Assessing the Role of Python Programming Gamified Course on Students' Knowledge, Skills Performance, Attitude, and Self-Efficacy

Manuel B. Garcia
College of Computer Studies
FEU Institute of Technology
Manila, Philippines
mbgarcia@feutech.edu.ph

Teodoro F. Revano Jr
College of Computer Studies
FEU Institute of Technology
Manila, Philippines
tfrevanojr@feutech.edu.ph

***Abstract*— Coding is widely regarded as a fundamental skill of the 21st century. Yet, there is still a shortage of programmers worldwide which disproportionately affect the innovation goals of many sectors. In this study, we evaluated the installment of a Python programming gamified course in higher education, and measure its effect on students' knowledge, attitude, self-efficacy, and skills performance. Two sections with 50 students each were randomly assigned to experimental or control groups. After one semester, the experimental group exhibited significantly higher scores in laboratory activities (skills performance) compared to the control group. Furthermore, they demonstrated a significant improvement with reference to attitude and self-efficacy before and after intervention. Therefore, we concluded that the use of a Python programming gamified course was an effective method for students to learn coding and programming concepts. The use and installation of a gamified course in learning other computer programming languages is highly recommended.**

Keywords—Computer Programming, Python Programming, Gamification, Online Course, Programming Education, Coding

I. Introduction

As the world steadily transitions to the era of automation, computer programming is becoming an essential skill in many areas of the society. Apart from the expected labor shortage in the field of Information Technology (IT) due to the impact of Coronavirus Disease 2019 (COVID-19) pandemic [1], many introductory programming students face learning difficulties for acquiring the necessary knowledge and skills in this area [2]. A recent study on computer programming highlighted the learning challenges faced by novice programmers in terms of affective factors (e.g., motivation and attitude) and individual differences (e.g., aptitude and mathematical ability) [3]. It also provokes discontent among students, and the difficulty of the subject is considered to be an important dropout factor in IT education [4]. Thus, to provide programming students with the necessary support, many have endeavored to develop policies, strategies, and tools for teaching computer programming. For instance, proposed pedagogies to facilitate the creation of an effective learning environment includes game-based learning [5], augmented reality [6], visual block programming-based instruction [7], gamification [8], robot programming [9], and intelligent tutoring systems [10], to name a few. Nevertheless, there is no agreement on what the most efficient method is as far as how and the extent to which these proposed approaches can be used to support students in learning programming [11]. COVID-19 and online education aside, most universities still utilize traditional teaching methods (e.g., lectures, homework, and coding demonstration), which is in contrast to the fact that innovative pedagogies are evidently more effective [12].

Following the recommendations that specialized tools are needed to allow students acquire detailed knowledge and be motivated in the course [13], we installed a gamified course in teaching Python programming in higher education. By doing so, we hypothesized that programming students would have enhanced (a) basic programming knowledge, (b) self-efficacy of computer programming, (c) coding skills performance, and (d) attitude towards the course. To test these hypotheses, we adopted a quasi-experimental study design where two sections with 50 students each were randomly assigned to experimental or control groups. This study contributes to the existing thread of discussion related to programming teaching strategies and gamified courses (notably in Python programming language). In addition, understanding how programming students learn, use, and interact with this innovative pedagogy may establish a basis for educational institutions, curriculum developers, and programming professors on what supplemental strategies can be used in teaching computer programming aside from what was mentioned and adopted on existing studies.

II. Background of the Study

With the wide array of programming languages available in different types (e.g., procedural, functional, object-oriented, scripting, etc.), choosing the best language that would satisfy the needs and conditions of a specific problem domain can be a challenging task. Nevertheless, regardless of the preferred language of any newbie, learning basic programming concepts is still more significant. With this rationale, programming is often introduced in primary education using pedagogies (e.g., game-based learning or visual block programming) suited for this level and without a necessary focus on the language itself. Still, an exploratory survey of 100,000 open source software projects found on GitHub showed that there is a difference in terms of popularity, interoperability, and impact of languages used in computer programming [14]. According to this study, earlier programming languages like C are still a popular option in modern software projects and the rapid spread of website applications has made Ruby and JavaScript pervasive. On the other hand, the most recent annual Developer Survey of Stack Overflow revealed Python as the third most popular language among 83,439 software developers from 181 countries [15]. This is partially attributed to the rise of artificial intelligence and machine learning applications where Python is the most endorsed programming language [16]. Another study [17] also listed other applications for Python such as data science and Internet-of-Things. Figure 1 presents different emerging fields where Python can be useful as the programming language. Its wide applications are attributed to its comprehensive standard library as well as its large developer community.

978-1-6654-0167-8/21/$31.00 ©2021 IEEE

Fig 1. Common Applications of Python Programming Language

Meanwhile, programming languages come and go but the teaching strategies used to teach its basic concepts stay. Thus, experts recommend to continue focusing on interventions in a regular classroom setting foster computational practices [18]. With the shift to online education due to pandemic, it is also important to modify such interventions and make it applicable to the current setup of teaching and learning process. Critical success factors for online learning during COVID-19 showed technology knowledge management (e.g., using software to facilitate learning) as one of the most significant consideration among educational institutions [19]. In addition, gamification was proposed as a technique to produce a sustainable learning environment amidst the COVID-19 pandemic [20]. With such method (e.g., the use of game elements such as leaderboards, badges, points, and virtual currencies), the positive impact on students' self-efficacy, grade motivation, self-determination, and career motivation are evident. As well, a literature review from 2014 and 2020 on gamification applications in an online learning shows that gamified courses could be a valuable tool for acquiring knowledge and skills such as decision-making, communication, and cooperation [21]. When gamification is adopted in a computing course, there are important findings and considerations based on existing studies. First, interface of the game should be user-friendly to allow players to easily understand the game mechanics and focus on learning [22]. It is also a significant factor that affects players' gameplay [23]. On the other hand, because the levels of difficulty (e.g., easy, moderate, hard) inculcate a competitive spirit among players [24], the gamified learning course is recommended to have a wide range of tasks with different level of complexity. This is crucial since there is a positive correlation between students' academic competitiveness and their involvement behaviors [25]. In the core of gamified programming courses, there must also a significant role for the development of problem-solving skills as the lack thereof is one of the reasons of demotivation [26]. Lastly, the impact of gamification is not only limited to the actual gameplay but also when students were instructed to develop their own game [27]. This is further validated by the utilization of game development in a recent study as a method for acquiring Information and Communications Technology related skills such as computer programming [28].

III. Methodology

A. Research Design

Following the research design used in evaluating Jigsaw teaching strategy in a computer programming course [3], we adopted a quasi-experimental research using a nonequivalent control group pretest-posttest design. Like a true experiment, a quasi-experimental design tests causal hypotheses but with the absence of a key ingredient: random assignment. Though, instead of randomization on a student level which may cause treatment contamination, a cluster (by section) randomization was adopted to prevent selection bias. Furthermore, students enrolled in their preferred class schedule, and the researchers including computer programming professors had no control over their course and section assignments. Lastly, this study was conducted in accordance with the ethical principles in the Declaration of Helsinki and of the University. All participants were asked and agreed to be part of the experiment.

B. Sample and Intervention

Sophomore students who were enrolled in an Integrative Programming and Technologies (IT0011) course from April to July 2021 were recruited in the study. Because this course is the fifth programming-related subject in the curriculum, all students have prior programming experience. Among the ten sections enrolled in IT0011 during the time of the study, we randomly selected and assigned two sections with 50 students each to participate in the study ($N = 100$). The same syllabus was used for both groups, but the experimental group had an additional gamified tasks throughout the semester using the CheckiO classrooms [29]. This gamified course platform for Python programming provides fun coding challenges which can serves as an additional resource for teaching and learning Python programming. Puzzles that are aligned with the lesson and student outcomes were given as an additional activity for the experimental group. Nevertheless, scores from these tasks did not affect students' final grade for lecture and laboratory.

C. Instruments

Two types of data (cognitive and affective scores) were collected from four different sources: (1) summative tests for knowledge, (2) technical summative assessments (laboratory activities) for skills performance, (3) Computer Programming Self-Efficacy Scale [30] instrument for self-efficacy, and (4) Attitude Scale of Computer Programming Learning [31] tool for attitude. In addition, demographic information (e.g., age, gender, college life satisfaction) was also incorporated on the final instrument. Both instruments for affective factors have a Cronbach's alpha ranging from 0.84 to 0.96.

D. Data Collection and Analysis

The survey instrument was distributed one week after the course orientation (April 29, 2021) to the experimental group (pre-test), and both experimental and nonequivalent groups completed almost the same survey instrument (post-test) on the last synchronous class session (July 19, 2021). Scores on both summative tests and laboratory activities were collected as well with consent from teachers and participants alike. All statistical analyses were performed using SPSS Statistics. In reporting homogeneity and data distribution, independent t-test, descriptive statistics, chi-square tests, and Fisher's exact test were used. Finally, independent t-test and paired t-test were used to compare score differences in all factors.

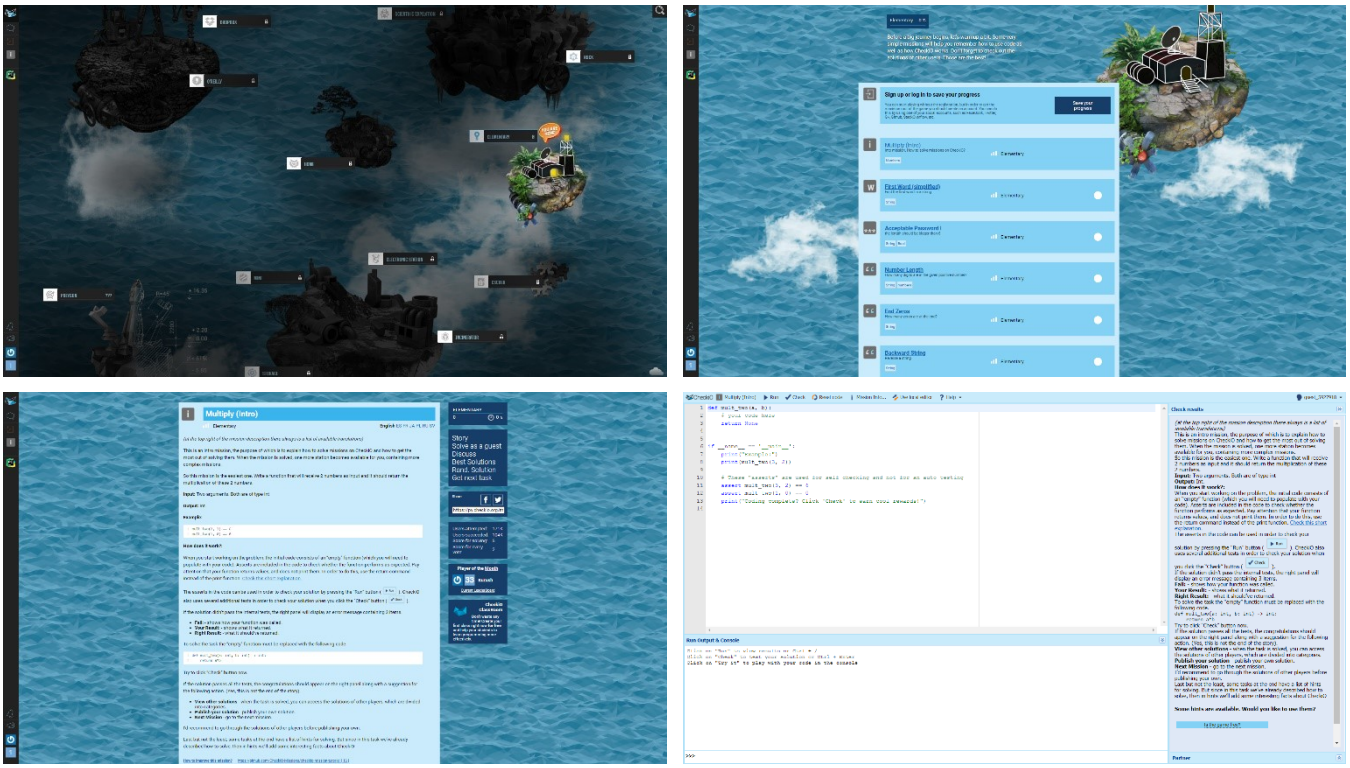

Fig 2. CheckiO Classrooms: Stations, List of Coding Challenges, Task Discussion, and online Integrated Development Environment (IDE)

IV. Results and Discussion

The purpose of the present study was to assess the effect of a Python programming gamified course on both cognitive (knowledge and skills performance) and affective (attitude and self-efficacy) domains in higher education. Participants (see Table 1 for baseline characteristics) were dominated by male students (93.75%) and the mean age was 22.56 ± 1.33 years. All participants experienced gamification in different platforms, including those in an online course (experimental group: 68% vs. control group: 58%, $\chi2 = 0.91$, p = .334) and in a physical classroom (experimental group: 24% vs. control group: 12%, $\chi2 = 0.34$, p = .332). Majority of the participants, from both experimental and control group (96%, $\chi2 = 0.41$, p = .292), are satisfied with their college life. Furthermore, both groups (experimental group: 58% vs. control group: 64%, $\chi2 = 0.02$, p = .593) are satisfied with their present online course experience. Lastly, for previous programming course grade, the experimental group received a 2.5 (34%) while the control group received a 3.0 (30%) grade.

Aside from the between-analysis as shown on Table 2, we also performed a within-analysis in terms of affective factors. Using paired t-tests to identify whether there is a significant difference on the pre- and post-test scores of the experimental group, we determined mixed findings. First, students' attitude did not significantly improve after using the gamified course (willingness: $p = 0.104$, negativity: $p = 0.626$, and necessity: $p = 0.061$). In contrary, students' self-efficacy did improve after using the gamified course (logical thinking: $p = 0.000$, algorithm: $p = 0.000$, debug: $p = 0.001$, control: $p = 0.022$, and cooperative: $p = 0.019$). As a supplemental perspective, we also plotted the mean grades of the groups from their most recent programming course as pre-tests (knowledge = lecture, skills performance = laboratory) for comparison.

TABLE I. Homogeneity Test of Characteristics

Characteristics and Categories	**Exp. Group ($N = 50$)**	**Con. Group ($N = 50$)**	**t or $\chi2$**	***p***
Age	19.36 ± 1.27	19.41 ± 1.32	0.53	.340
Gender				
Male	43 (86)	41 (82)	0.01	.635
Female	7 (14)	9 (18)		
Gamification Experience				
Online Course	34 (68)	29 (58)	0.91	.334
Physical Classroom	12 (24)	6 (12)	0.34	.332
College Life Satisfaction				
Yes	48 (96)	48 (96)	0.41	.292
No	2 (4)	2 (4)		
Online Course Satisfaction				
Yes	29 (58)	32 (64)	0.02	.593
No	21 (42)	18 (36)		
Previous Programming Course Grade				
4.0 (95.8 – 100)	4 (8)	6 (12)	0.88	.626
3.5 (91.5 – 95.7)	9 (18)	4 (8)		
3.0 (87.2 – 91.4)	11 (22)	15 (30)		
2.5 (82.9 – 87.1)	17 (34)	9 (18)		
2.0 (78.6 – 82.8)	1 (2)	13 (26)		
1.5 (74.3 – 78.5)	4 (8)	1 (2)		
1.0 (70.0 – 74.2)	3 (6)	1 (2)		
Failed (Below 70.0)	1 (2)	1 (2)		

Data: Mean ± Standard Deviation and Frequency (Percentage)

TABLE II. MEAN DIFFERENCES BETWEEN GROUPS' SCORES IN COGNITIVE AND AFFECTIVE DOMAINS

Variables	Groups	Pre-test		Post-test		Difference	
		M* ± *SD	***p*-value**	***M* ± *SD***	***p*-value**	***M***	***p*-value**
Knowledge	Experimental	74.23 ± 10.56	0.629	82.56 ± 9.35	0.525	8.33	0.001
	Control	78.64 ± 16.74		79.53 ± 8.26		0.89	
Skills Performance	Experimental	87.91 ± 12.67	0.152	94.23 ± 6.89	0.042	6.32	0.000
	Control	82.57 ± 11.11		84.35 ± 15.88		1.78	
Attitude	Experimental	2.93 ± 1.19	0.247	4.14 ± 0.76	0.001	1.21	0.032
	Control	2.50 ± 1.06		2.94 ± 1.19		0.44	
Self-Efficacy	Experimental	3.13 ± 0.98	0.677	4.22 ± 0.94	0.019	1.09	0.046
	Control	3.45 ± 0.90		3.88 ± 0.97		0.43	

A. Knowledge and Skills Performance (Cognitive)

Although not significant ($p = 0.525$), the mean knowledge score in the experimental group was 82.56 ± 9.35 which is higher compared to the 79.53 ± 8.26 in the control group. This result indicates that the use of a gamified course for learning Python programming does not affect the extent of knowledge gained by students. This result echoes a previous research that evaluated gamification in C programming course, where the gamified learning approach has a positive effect but with lack of statistical significance on students' knowledge [32]. On the other hand, the mean skills performance score in the control (84.35 ± 15.88) and experimental (94.23 ± 6.89) groups were statistically significantly different ($p = 0.042$). This discovery indicates that gamification in computer programming is more effective when it comes to hands-on activities (e.g., coding) than gaining theoretical understanding. This can be partially attributed to the many coding challenges readily available in the gamified course, which a player needs to solve in order to progress to another world (game level). In addition, problem solving is a common learning difficulty among students [33]. Therefore, by reinforcing coding activities, students are more likely to manifest their metacognitive control skills in search for different solutions to the present machine problems. It is therefore recommended for future adopters of gamification in computer programming (whether Python or other languages) to incorporate learning tasks within the game mechanics.

B. Attitude and Self-Efficacy (Affective)

Gamifying the programming course elicit a significantly different scores for both attitude ($p = 0.001$) and self-efficacy ($p = 0.019$). The attitude score (4.14 ± 0.76) of students from the gamified course of Python programming was higher than the score (2.94 ± 1.19) of students from the control group. In learning computer programming, a positive attitude towards the course is vital for students to succeed [3]. When students have a positive attitude, it improves their learning efficiency, and are more willing to learn and practice coding out of their own willingness. On the other hand, under a negative attitude, students may struggle to learn new concepts proactively and may result in a dislike of programming [34]. Therefore, the positive attitude recruited by gamifying the course holds an important implication in computer programming education. One possible explanation is the programming experience they acquired from a series of coding challenges and tasks as part of the game mechanics. In an exploratory study of computer programming in the 21st century, programming experience is an important factor that influences student's attitude [35]. In the case of self-efficacy, the experimental group (4.22 ± 0.94) has a significantly higher score than the control group (3.88 ± 0.97). This outcome supports the literature asserting that the participation of individuals in a gamification program wields a positive influence on perceived self-efficacy [36]. Because students with high self-efficacy are more likely to accept the challenging tasks than students who have low self-efficacy, it only means that the more programming tasks they receive and solve, the higher their self-efficacy becomes. In a traditional programming classroom where time is limited, teachers give enough machine problems for students to solve in the class. This is different from a gamified course where many tasks are already available for students to solve. Moreover, because it is in the game mechanics to solve these tasks to progress the level, students are more motivated to put forth more effort to accomplish the machine problems [8]. Similar findings were found in gamifying basic Java computer programming [37].

V. CONCLUSION

The present study brings attention to the effectiveness and important role of gamification in computer programming. By using a Python programming gamified course, we found out that students from the experimental group had significantly higher scores in skills performance (hands-on activities) than students from the control group. They also had higher scores in terms of theoretical understanding (knowledge) although not significantly. Conversely, both attitude and self-efficacy of students who participated in a gamification program were significantly higher than students who had a traditional setup. It is therefore recommended to use gamification as part of an intervention strategy in teaching computer programming and for fostering computational practices. These results contribute to the existing thread of discussion in computer programming education and gamification (individually and collectively).

One advantage of this study was that it examined the use of gamification in both cognitive and affective domains. We also performed the experiment for the whole term, rather than getting a single set of pre- and post-test scores for comparison of the treatment intervention. Nevertheless, our study has still limitations that future research may address. First, we utilized an existing gamified course where we do not have control on any of its aspect from learning contents to game mechanics. Thus, we cannot tailor the gamification program according to our needs (e.g., personalized contents and challenges) and the capabilities of our learning management system (e.g., points and leaderboard within the institution). Meanwhile, caution should be exercised so as not to overgeneralize the results due to a relatively small sample size. Although, it was appropriate to still run the statistical analysis of the study. Finally, future researchers should try other game elements (e.g., badges).

Despite these limitations, our study demonstrates the vital role of gamification in the context of computer programming course, particularly during the time of a pandemic. Although online learning has been widely accepted in the past [38], the shift to emergency remote learning brought new challenges for teachers and students. As the present situation continues to demotivate students, educational leaders and policymakers should introduce interventions to inspire purpose, motivation, and confidence in their academic communities. Educators as well are encouraged to develop innovative strategies to adapt current learning to new academic needs [39]. Gamification, an engaging strategy to deliver curricula materials, is one way to encourage a positive behavior during this negative time.

REFERENCES

[1] T. Breaux and J. Moritz, "The 2021 Software Developer Shortage Is Coming," *Communications of the ACM*, vol. 64, no. 7, , 2021.

[2] J. Prather, R. Pettit, K. McMurry, A. Peters, J. Homer, and M. Cohen, "Metacognitive Difficulties Faced by Novice Programmers in Automated Assessment Tools," presented at the ACM International Computing Education Research, Espoo, Finland, 2018.

[3] M. B. Garcia, "Cooperative Learning in Computer Programming: A Quasi-Experimental Evaluation of Jigsaw Teaching Strategy with Novice Programmers," *Education and Information Technologies*, vol. 26, pp. 4839–4856, 2021.

[4] M. N. Giannakos et al., "Identifying dropout factors in information technology education: A case study," in *2017 IEEE Global Engineering Education Conference (EDUCON)*, 2017.

[5] P. M, N. M. N, R. Dakshina, S. S, and B. S. R, "Learning Analytics: Game-based Learning for Programming Course in Higher Education," *Procedia Computer Science*, vol. 172, 2020.

[6] C.-H. Teng, J.-Y. Chen, and Z.-H. Chen, "Impact of Augmented Reality on Programming Language Learning: Efficiency and Perception," *Journal of Educational Computing Research*, 2017.

[7] J. M. Sáez-López, J. del Olmo-Muñoz, J. A. González-Calero, and R. Cózar-Gutiérrez, "Exploring the Effect of Training in Visual Block Programming for Preservice Teachers," vol. 4, no. 3, 2020.

[8] J. Figueiredo and F. J. García-Peñalvo, "Increasing student motivation in computer programming with gamification," in *2020 IEEE Global Engineering Education Conference (EDUCON),* 2020.

[9] J. M. Rodríguez-Corral, A. M. Estévez, D. Molina, F. J. García-Peña, C. A. A. Rodríguez, and A. A. C. Balcells, "Application of Robot Programming to the Teaching of Object-Oriented Computer Languages," *International Journal of Engineering Education*, 2016.

[10] T. Crow, A. Luxton-Reilly, and B. Wuensche, "Intelligent tutoring systems for programming education: a systematic review," in Proceedings of the 20th Australasian Computing Education Conference: Association for Computing Machinery, 2018.

[11] N. Bubica and I. Boljat, "Strategies for Teaching Programming to Meet New Challenges: State of the Art," in *Contemporary Issues in Economy & Technology*, Split, Croatia, 2014.

[12] A. Zhang et al., "Effects of Innovative and Traditional Teaching Methods on Technical College Students' Achievement in Computer Craft Practices," *SAGE Open*, vol. 10, no. 4, 2020.

[13] Kanika, S. Chakraverty, and P. Chakraborty, "Tools and Techniques for Teaching Computer Programming: A Review," *Journal of Educational Technology Systems*, vol. 49, no. 2, pp. 170-198, 2020.

[14] T. F. Bissyandé, F. Thung, D. Lo, L. Jiang, and L. Réveillère, "Popularity, Interoperability, and Impact of Programming Languages in 100,000 Open Source Projects," in *2013 IEEE 37th Annual Computer Software and Applications Conference*, 2013.

[15] Stack Overflow. (2021). 2021 Annual Developer Survey. Available: https://insights.stackoverflow.com/survey/2021

[16] Z. Lin, "A Methodological Review of Machine Learning in Applied Linguistics," *English Language Teaching*, vol. 14, no. 1, 2021.

[17] P. N. S. Jyothi and R. Yamaganti, "A Review on Python for Data Science, Machine Learning and IOT," *International Journal of Scientific & Engineering Research*, vol. 10, no. 2, 2019.

[18] S. Y. Lye and J. H. L. Koh, "Review on teaching and learning of computational thinking through programming: What is next for K-12?," *Computers in Human Behavior*, vol. 41, pp. 51-61, 2014.

[19] A. Y. Alqahtani and A. A. Rajkhan, "E-Learning Critical Success Factors during the COVID-19 Pandemic: A Comprehensive Analysis of E-Learning Managerial Perspectives," vol. 10, no. 9, p. 216, 2020.

[20] S. Park and S. Kim, "Is Sustainable Online Learning Possible with Gamification?—The Effect of Gamified Online Learning on Student Learning," *Sustainability*, vol. 13, no. 8, p. 4267, 2021.

[21] A. N. Saleem, N. M. Noori, and F. Ozdamli, "Gamification Applications in E-learning: A Literature Review," *Technology, Knowledge and Learning*, 2021/01/02 2021.

[22] P. Chandel, D. Dutta, P. Tekta, K. Dutta, and V. Gupta, "Digital Game Based Learning in Computer Science Education," *CPUH-Research Journal*, vol. 1, no. 2, pp. 33-37, 2015.

[23] I. Ahmad, E. Hamid, N. Abdullasim, and A. Jaafar, "Game Interface Design: Measuring the Player's Gameplay Experience," in Advances in Visual Informatics, Cham, 2017, pp. 500-509: Springer International Publishing.

[24] S. Combéfis, G. Beresnevičius, and V. Dagiene, "Learning Programming through Games and Contests: Overview, Characterisation and Discussion," *Olympiads in Informatics*, 2016.

[25] S. Shimotsu-Dariol, D. H. Mansson, and S. A. Myers, "Students' Academic Competitiveness and Their Involvement in the Learning Process," *Communication Research Reports*, vol. 29, no. 4, 2012.

[26] S. Papadakis and M. Kalogiannakis, "Using Gamification for Supporting an Introductory Programming Course. The Case of ClassCraft in a Secondary Education Classroom," in Interactivity, Game Creation, Design, Learning, and Innovation, Cham, 2018, pp. 366-375: Springer International Publishing.

[27] O. Shabalina, C. Malliarakis, F. Tomos, and P. Mozelius, "Game-Based Learning for Learning to Program: From Learning Through Play to Learning Through Game Development," in European Conference on Games Based Learning, Graz, Austria, 2017.

[28] M. B. Garcia et al., "Game Development as a Pedagogical Methodology in Learning Related ICT Skills: Perspectives of Freshmen from FEU Institute of Technology," *International Journal of Simulation: Systems, Science & Technology*, vol. 20, 2019.

[29] CheckiO. CheckiO - Coding Games and Programming Challenges for Beginner and Advanced. Available: https://py.checkio.org/

[30] M.-J. Tsai, C.-Y. Wang, and P.-F. Hsu, "Developing the computer programming self-efcacy scale for computer literacy education," *Journal of Educational Computing Research*, vol. 56, no. 8, 2019.

[31] Ö. Korkmaz and H. Altun, "A validity and reliability study of the attitude scale of computer programming learning (ASCOPL)," *MEVLANA International Journal of Education (MIJE)*, vol. 4, 2014.

[32] S. P. Shorn, "Teaching Computer Programming Using Gamification," in 14th International CDIO Conference, Japan, 2018.

[33] C. S. Cheah, "Factors Contributing to the Difficulties in Teaching and Learning of Computer Programming: A Literature Review," *Contemporary Educational Technology*, vol. 12, no. 2, 2020.

[34] S. Fincher et al., "Predictors of success in a first programming course," presented at the 8th Australasian Conference on Computing Education - Volume 52, Hobart, Australia, 2006.

[35] J. Yang, G. K. W. Wong, and C. Dawes, "An Exploratory Study on Learning Attitude in Computer Programming for the Twenty-First Century," in *New Media for Educational Change*, Singapore, 2018, pp. 59-70: Springer Singapore.

[36] A. I. Polo-Peña, D. M. Frías-Jamilena, and M. L. Fernández-Ruano, "Influence of gamification on perceived self-efficacy: gender and age moderator effect," *International Journal of Sports Marketing and Sponsorship*, vol. 22, no. 3, pp. 453-476, 2021.

[37] N. L. Mingoca and E. L. R. Sala, "Design and Development of Learn Your Way Out: A Gamified Content for Basic Java Computer Programming," *Procedia Computer Science*, vol. 161, 2019.

[38] M. B. Garcia, "E-Learning Technology Adoption in the Philippines: An Investigation of Factors Affecting Filipino College Students' Acceptance of Learning Management Systems," *The International Journal of E-Learning and Educational Technologies in the Digital Media (IJEETDM)*, vol. 3, no. 3, pp. 118-130, 2017.

[39] M. Abdulsalam Salihu, A. Gamal Abdulnaser, O.-A. Kingsley, V. Vanye Zira, B. G. Manuel, and B. Yahia, "Gamification of E-Learning in African Universities: Identifying Adoption Factors Through Task-Technology Fit and Technology Acceptance Model," in *Next-Generation Applications and Implementations of Gamification Systems*, P. Filipe and Q. Ricardo, Eds., ed Hershey, PA, USA: IGI Global, 2022, pp. 73-96.