condition, individual items were produced through the AI-assisted generation pipeline described in the Technical Validation subsection, in which GPT-4o-mini drafts isomorphic variants of seed problems and GPT-4o performs an automated sanity check on the generated solution and associated unit tests; the pipeline is explicitly designed to preserve the underlying problem schema (input structure, expected output type, required control flow, and cognitive load) while varying only surface-level features such as variable names, numeric parameters, and narrative framing. Third, the traditional paper-based examination was constructed by the author using task schemas drawn from the same seed set, leading to the assumption that the underlying cognitive demands of the two formats were comparable.

It is important to acknowledge, however, that no formal psychometric validation of difficulty equivalence was carried out. In particular, the study did not apply item response theory (IRT) models, Rasch analysis, or independent expert rating of item difficulty, nor was a pilot calibration conducted on a separate sample prior to the main examination. The assumption of task comparability therefore rests solely on structural design considerations and the automated unit-test validation built into the AI generation pipeline, rather than on empirically estimated difficulty parameters. The lack of formal psychometric validation remains a central limitation of the present study; any underlying difficulty mismatch between the gamified and traditional items acts as a potential confound in the between-group comparison and cannot be ruled out with the current data. This limitation is revisited in the Discussion, and a formal difficulty-equivalence validation is identified as a priority for future work.

4.2. The gamified application

Fig. 1. The game area (own screenshot)

CodingUs is a client–server web application whose visual theme is inspired by the popular game Among Us (Fig. 1). The social-deduction mechanics of Among Us have recently attracted pedagogical attention in other contexts as well. The frontend, built with Angular, presents students with a login screen, a challenge-selection interface (Fig. 2), and an animated game area in which collectable coins trigger programming tasks displayed in a modal window (Fig. 3). On the server side, Firebase handles user authentication and static-asset hosting, while a Node.js backend running on a dedicated server is responsible for the computationally critical operations: dispatching AI-generated tasks, executing submitted Python code in a sandboxed environment, and running the automated unit-test suite that determines the correctness of each submission. This client–server separation is architecturally significant for the study because it ensures that code execution and grading occur entirely on the server, preventing client-side tampering with test results, and it provides the infrastructure for the integrity mechanisms described in the next subsection.

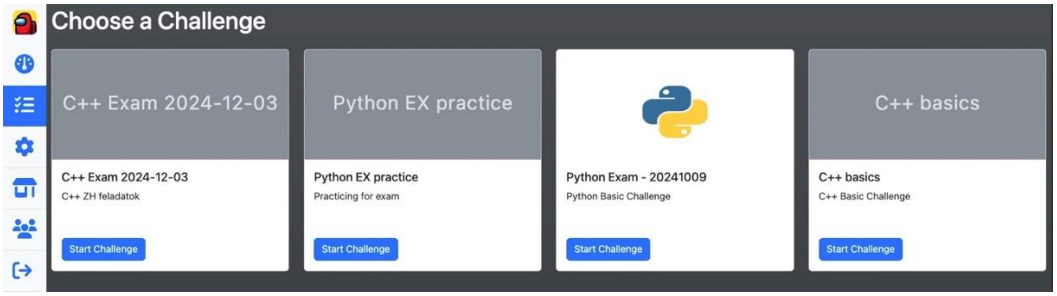

Fig. 2. Challenge screen (own screenshot)

4.3. Integrity and Security: Gamification as a Fraud Prevention Strategy

Academic integrity in computer science education faces unprecedented challenges in the era of generative artificial intelligence. The widespread availability of large language models capable of producing syntactically correct and logically coherent code has fundamentally altered the landscape of programming assessment (Lancaster, 2023; Lodge, 2024). Traditional approaches—such as standardised problem sets administered under proctored conditions—are increasingly vulnerable, as students can leverage AI-assisted tools, collaborative messaging applications, or pre-shared solution repositories to circumvent individual assessment requirements. In this context, the design of CodingUs was motivated not only by engagement goals, but also by the practical need to develop an assessment environment in which opportunities for dishonest behaviour are structurally minimised.

A central integrity mechanism in CodingUs is the elimination of uniform problem sets. The AI-driven task generation pipeline (described in the Technical Validation subsection) produces unique, isomorphic variants of each programming challenge. Although the tasks are equivalent

in difficulty and assess the same learning objectives, the specific variable names, data structures, input parameters, and expected outputs differ between individual instances. Consequently, even students seated next to each other receive different problem instances, which effectively neutralises answer-copying.

The CodingUs web application also incorporates several technical countermeasures. The built-in code editor disables clipboard operations (copy, cut, and paste), including both keyboard shortcuts and context menu interactions. While these client-side restrictions are not impervious to circumvention by technically sophisticated users, requiring that all code be manually typed introduces a structural friction that may theoretically discourage casual copying. This constraint carries an additional pedagogical benefit: it reinforces familiarity with programming syntax and language constructs. Furthermore, the distinctive visual interface—featuring an animated character navigating a virtual environment and interacting with task-bearing coins—produces a screen display that is immediately recognisable during proctored sessions. A brief visual scan of the examination room was sufficient to verify whether each student was engaged with CodingUs or had navigated to an external resource, representing a practical improvement over traditional text-editor-based examination.

Beyond technical measures, exploratory classroom observations suggested that the gamified format might influence the motivational dynamics surrounding assessment, though this remains an anecdotal hypothesis requiring formal validation. When an examination is experienced as an engaging, mission-like challenge rather than a purely evaluative event, the motivational dynamics shift in ways that are unfavourable to cheating. This is consistent with SDT: students whose needs for autonomy, competence, and relatedness are satisfied are more likely to be intrinsically motivated to demonstrate their own capabilities (Ryan & Deci, 2000). Observational evidence from the experimental sessions supports this interpretation—students in the gamified condition appeared more consistently focused on their own tasks and exhibited fewer behavioural indicators of attempted collusion compared to students in the traditional setting. While this observation was not formally quantified, it is consistent with the broader gamification literature suggesting that increased engagement reduces both the motivation and perceived need for dishonest behaviour (Hamari et al., 2014).

It is instructive to contrast this design-based approach with the surveillance-based methods that have become prevalent in higher education, particularly since the shift to remote assessment during the COVID-19 pandemic. Automated proctoring systems—employing webcam monitoring, screen recording, and eye-tracking—have been widely adopted but have also attracted significant criticism regarding their intrusiveness, potential for bias, the anxiety they

induce, and their effectiveness (Lodge, 2024). CodingUs explores an alternative design philosophy: rather than relying solely on post-hoc detection, the system attempts to make honest engagement a more natural pathway through individualised tasks and technical restrictions. However, the actual efficacy of this approach in reducing academic misconduct remains an open empirical question.

It should be acknowledged that the integrity mechanisms described here are not absolute. Determined students with advanced technical knowledge could potentially circumvent client-side restrictions, and no automated system can fully eliminate the possibility of collusion. Moreover, the present study did not include a formal assessment of cheating incidence. Future iterations of the platform could address these limitations by incorporating server-side behavioural analytics—such as keystroke dynamics, typing speed patterns, and mouse movement trajectories—to provide quantitative indicators of individual versus assisted work. Nevertheless, the combination of structural, technical, and psychological integrity measures represents a meaningful advancement over traditional examination formats, aligning the goals of academic integrity with the pedagogical benefits of gamified assessment.

4.4. AI-Assisted Task Generation and Technical Validation

Large language models (LLMs) can assist in the creation of programming tasks, unit tests, and accompanying explanations, provided that an appropriate model-selection strategy is employed. The present pipeline uses two OpenAI models.

Because the gamified assessment required hundreds of distinct programming items, manual authoring was impractical. A multi-stage AI workflow was therefore designed to automate the generation process.

The first model, GPT-4o-mini, is computationally efficient and cost-effective for generating coding challenges. Due to its reduced parameter count, however, it is less proficient at producing natural-language explanations.

The second model, GPT-4o, is approximately ten times more expensive per token but substantially more capable of producing longer textual outputs and detailed explanations.

Combining the two models allows each to be deployed for the task class to which it is best suited: GPT-4o-mini for initial code generation and GPT-4o for quality-assurance review.

The generation process consists of five rounds.

1. Creating the programming task. In this round, the Node.js backend sends the prompt to the mini model through the OpenAI API. The instructor does not need to compose the entire

prompt, as the application supplies reusable template segments. In most cases, the instructor provides only a brief topic sentence. Once the response is received, the backend extracts the generated source code together with a detailed task explanation.

2. The generated code is forwarded to GPT-4o for evaluation. If GPT-4o determines that the code does not meet the specified requirements, the generation–evaluation cycle is repeated up to a maximum of three iterations. If all three attempts fail, the pipeline terminates the process.

3. The generated task file undergoes a dry run: the pipeline verifies that the code executes without runtime errors but does not evaluate its semantic correctness at this stage.

4. An automated unit test is generated for the task, following the same two-model generation–evaluation procedure with a maximum of three attempts.

5. The task file is executed together with its unit test. If both pass successfully, the Node.js backend persists the files to disk and returns them to the front-end interface, where the instructor can assign the new item to the appropriate challenge.

Based on the measurements, the average generation time is 30-45 seconds. Creating an equivalent task manually requires at least one hour.

The iterative development of effective prompts presented several practical challenges.

LLMs frequently append unsolicited explanations to generated code. To suppress this behaviour, the prompt included the directive: "Provide no additional explanation beyond the example code."

A further directive required each generated function to begin with a docstring describing its expected functionality and arguments.

The prompt for unit-test generation was substantially more complex and included the following constraints:

- Because the tested file’s name is determined at runtime, the prompt required a standardised import preamble at the head of the test file.
- Do not give any explanation; the answer only contains the codes.
- All test cases should be in a new method.
- Hide the messages of the test class.
- Hide any other messages that are generated in the test process.
- Hide the Python log during the testing.
- After the whole test process, print a report about the test in JSON format.
- The last command includes the function that is the base of the test.

Generated Test

```
1
2  import unittest
3  import sys
4  import os
5
6  file_name = sys.argv.pop()
7  current = os.path.dirname(os.path.realpath(__file__))
8  parent = os.path.dirname(current)
9  sys.path.append(parent)
10 __import__('solutions.' + file_name)
11 mod = sys.modules['solutions.' + file_name]
12
13 class TestIntersectLists(unittest.TestCase):
14
15     def test_common_elements(self):
16         result = mod.intersect_lists([1, 2, 3, 4], [3, 4, 5, 6])
17         expected = [3, 4]
18         self.assertCountEqual(result, expected)
19
20     def test_no_common_elements(self):
21         result = mod.intersect_lists([1, 2, 3], [4, 5, 6])
22         expected = []
23         self.assertCountEqual(result, expected)
24
25     def test_all_common_elements(self):
26         result = mod.intersect_lists([1, 2, 3], [1, 2, 3])
27         expected = [1, 2, 3]
28         self.assertCountEqual(result, expected)
```

Fig. 3. The automatically generated unit-test (own screenshot)

4.5. Experiment

The comparative study was conducted in the Advanced Programming course at John von Neumann University, which was selected because Python is an accessible first language for assessment purposes and because a mature ecosystem of unit-testing tools makes automated correctness evaluation feasible.

The final analytic sample comprised 63 first-year Information Technology students enrolled in the Advanced Programming course at John von Neumann University during the 2024 autumn semester. Of these, 32 were full-time students and 31 were correspondence students. Group allocation to the assessment format was determined by student self-selection at the beginning of each examination session, as further described in the paragraph on study design below. After allocation and application of the remedial-examination procedure (see below), 36 students were classified as having taken the gamified assessment as their primary format (19 full-time, 17 correspondence) and 27 as having taken the traditional paper-based assessment (13 full-time, 14 correspondence). None of the participants had prior formal Python programming experience; all had previously completed an introductory programming course delivered in a different language. The earlier drafts of this manuscript referred to the cohort as comprising “around

seventy” or “seventy-one” students; this figure reflected initial enrolment, whereas the number reported here reflects the final analytic sample after exclusion of students who did not complete the examination in either format. Demographic variables beyond study programme (e.g., age, gender) were not collected for the purposes of this investigation and consequently cannot be reported.

This Python programming course was at the basic level. The students already knew the basic concepts of programming, but most of them did not have Python experience. We chose this subject because Python is an easy-to-learn language, and there are several tools to evaluate and test the codes. The automated tests were important for us because the students immediately had feedback about their codes.

CodingUs was deployed throughout the semester so that students could become familiar with its interface and game mechanics prior to the examination, on the rationale that an unfamiliar system would introduce construct-irrelevant difficulty at the point of assessment. A typical laboratory session combined a theoretical introduction to the relevant Python topic with a set of newly generated in-game tasks addressing the same content.

Based on the students' feedback, the game was engaging and helped them practice new things. Moreover, students provided many additional ideas for developing the program. For example, some of them complained about the coins. In the original gameplay, someone takes a coin and declines the task; they cannot return to this task because the coin disappears immediately. Changing this behaviour was challenging because the Angular user interface's communication with the game engine was overcomplicated. We had to keep the coin in the background and remove it from the game screen only if the students sent their solution to the server.

The comparative component of the study followed a quasi-experimental design with self-selection rather than a randomised controlled trial or a formal A/B test. At the beginning of each examination session, students were informed that two equivalent assessment formats were available—a traditional paper-based Python exam and the gamified CodingUs exam—and were permitted to choose the format in which they wished to be assessed. As described in the Materials and Methods section, this autonomy-supportive arrangement was a deliberate pedagogical choice grounded in Self-Determination Theory, intended to avoid the “mandatory fun” effect documented in the gamification literature. A methodological consequence of this design, however, is that group membership was not randomly assigned; baseline differences in motivation, prior digital literacy, or attitudes toward game-based learning may therefore be confounded with the assessment format itself. The University’s examination regulations further provide that students whose first-attempt result is unsatisfactory may take a remedial

examination during the final week of the semester. For students who exercised this option, the alternative format was offered at the second attempt, and the best of the two scores was retained for the present analysis. This remedial-examination crossover introduces an additional source of non-independence between the conditions and is acknowledged as a limitation of the design; its implications for the interpretation of the results are discussed in the Discussion section.

We blocked the copy-and-paste function in the game's built-in online code editor to prevent fraud. This feature also blocked the control+c, control+v, and right-click context menus. However, it was not enough to block all cheats; students had to type codes into the editor, which helped them practice the syntax and language elements. Because of that, in some cases, the gamified exam can be more difficult than the traditional one.

Fig. 4. A generated Python task (own screenshot)

To ensure procedural comparability, both assessment environments were administered under identical time constraints of 90 minutes. Grading in both formats evaluated the same core competencies, including functional correctness, algorithmic logic, and proper syntax use. In the gamified CodingUs condition, the integrated unit-testing pipeline provided students with immediate automated feedback upon submission, whereas students in the traditional condition received manual grading feedback only after the examination session concluded. Both groups were permitted one primary attempt during the session, with final scores calculated based on the percentage of test cases passed.

5. Results

We compared the results with several statistical methods.

Mean comparison. Students assigned to the gamified condition obtained a mean score of 63.06% (SD = 31.61), whereas students in the traditional condition achieved a mean of 68.89% (SD = 34.68). The resulting mean difference of 5.83 percentage points, in favour of the traditional format, is modest in magnitude and, as the inferential analyses reported below demonstrate, does not reach statistical significance. The present descriptive comparison should therefore not be interpreted as evidence that either format is systematically superior in producing higher scores; the observed gap is well within the range of variation expected from random sampling under the null hypothesis of no group difference.

Table 1 summarises the descriptive statistics for the two assessment formats side by side, reporting the mean, standard deviation, and median score for the gamified and traditional groups as well as the final analytic sample sizes. As shown, the gamified group obtained a slightly lower mean (63.06%) and a markedly lower median (60%) than the traditional group (68.89% and 80%, respectively), while both groups exhibited comparable dispersion around their centres.

Table 1. Result of the experiment

Metric	Gamified	Traditional
Mean	63.06%	68.89%
Std Dev	31.61	34.68
Median	60%	80%
Count	36	27

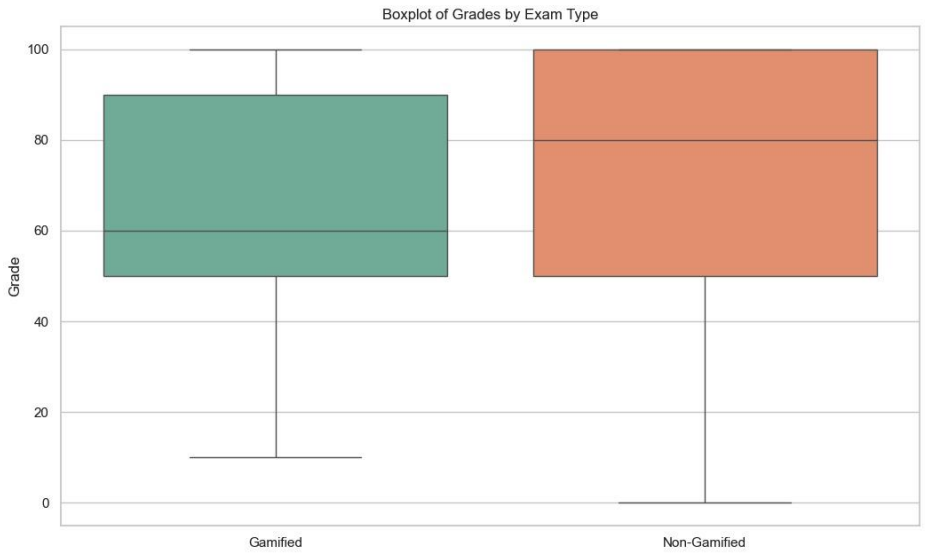

Fig. 5: boxplot of grades (own screenshot)

Figure 5 presents boxplots of the two score distributions side by side. The traditional group's box is concentrated in the upper half of the scale with a high median, whereas the gamified group's box is lower and wider, indicating greater score variability and a more symmetric spread around a lower centre.

Median comparison. The median scores diverged more markedly than the means: 60% in the gamified cohort versus 80% in the traditional cohort. Expressed differently, half of the students in the traditional group scored at least 80%, whereas only half of the students in the gamified group reached 60%. While this asymmetry illustrates different score concentrations, the pronounced differences in distribution shapes (bimodal versus right-skewed) mean that this median difference should not be over-interpreted as a definitive measure of comparative performance. Several procedural factors may contribute to this pattern. In particular, the gamified environment disabled clipboard operations in the in-browser editor, distributed individually generated isomorphic tasks, and required all code to be typed manually—conditions that are more stringent than those of a conventional paper-based examination. These constraints should be interpreted as potential confounds that may have depressed gamified scores rather than as evidence that the gamified format yields higher measurement precision. Any claim regarding the comparative precision of the two formats would require a dedicated psychometric analysis of task difficulty and item equivalence, which falls beyond the scope of the present study.

Dispersion and homogeneity of variance. The standard deviations of the two groups were comparable (SD_gamified = 31.61; SD_traditional = 34.68; absolute difference = 3.07 percentage points), indicating similar score dispersion across conditions. The homogeneity of variances was formally confirmed using Levene's test, which did not detect a significant departure from equal variance, $F(1, 61) = 0.014$, $p = .907$. This result justified the subsequent use of pooled-variance inferential statistics alongside their non-parametric counterparts. The substantial within-group dispersion observed in both conditions (standard deviations exceeding 30 percentage points in each case) reflects the heterogeneous prior programming experience of first-year Information Technology students and is consistent with the distributions typically reported in introductory programming courses.

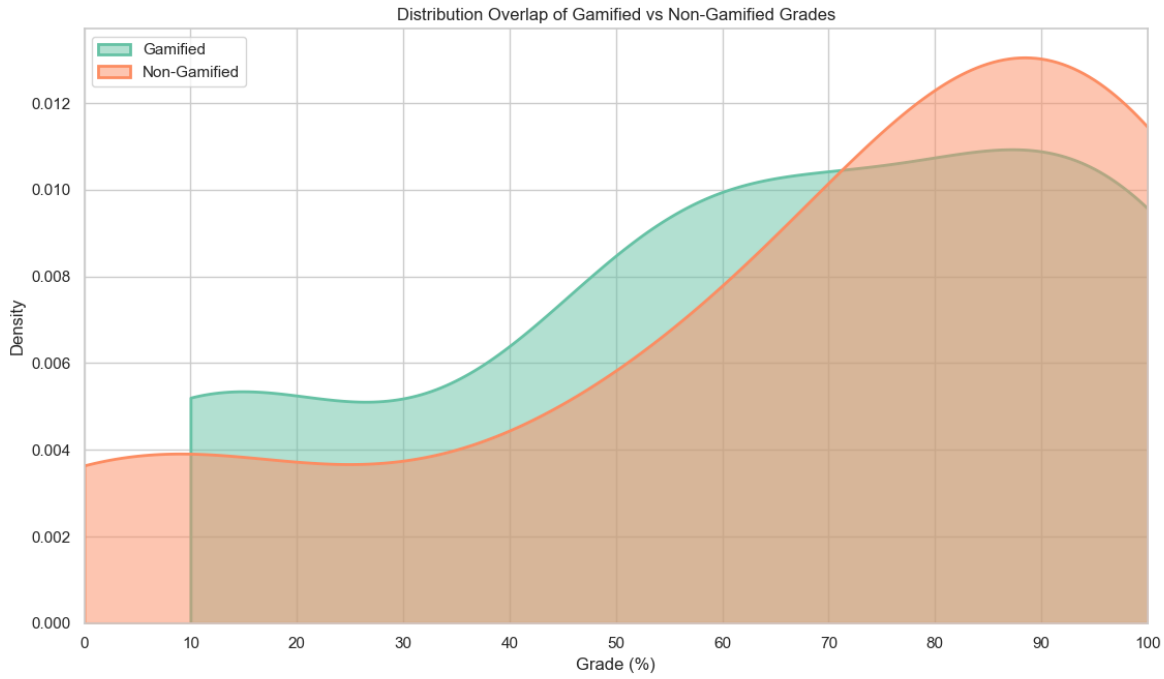

Fig. 6. Density of grades (own screenshot)

Sample composition and distributional assumptions. The final analytic sample comprised 63 students (n_gamified = 36; n_traditional = 27). The moderate imbalance between groups arose from two sources: student self-selection into the assessment format offered at the beginning of each examination session, and the remedial-examination crossover procedure described in the Methods section, under which students who requested a second attempt were reassigned to the alternative format. The normality of the score distributions within each group was assessed using the Shapiro–Wilk test. Contrary to the assumption of approximate normality that had been inferred from sample size alone, both groups exhibited statistically significant departures from a normal distribution (gamified: $W = 0.881$, $p = .001$; traditional: $W = 0.809$, $p < .001$). Visual inspection of the histograms (Figure 7) corroborated this conclusion, revealing pronounced bimodality in the traditional group and a right-skewed, heavy-tailed distribution in the gamified group. In light of these findings, the Mann–Whitney U test was adopted as the primary inferential procedure, and the independent-samples t-test was retained only as a supplementary, convergent comparison.

Primary inferential analysis. The Mann–Whitney U test, which compares the rank distributions of two independent samples without requiring normality, did not detect a statistically significant difference between the gamified and traditional groups, $U = 423.50$, $p = .383$ (two-tailed). This result indicates that, under the ranking metric, the probability that a randomly selected gamified score exceeds a randomly selected traditional score does not differ significantly from chance. The non-parametric test was selected as the primary procedure because the Shapiro–Wilk

results reported above ruled out the parametric normality assumption, and because the Mann–Whitney U statistic is robust against the presence of heavy tails and outliers observed in the empirical distributions.

Supplementary parametric analysis. For completeness, and to enable comparison with prior literature that conventionally reports parametric statistics, an independent-samples t-test assuming equal variances was also conducted. This analysis produced a conclusion fully convergent with the non-parametric test, $t(61) = -0.70$, $p = .490$. Because Levene's test had already confirmed the homogeneity of variances, the pooled-variance formulation was appropriate; nevertheless, Welch's adjusted t-test was additionally computed as a robustness check and likewise failed to reject the null hypothesis, $t(53.70) = -0.69$, $p = .496$. The mean difference between groups (gamified minus traditional) was −5.83 percentage points, with an associated 95% confidence interval spanning from −22.61 to +10.94 percentage points. The breadth of this interval reflects substantial uncertainty: the data remain statistically compatible with a practically meaningful disadvantage for the gamified format (up to a 22.61 percentage point deficit). Consequently, the present study is not sufficiently powered to draw firm conclusions, and the observed performance metrics must be interpreted with high caution.

Effect size. The standardised effect size, computed as Cohen's d using the pooled standard deviation, was −0.18. According to the conventional benchmarks proposed by Cohen (1988), this value falls within the small-effect range ($|d| < 0.20$ is typically interpreted as negligible, and $0.20 < |d| < 0.50$ as small). Thus, even if a systematic performance difference between the two assessment formats were to exist in the population, its practical magnitude under the present experimental conditions would be limited. The convergence of the Mann–Whitney U test, the two t-test variants, the small effect size, and the wide confidence interval provides a consistent—though necessarily cautious—evidentiary basis for the interpretation that the present data do not support the hypothesis of a meaningful performance gap between the gamified and the traditional examination formats.

Interpretive caveat. It should be emphasised that a non-significant null-hypothesis test does not, in itself, constitute evidence for the equivalence of the two formats. The absence of a detected difference must be distinguished from a positive demonstration of no difference, the latter of which would require either an equivalence-testing framework with a pre-specified smallest effect size of interest or a substantially larger sample. Consequently, the most defensible interpretation of the present results is that they are consistent with the absence of a large performance advantage for either format, under the specific boundary conditions of the present study (introductory Python programming, a mixed full-time and correspondence cohort, a

single semester at a single institution, and AI-generated isomorphic tasks of assumed but not formally validated difficulty equivalence). Broader claims regarding the relative psychometric precision, diagnostic validity, or generalisability of gamified assessment would require additional studies employing randomised allocation, larger and more diverse samples, and validated measures of task difficulty and student engagement.

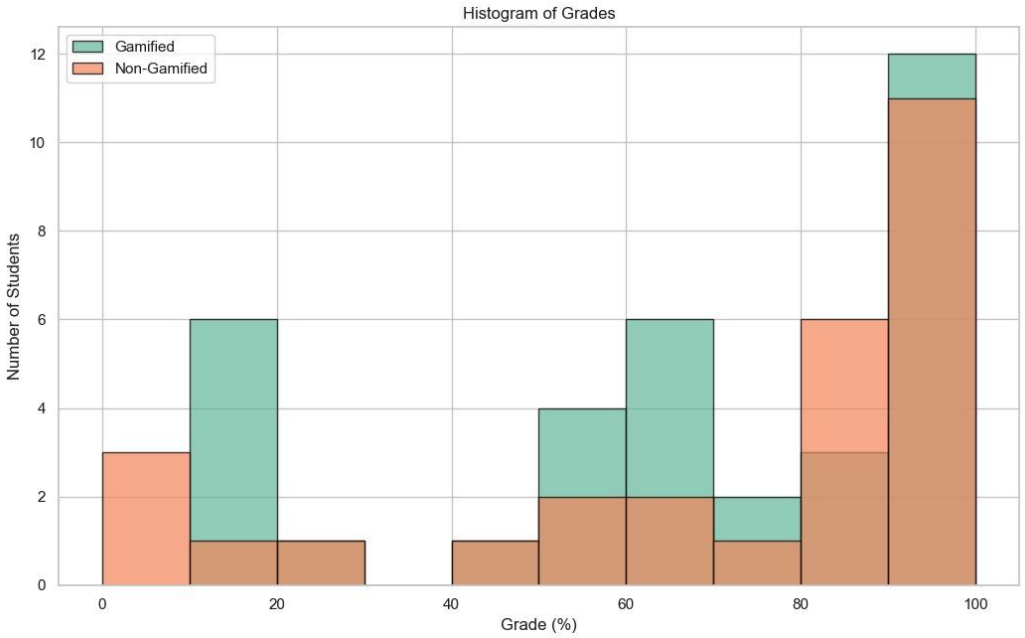

Fig. 7. Histogram of grades (own screenshot)

Figure 7 presents the histograms underlying the density estimate. The traditional group displays a pronounced bimodal pattern with concentrations near the upper end of the scale, while the gamified group exhibits a right-skewed, heavy-tailed shape. Together with the Shapiro–Wilk results reported above, these visual patterns motivated the choice of the Mann–Whitney U test as the primary inferential procedure.

6. Discussion

We did not find a significant difference between these two measuring methods. Naturally, this does not mean a gamified system applies in all cases. We can only make cautious statements.

A central caveat in interpreting the present results concerns the nature of the study design. Although the comparison between the gamified and traditional formats was originally framed as an A/B test, it is more accurately characterised as a quasi-experimental comparison with student self-selection. Because participants were permitted to choose the format in which they wished to be assessed, rather than being randomly allocated, the two groups may differ systematically on unmeasured background variables—most plausibly on baseline motivation, affinity for digital and game-based environments, prior informal programming exposure, and

attitudes toward formal testing. Any observed or unobserved performance differences can therefore reflect the joint contribution of the assessment format and these pre-existing dispositional factors, and the two sources of variance cannot be disentangled within the current data. This is a well-known limitation of voluntary-allocation designs in educational research, and it is the reason why the statistical results are reported in this paper as non-significant rather than as evidence of equivalence between the two formats.

A second, related limitation concerns the remedial-examination procedure. Because students who requested a second attempt were offered the alternative assessment format and because their best score was retained for analysis, a subset of participants effectively contributed information to both experimental conditions. This partial crossover further attenuates the independence of the two groups and may be expected to bias the estimated mean difference toward zero, since the same individuals contribute, in part, to both distributions. Although the overall direction of the results—a small, non-significant advantage for the traditional format—is unlikely to be an artefact of this procedure alone, the magnitude of the observed difference should be interpreted with this structural feature of the data in mind. A fully randomised design with independent first-attempt cohorts, combined with a larger sample size, would be required to draw firmer causal conclusions about the relative effectiveness of the two assessment formats.

Within the specific boundary conditions of this study—namely, students who had been familiarised with the gamified platform prior to the examination and tasks that were designed to be of comparable difficulty across the two formats—the gamified assessment yielded mean performance scores that were statistically indistinguishable from those of the traditional paper-based exam. The present results indicate that no statistically significant performance difference was detected in this sample; however, they do not establish that the gamified format is equivalent, more measurement-precise, more valid, or more reliable than the traditional one. Any such psychometric claim would require a dedicated reliability and validity study, including a formal analysis of task difficulty equivalence, internal consistency, and criterion-related validity, none of which fall within the scope of the present investigation.

We measured only a special topic: programming education. Additional experiments are needed to expand these experiences to other subjects.

7. Conclusion

The present study set out to examine whether a carefully designed gamified assessment format can serve as a viable alternative to a conventional paper-based examination in an introductory

university programming course. Within the specific boundary conditions of the investigation—a single cohort of 63 first-year Information Technology students at one institution, a quasi-experimental design with student self-selection, AI-generated isomorphic Python tasks whose difficulty equivalence was structurally designed but not formally psychometrically validated, and informal rather than instrumented measurement of student experience—no statistically significant performance difference was detected in this sample between the two formats, and the exploratory qualitative indicators were consistent with, but did not establish, the motivational and integrity-related benefits described in the gamification literature. The study's contribution should therefore be understood as a feasibility demonstration and a methodological template for AI-assisted gamified assessment, rather than as definitive evidence of the relative effectiveness of the gamified format. Several avenues for future research follow directly from the limitations described above.

Firstly, fraud detection can be more precise with behavioural checks. For example, we grab behavioural data from the students while they play and work in the application. The system can save metrics from mouse paths, the speed of typing or frequently used text patterns, and so on. Based on these data, we can see who is doing the task without having to watch the students with a camera.

Secondly, AI-generated tasks are a relatively new and fascinating field. It is not only an experimental area; it can free teachers from having to mark hundreds of exams manually and speed up exam creation. Some AI orchestrators can connect LLMs to become complicated networks. These systems can help run repeating tasks, for example, task and test generation, safely. Automated tasking and evaluating systems are also a promising area for future research. Further work should also address the alignment of such tools with institutional assessment regulations and intellectual-property frameworks, so that instructor-facing workflows remain both pedagogically sound and legally compliant.

In sum, the present investigation provides a feasibility template and an initial empirical data point for the design of AI-supported gamified summative assessment in programming education, while clearly delineating the methodological steps—randomised allocation, psychometric item calibration, and validated instruments for engagement and integrity—required before stronger claims about the format's comparative effectiveness can be made.

Funding statement

The author received no financial support for the research, authorship, and/or publication of this article.

Declaration of competing interest

The author declares that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abrache, M.-A. & Oubahssi, L. (2024). The Effects of Gamification on Students' Flow Experience: A Controlled Experimental Study. In Proceedings of the 2024 IEEE 24th International Conference on Advanced Learning Technologies (ICALT), Nicosia, Cyprus, pp. 232–234. https://doi.org/10.1109/ICALT61570.2024.00074

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: defining gamification. In Proceedings of the 15th International Academic MindTrek Conference, Tampere, Finland, pp. 9–15.

Fakhoury, S., Naik, A., Sakkas, G., Chakraborty, S., & Lahiri, S. K. (2024). LLM-Based Test-Driven Interactive Code Generation: User Study and Empirical Evaluation. IEEE Transactions on Software Engineering, 50(9), 2254–2268. https://doi.org/10.1109/TSE.2024.3428972

Frankford, E., Antensteiner, T., Vierhauser, M., Sauerwein, C., Wallner, V., Groher, I., Plösch, R., & Breu, R. (2025). A Survey on Feedback Types in Automated Programming Assessment Systems. ACM Transactions on Computing Education. https://doi.org/10.1145/3773911

Hamari, J. (2019). Gamification. The Blackwell Encyclopedia of Sociology. Malden: Blackwell Pub., pp. 1–3.

Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does Gamification Work? — A Literature Review of Empirical Studies on Gamification. In Proceedings of the 47th Hawaii International Conference on System Sciences, Waikoloa, HI, USA, pp. 3025–3034.

Kapp, K. (2012). The gamification of learning and instruction: game-based methods and strategies for training and education. John Wiley & Sons.

Karl, É., Nagy, E., & Molnár, Gy. (2025). Advanced Examination Systems: Applying Fuzzy Logic and Machine Learning Methods in Education. In A. Szakál (Ed.), IEEE 12th International Joint Conference on Cybernetics and Computational Cybernetics, Cyber-Medical Systems (ICCC 2025), pp. 231–235.

Lampropoulos, G. & Sidiropoulos, A. (2024). Impact of Gamification on Students' Learning Outcomes and Academic Performance: A Longitudinal Study Comparing Online, Traditional,

and Gamified Learning. Education Sciences, 14(4), 367. https://doi.org/10.3390/educsci14040367

Lancaster, T. (2023). Artificial intelligence, text generation tools and ChatGPT – does digital watermarking offer a solution? International Journal for Educational Integrity, 19, 1–12.

Lindsay, G. (2011). Gamifying Archives, A Study of Docugames as a Preservation Medium. 16th International Conference on Computer Games (CGAMES). IEEE Press, pp. 172–176.

Lodge, J. M. (2024). The evolving risk to academic integrity posed by generative artificial intelligence. TEQSA Report.

Michie, S., van Stralen, M. M., & West, R. (2011). The behaviour change wheel: A new method for characterising and designing behaviour change interventions. Implementation Science, 6(1), 42.

Mirmotahari, O., Grun, H. N., & Berg, Y. (2025). Gamification as a transferable pedagogical innovation for technology education: developing 21st-century skills through collaborative game-based learning. Frontiers in Education, 10, 1684459. https://doi.org/10.3389/feduc.2025.1684459

Molnár, Gy. & Cserkó, J. (2022). AI Based Plagiarism Checking: Ease of use and applicable system for teachers to find similarities in students' assessments. IEEE 5th International Conference and Workshop in Óbuda on Electrical and Power Engineering (CANDO-EPE).

Molnár, Gy. & Nagy, E. (2025). Current Issues in Effective Learning: Methodological and Technological Challenges and Opportunities Based on Modern ICT and Artificial Intelligence. EAI/Springer Innovations in Communication and Computing, 1, pp. 1–11.

Molnár, Gy., Cserkó, J., Nagy, E., & Balogh, Z. (2023). Creating own AI datasets from different language sources efficiently: Advantages, Disadvantages and Best Practices to Make Own Datasets. IEEE 6th International Conference and Workshop Óbuda on Electrical and Power Engineering (CANDO-EPE).

Nakamura, J., & Csikszentmihalyi, M. (2002). The concept of flow. In C. R. Snyder & S. J. Lopez (Eds.), Handbook of positive psychology (pp. 89–105). Oxford University Press.

Oliveira, W. & Hamari, J. (2025). Flow Experience in Gameful Approaches: A Systematic Literature Review, Scientometric Analysis, and Research Agenda. International Journal of Human–Computer Interaction, 41(20), 13113–13139. https://doi.org/10.1080/10447318.2025.2470279

Ruhi, U. (2015). Level Up Your Strategy: Towards a Descriptive Framework for Meaningful Enterprise Gamification. Technology Innovation Management Review, 5(8), 5–16.

Ryan, R. M. & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. American Psychologist, 55, 68–78.

Sailer, M., Hense, J. U., Mayr, S. K., & Mandl, H. (2017). How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. Computers in Human Behavior, 69, 371–380.

Sarsa, S., Denny, P., Leinonen, J., & Hellas, A. (2022). Automatic Generation of Programming Exercises and Code Explanations Using Large Language Models. In Proceedings of the 2022 ACM Conference on International Computing Education Research V.1, Lugano, Switzerland, pp. 27–43.

Serra, P. & Oliveira, A. (2025). AI-Powered Prompt Engineering for Education 4.0: Transforming Digital Resources into Engaging Learning Experiences. Education Sciences, 15(12), 1640. https://doi.org/10.3390/educsci15121640

Shoaib, S., Yakubu, M., Khanzada, A., Abdullahi, S., Zakari, A., Nasser, M., Danyaro, U., & Umama. (2025). LLM-Based Code Generation: A Systematic Literature Review with Technical and Demographic Insights. IEEE Access, 13, 1124–1142.

Ulloa, E. & Carcausto, W. (2024). Gamification in Educational Assessment: Transforming Learning into Interactive Experiences through a Systematic Review. Revista InveCom, 4, 1–15.

Zichermann, G. & Cunningham, C. (2011). Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps (1st ed.). Sebastopol, California: O'Reilly Media.

Zourmpakis, A.-I., Kalogiannakis, M., & Papadakis, S. (2024). The Effects of Adaptive Gamification in Science Learning: A Comparison Between Traditional Inquiry-Based Learning and Gender Differences. Computers, 13(12), 324. https://doi.org/10.3390/computers13120324