IIAI Open Conference Publication Series
IIAI Letters on Informatics and Interdisciplinary Research
Vol. 004, LIIR090
DOI: https://doi.org/10.52731/liir.v004.090

Using Game Design to Enhance the Learning Motivation of Programming Courses

Chih-Kai Chang [*], Yu-Chi Yeh [*],
Shuo-Heng Hsu [*], Yan-Yi Chen [*]

Abstract

The ability of computational thinking can be improved through learning programming design and practicing logical thinking and problem-solving skills. However, the cognitive load of programming courses can be heavy, and students from various majors may not have sufficient knowledge about computer science. To address this, innovative teaching methods and content should be considered to reduce learning anxiety and improve motivation and performance. This study aims to redesign the Python programming course. In the first stage, a traditional lecture will be used to teach the basic concepts of programming. The CodeCombat platform will be introduced in the second stage, utilizing game-based learning strategies to teach Python programming. In the third stage, the curriculum will integrate the Pygame module for game design and implementation. After each stage of implementation, students will be given a questionnaire on learning motivation. The questionnaire will collect data on students' motivations, student work, and student thoughts for statistical analysis. According to the statistical results, the overall average score of students' learning motivation in the third stage is the highest, while the first stage has the lowest score. Therefore, it can be concluded that the teaching method of learning programming through game design is most effective in improving students' motivation to learn.

Keywords: Computational Thinking, Learning Motivation, Game-based Learning, CodeCombat, Pygame.

1 Introduction

In the digital age, teaching students how to use technology for self-learning and problem-solving is an important issue in education. Besides enhancing students' information literacy and using technology to learn subjects such as reading, math, and science, it is crucial to enable students to utilize computational skills to solve problems. This way, the benefits of technology can be fully utilized. Computational thinking utilizes concepts from computer science to break down problems, establish modules, and design methods to automate problem-solving. Although computational thinking originated from math, engineering, and science, its problem-solving techniques leverage the advantages of fast computational operations and enable more efficient problem-solving. As a result, many advanced countries are currently advocating for reforms in technology education. They aim to incorporate computational thinking into computer science courses in schools.

[*] National University of Tainan, Tainan City, Taiwan

Programming calls for precision. It requires sufficient logical thinking and problem-solving abilities. Learning how to program can enhance one's computational thinking. The description-execution-reflection-debugging-description cycle of programming provides students with a chance to practice computational thinking with great efficiency [19]. Learning how to program also allows students to learn how to think in an organized manner as well as practice geometrical and systematic thinking. This can help improve their understanding of math and science and enhance their logical judgement and problem-solving skills as well as their productivity and creativity [4, 12, 18]. Whether it's structural problems or daily computational problems that students need to learn to solve, computational thinking is required. By learning how to program, one can practice computational thinking, which allows students to build problem-solving and creative-thinking abilities. As a result, programming courses for students have been paid more attention to.

However, learning how to program is a rather difficult task. Learners must learn specific programming syntax and rules, in which ambiguous names often cause confusion [7]. Teachers often teach in a traditional manner, that is, they teach by simply using textbooks or worksheets. Learning how to program not only requires a certain level of logical reasoning, math, and abstract thinking, the amount of knowledge needed (such as keywords, reserved words, and block symbols) is quite a lot. Program languages are too abstract at times, students have a hard time connecting the concepts to tangible things in real life. The complexity of the syntax, misunderstandings of information, slow learning progress, and other learning obstacles are all difficulties students encounter that might decrease their motivation of learning, which affects their learning outcomes [5].

The majority of students aren't very interested in programming courses and there are also gaps in their information capabilities. They have trouble understanding the operation of the internal workings of a computer. Lacking the concepts of detailed programming modules causes students to have difficulty understanding basic programming structures. This results in students not being able to use problem-solving skills learned in programming. Those who find it more difficult than others are more likely to copy others' works or give up because they are unable to work independently. Programming beginners from information-related departments may feel frustrated because they have to spend more time learning syntax to be able to write a simple program, not to mention those from other departments. Strictly word-editing environments also bore students more easily [2]. Nowadays, whether it's common or general courses, teachers have opened programming-related classes. Teachers may encounter students from different backgrounds, without knowing whether they have sufficient knowledge of programming. As a result, teachers should think of innovative methods to teach with. This not only lowers students' fears about programming languages in the initial stages but also boosts their learning motivation and interests, which in turn improves their learning outcomes.

The effects playing digital games have on students' learning has been a topic of discussion and research in recent years. Several studies have shown it to be beneficial to learning. Learners can have a better understanding of the purpose of learning by establishing habits and learning styles that align with their interests [15]. Digital games are products of programming. Past digital-gaming studies have also used finished products as teaching materials. Slowly, students started to learn programming by designing digital games. Gamified learning methods successfully and efficiently allows students to enter the Flow Experience, increase their learning motivation, and place them in the best possible learning state. This turns programming into something interesting [9, 1]. Students are filled with satisfaction when they are able to make something out of nothing. This boosts their learning motivation as this shows how interesting and pleasant programming is. Game design is a combination of programming, creative thinking, and multimedia applications. As the popularity of digital games rises, students' interests and motivation towards learning to program can be boosted by digital-game-centered topics. This also cultivates logical thinking and problem-solving skills, further practicing computational thinking course designs. Rather than learning complex syntax or developing tools, those learning how to program can quickly acquire logical thinking and problem-solving skills with high interaction.

The topic of this research paper is "Using game design to enhance the learning motivation of programming courses". Using gamified learning methods differentiates from traditional teaching methods. Its main research purpose is to make programming courses more suitable for students of all backgrounds and boost students' motivation in learning to program.

2 Literature Review

A few features that make digital games so popular are as follows. It's entertaining, gameful, regulated, interactive, and adaptable, and it gives learners a sense of victory and conflict. Aside from providing results and feedback, it also challenges users, allowing them to solve problems and social interaction, images, and plot lines. Because digital games are entertaining, learners enjoy the process of learning. Its game-like quality makes learners want to participate eagerly, while its rules provide students with the structure of the games. Its goal-oriented quality adds to learners' motivation to play the games and its human-machine interaction allows learners to utilize the computer operation. Its results and feedback provide students with learning opportunities. Adaptability allows learners to move smoothly through the duration of the games and a sense of victory gives them self-satisfaction. Challenges and a sense of conflict evoke excitement in learners. Its problem-solving quality brings about students' creativity and its social interaction allows students to form groups. Its images and plot lines provoke our emotions when playing the games [14]. Therefore digital-game designs should include rules, challenges and competition, backgrounds, interaction, tasks and goals, modules and structures, authenticity and storytelling, and other elements. Different from passive entertainment, such as films and television, a digital game is a form of active entertainment. It has rules to fix the operation of the entire game. And its challenge, conflict, and victory mechanisms turn it into a game. Background refers to the location in which the game is set. Interactive modules refer to how learners play and control characters. Point-of-view refers to how learners observe the game world. Games must include tasks and goals that learners will have to accomplish. The modules explain how the games work and the differences in modules provide learners with different gaming experiences. The structure of the game determines the switches in the module and their time and reasons for doing so. Imitating reality is also a crucial element. Game designers need to strike a balance between what is real and what isn't and create situations and activities. Plot lines often make the games more meaningful [16].

2.1 Game-based Learning

Digital games refer to gaming software that uses electronic forms with programming languages that can appear on screens and can be stored and operated on personal computers. Studies have shown that one reason to use digital games to learn is that gamified learning is meaningful and liked by learners. Using digital games as teaching materials brings about positive effects on learners' cognitive development, ability to focus on learning, learning motivation, and learning results. Studies conducted in Taiwan have shown that education is still mainly tested and enrollment oriented and focus is placed on the subject matter. Whereas gamified learning has long been under development overseas. This allows learners to learn through making games or discussing the benefits designing games have on learning. The learners are mainly elementary school students. Studies have also shown that they already possess the ability to operate game-designing software, which is beneficial to learning by game design [6, 11].

The content and materials used in most programming-related courses don't grab students' attention. To teachers, finding something that'll spark students' interest and guide students to learn programming is something to take into consideration when preparing for a class. Many scholars have tried incorporating different methods into programming courses such as combining language research and principle-oriented teaching methods, analytic methods, or making digital gaming development an official course to make learning programming more diversified [13, 20]. Through the gaming elements, learners can immerse themselves in the learning process. The game's challenges, unpredictability, and competitiveness motivate learners to play. This can also spark learners' curiosity and internal motivation. Incorporating digital games in teaching can kindle learners' motivation which helps with their thinking abilities.

2.1 Learning Motivation

Learning motivation, as defined by numerous psychology textbooks, is an internal process involving the origins of actions and the extension of those actions. Needs, desires, motives, curiosity, discovery, and expectations all spark motivation [10]. Motivation drives people to chase after what they've put their minds to as well as brings about expectations [3]. Students often say, "I really want to do well in school, but I'm just not interested in studying so I'm not able to persist in doing it." or "I'm not interested in studying so I don't do well in school." This makes the correlation between interest and studying or learning very evident. By studying hard and getting good grades, students will be satisfied, want to study even more, and spark interest and motivation to study. Psychologists believe that all human behavior arises from needs. Humans need to satisfy their needs before going after things like self-satisfaction. Needs are motivated internally, not externally, or by external control or restrictions.

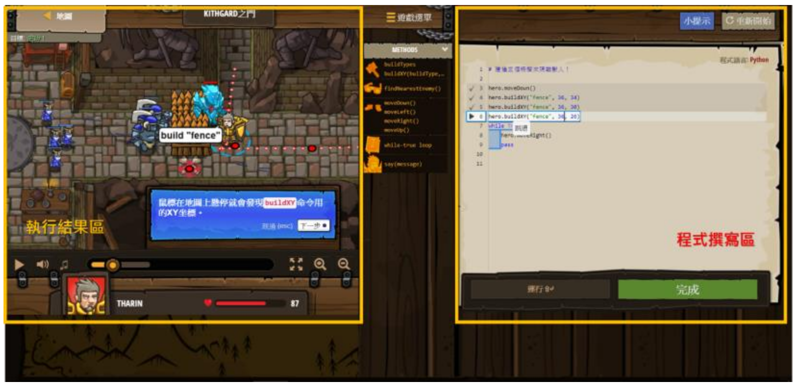
[17] believe when learners do something out of interest, this is called being intrinsically motivated. In terms of behavior models, a learner's actions have their motives. The degree of the motives directly affects the actions. If the motives disappear, the actions will also cease to exist. The same goes for learning. If there is no motive, there won't be any learning. Therefore, learning motivation can be defined as the impetus that drives humans to learn. The text mentioned above tells us that the learning motives that push learners are the key to learning. The ARCS and MSLQ motivation models are tools used to measure motivation. Although there are many learning motivation models based on other theories, such as self-efficacy, goal-setting, or expectancy-value theory, the ARCS model is particularly useful for instructional designers because it provides a framework for designing learning experiences that are motivating and engaging. This research will mainly use the ARCS model, which will be introduced in the research tools section.

3 Research Methodology and Tools

According to the research motive and topic purpose mentioned above, looking into questionnaires that discuss students' learning motives are separated into pre-test, post-test, and second post-test. Learning by playing games and learning by designing games are two different approaches to using games for learning programming. Learning by playing games (i.e. CodeCombat in this study) involves playing existing games that have been designed to teach specific concepts or skills. Learning by designing games involves creating your own games that teach specific concepts or skills. This approach can be more challenging than learning by playing games, but it can also be more rewarding. We introduced Pygame for students to create their own games. After comparing the three, we'll look at the relations and effects of the innovative teaching strategy. The subjects of this study are 50 freshmen students from the Department of Information and Learning Technology of the National University of Tainan. Students enrolled by taking the GSAT or the Advanced Subjects Test. This is fair since students come from all types of backgrounds. The classroom is located in the computer classroom and the course will be for three hours each week for 18 weeks.

3.1 CodeCombat for Learning Programming

CodeCombat, a strategy game made for programming beginners, was established in 2013 with the aim of making learning programming more relaxed and fun. With the levels made by over a hundred volunteers, the game team can incorporate more functions, correct errors, test levels, and translate the game into over 50 different languages, illustrated as Figure 1. Programming languages like Python, JavaScript, CoffeeScript, Lua, and more can be used. There are now over 110 levels that users can play for free. While playing the game, players can learn all kinds of basic programming syntax, logic structures, and set variables. There are also multiple levels to choose from and set-by-step levels allow learners to learn about the programming in a complete and in-depth manner. All the movements in the game are controlled by codes. For example, if you need to fight an enemy to go on to the next level, press the play button on the lower lefthand corner to carry out commands. And you'll see that the character moves according to the commands as shown in the lower left-hand corner of the image below. The program editing section in the upper right-hand corner displays the section that currently operating. If there is a problem in operation, for example, the character runs into a wall or misses a target, players can correct the code and rerun it, which is the process of debugging as shown in the lower right-hand corner of the image below. In summary, CodeCombat is an effective tool for increasing motivation for learning Python. The game is engaging and fun, and it has been shown to be effective in improving students' understanding of Python concepts and their ability to write Python code.



```
# [illegible]

hero.moveDown()
hero.buildXY("fence", 36, 34)
hero.buildXY("fence", 36, 30)
hero.buildXY("fence", 36, 26)
while T
    hero.    Right()
    pass
```

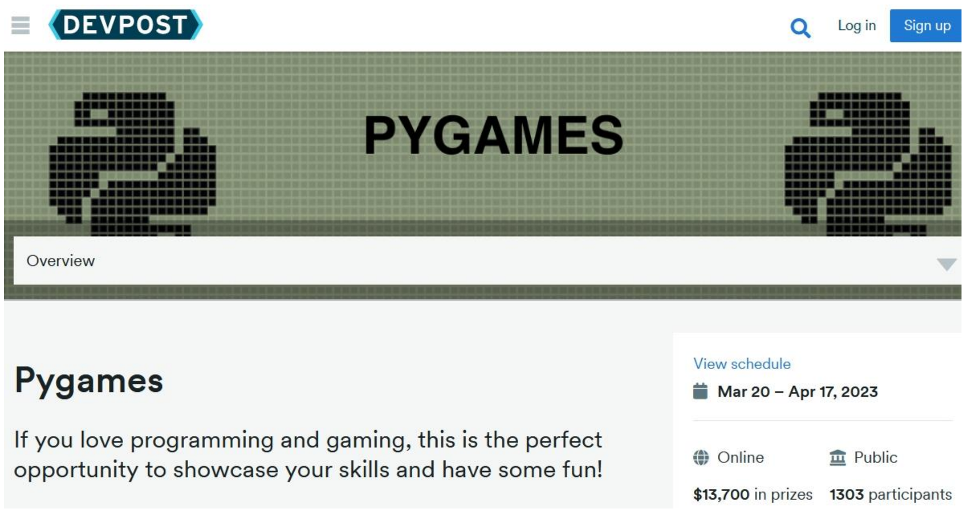
Figure 1: CodeCombat game-based learning platform.

3.2 Pygame for Learning by Designing Games

Game development is an important branch of software engineering that allows students to understand programming and computational thinking by learning about digital games. Pygame (https://www.pygame.org) is a Python module based on the game development wrapper by SDL that is used to code games. SDL, or Simple Direct Media Layer, is a multi-platform media library with open-source codes that provides computational multimedia hardware such as volume control, input, videos, and other functions. It also supports cross-platform application software for developers. Therefore, developers can use Pygame through Python to create interfaces for games and multimedia procedures. Free of the control of low-level languages such as machine language, it can realize electronic game development. Based on this theory, game developers can simplify and incorporate the functions and concepts needed into the game itself. All resources can be provided by high-level languages such as Python, thus lowering the difficulty of game development to the lowest.

Furthermore, programming novices can have fun and meet new people through online Pygame activities. For instance, the Pygame Hackathon, hosted by Microsoft™, is a 24-hour event where participants can build games using the Pygame library. The hackathon is open to anyone who is interested in game development, regardless of experience level. According the webpage shown as Figure 2, there are 1303 participants and $13,700 prizes including cash and scholarships. This study uses the same judging criteria as instructional guides. Hence, students will design their own games according to the following criteria:

- Quality of the Idea: How creative and original is the game? Is it well thought out?
- Fun Factor: How fun, interesting, and exciting is the game to play? Is it easy to use, efficient, and visually appealing?
- Nostalgia: How relatable is the game throughout the user experience and interaction? Does it feel familiar and move the player to reminisce?
- Game Difficulty: How complex is the game and how much skill is required to progress normally through the game experience? Does the player need to use problem-solving, precision, or other skills?
- Technical Implementation: How well was the project built? Did the developers use Python effectively?



Pygames

If you love programming and gaming, this is the perfect opportunity to showcase your skills and have some fun!

Figure 2: Pygames hackathon (https://pygames.devpost.com/).

3.2 ARCS Learning Motivation Questionnaire

This study uses the ARCS motivation survey and Zi-Hui Yang's 2010 survey to measure learning motivation before and after cooperative learning activities. The survey is based on Keller's ARCS motivation module. After evaluation, corrections, and reliability and validity analysis made by professionals, the survey is sectioned into four parts (attention, relevance, confidence, and satisfaction) with a total of 24 questions. J. Keller came up with the ARCS motivation module in 1983. It integrates motivation theories from psychology and the results of teaching-design modules. The belief of ARCS is that teaching materials must spark students' interests or attention to reach the desired learning results. This perspective is quite different from the traditional method that emphasizes teaching content. As a result, ARCS has the following features: attention motive, affection causation, emphasis on the pleasantness of learning, and using strategies to hold learners' interests [8].

ACRS motivation module has four dimensions:

1. Attention: to engage students' interest and spark their curiosity and focus. This includes: (1) providing variability and establishing various examples, modules, activities, and expressions; (2) giving rise to the need to learn such as innovative, surprising, or undetermined knowledge; (3) utilizing questioning skills and giving students difficult learning tasks.

2. Relevance: to satisfy the student's personal needs and goals and produce positive learning attitudes. Its teaching strategies include: (1) how students use prior knowledge and skills or understand, memorize, and familiarize themselves with new knowledge; (2) the purpose of teaching must focus on the student's goals and making sure they accomplish them; (3) designing different teaching activities according to the student's needs.

3. Confidence: to help students become more confident in believing that they have what it takes to succeed. This includes: (1) clearly defining the standards and expectations of succeeding; (2) creating opportunities for self-control; (3) providing chances for success.

4. Satisfaction: refers to the internal and external encouragement and feedback students get from their academic achievement. Common teaching strategies used are: (1) engaging ac-

tivities that allow students to show off what they've learned; (2) giving fair feedback and rewards; (3) maintaining fair and equal character transitions, encouraging self-evaluation, establishing habits of practicing and testing after class, and not over-praising students.

In the first four weeks, the teacher will teach students all the basics of Python, for example, its installation, operation, loops, variables, and other basic concepts. The teacher will use a traditional teaching method with worksheets and textbooks. Students will also have to complete assignments each week and upload them to the course website. Moreover, after class is completed in the fourth week, students will be asked to fill out the first portion of the learning motivation survey. In weeks 5 through 8, the teacher will introduce CodeCombat and its operation and ask students to create a username and password for it. After that, taking the place of traditional teaching methods, the teacher will combine Python and CodeCombat to teach game design. What's more, students will be asked to complete specific CodeCombat levels and to fill out the latter portion of the survey at the end of week 8. After completing the CodeComabt game design courses, students will have a midterm exam in week 9.

Through weeks 10 to 13, the teacher will go more in-depth into the Pygame module of Python. The first two hours of the class will be spent on teaching Pygame and its operation and examples. In the last hour, students will be asked to make a small game independently and upload it to the class website. In weeks 14 to 17, students will focus on making a game using Pygame for their final projects. The first two hours of the class will be focused on more in-depth applications of the Pygame module. Students will be categorized into 8 groups, each containing 5 members. The last hour of the class will be used as discussion time for brainstorming about their completed Python game. Students will be asked to fill out the learning motivation survey as well as a course evaluation questionnaire. Students will be asked to present their final projects in the last week and upload their projects to the class website. The semester's syllabus is as follows.

4 Teaching and Research Outcomes

The teacher will start the course with the traditional programming textbook. Regarding the ARCS of this teaching method, the averages of attention, relevance, confidence, and satisfaction are 3.84, 3.94, 3.23, and 3.61 respectively. The overall average is 3.56, as the following graph shows. Then, students are asked to use the gaming content knowledge from CodeCombat to learn Python. Regarding the ARCS of this teaching method, the averages of attention, relevance, confidence, and satisfaction are 3.72, 3.66, 3.39, and 3.63 respectively. The overall average is 3.60, as the following graph indicates. Finally, students were asked to use or alter Pygame modules to learn Python. Regarding the ARCS of this teaching method, the averages of attention, relevance, confidence, and satisfaction are 3.97, 4.07, 3.33, and 3.86 respectively. The overall average is 3.81, as the following graph shows.

Table 1: Comparison of learning motivation by three teaching methods.

	Attention	Relevance	Confidence	Satisfaction	Avg.
Traditional	3.84	3.94	3.23	3.61	3.56
CodeCombat	3.72	3.66	3.39	3.63	3.60
Pygame	3.97	4.07	3.33	3.86	3.81

In the first and second stages, traditional and gamified teaching methods are used sequentially. In the third stage, teachers learn programming by designing games. By comparing the attention aspect of the three stages, we conclude that the third stage is the most effective, followed by the second stage, then the first stage. By looking at the relevance aspect of the three stages, we deduce that the third stage is the most potent, followed by the first stage, then the second stage. In terms of relevance, most students might've thought using CodeCombat was too easy since they had already used it in the past. Moreover, they were learning the basics of Python, which wasn't that relevant to them. Regarding the confidence aspect, the second stage was the most capable, followed by the third stage, then the first stage. The majority of students most likely thought the content of CodeCombat was relatively simple and easy to understand, hence the second stage was the most effective. In satisfaction aspects, the third stage was placed first, followed by the second stage, then the first stage. Regarding the overall averages, the third stage had the highest, followed by the second stage, then the first stage. In terms of the ARCS, students responded more positively to the gamified teaching method than the traditional one.

Due to the fact that the semester only has 18 weeks, the gamified course couldn't be well-rounded. Moreover, having students learn in three different learning methods may cause confusion and not allow them to get used to the learning methods and learn Python to a certain degree. Therefore, we should've planned a longer course and compared the learning motives of each phase. We hope that we can use gamified teaching methods in different programming courses in the future to find out whether it enhances the learning motives of students when implemented in other programming languages.

There are five open-ended questions regarding the learning motivation questionnaire. Question #1 asks: "If one isn't interested in the topic, it most certainly affects one's learning results. What kind of learning topics would interest you the most and why?" Most students expressed that gamified courses would be more exciting and better at retaining their attention. A few students' responses are as follows.

- Student A: Gamified learning courses are more interesting because programs written under the traditional method often lack creativity and change.
- Student B: I prefer gamified learning because the results tend to be more like games, so they appear to be more interesting.
- Student C: Gamified learning has more attraction because it results in a game and gives one a sense of accomplishment.

Overall, students believe that gamified courses are more effective at teaching and learning than traditional courses.

Question #2 asks: "When faced with task-learning, one tends to lean towards information or knowledge that one is already familiar with. Which learning methods do you most identify with and why?" Some students identified more with gamified learning methods, others preferred the traditional methods, and still others agreed with both methods. A few responses from the students are as follows.

- Student A: I prefer gamified program designing because I'm familiar with many of the games so it's easier for me to grasp concepts and edit the codes.

- Student B: I prefer the traditional method because the finished products are more applicable in real life, and not just games that people can play.
- Student C: I identify with both methods because, after each class, there are tests and assignments to evaluate our progress.

Overall, students believe that both gamified and traditional learning methods can be effective for learning.

Question #3 asks: "Confidence and students' expectations of success are closely related, and it affects students' effort and performances. Of the modules, which is moderately or not challenging and motivates you to keep learning? Please give your reason." Most students expressed that the gamified method made them feel more confident. A few responses from the students are as follows.

- Student A: By using the gamified learning method, you only have to make simple changes to the codes to finish the assignments. This gives me the confidence to keep learning.
- Student B: The gamified learning method allows discussions among classmates, which increases its chances of being successful and gives students more confidence.
- Student C: The gamified learning methods give us more confidence because we can solve problems by discussing and expressing our opinions.

Overall, students find the gamified method to be more confidence-building.

Question #4 asks: "Satisfaction is an evaluation that stems from learning results. Self-satisfaction is a crucial element in maintaining learning motives. Of the modules, which provides you the most satisfaction from retaining knowledge? Please give your reasons." Most students think the gamified learning method satisfies them the most. Below are a few students' responses.

- Student A: I prefer gamified program designing because now I can create the games I used to play when I was little on my own.
- Student B: I prefer the gamified learning method because I can see tangible results from the assignments the teacher asks us to complete.
- Student C: I prefer gamified program designing because I can now design games that I used to play as a kid by myself.

In general, students seem to be satisfied with the gamified learning method because it is engaging and allows them to see tangible results from their learning.

Question #5 asks: "If you were the teacher, how would you go about arranging things like the types of games, the course syllabus, and course content?" Most students gave their suggestions which are as follows.

- Student A: Arranging the course according to the needs of the students.
- Student B: Have everyone design a game at the end of the semester so students can observe each other's work.

- Student C: Explain the basic syntax and sign of programming, then the language and operation of the games. Don't explain everything first then have us do it. Let us perform the tasks while you teach. This makes us feel involved.

In general, the students' suggestions seem to focus on making the course more engaging and relevant to the students' needs. They also seem to emphasize the importance of hands-on learning.

5 Conclusion

The study discusses the benefits of learning programming design in improving computational thinking and problem-solving abilities. While instructors typically use textbooks to explain programming concepts, the heavy cognitive load of programming courses and the diverse backgrounds of students necessitate innovative teaching methods and content to reduce anxiety and improve motivation and performance. The proposed solution is a redesigned Python programming course with three stages: a traditional lecture for basic concepts, CodeCombat for game-based learning, and the Pygame module for game design and implementation. Student questionnaire data, works, and thoughts were collected after each stage for statistical analysis. The results show that the game-based learning strategy in the third stage improves students' motivation the most.

Acknowledgement

Author can include an acknowledgement of this work here. The research reported in this paper has been supported by the Ministry of Science and Technology under research project numbers MOST 108-2511-H-024-008-MY3 and Ministry of Education under research project number PED1101026 in Taiwan.

References

[1] E. F. Anderson and L. McLoughlin, "Critters in the classroom: a 3d computer-game-like tool for teaching programming to computer animation students," in *ACM SIGGRAPH 2007 educators program*, 2007, pp. 7-es.

[2] C. Bishop-Clark, "Comparing understanding of programming design concepts using visual basic and traditional basic," *Journal of Educational Computing Research*, vol. 18, no. 1, pp. 37–47, 1998.

[3] C. S. Carver and M. F. Scheier, *On the self-regulation of behavior.* cambridge university press, 2001.

[4] D. H. Clements and J. S. Meredith, "Research on Logo: Effects and efficacy," *Journal of Computing in Childhood Education*, vol. 4, no. 4, pp. 263–290, 1993.

[5] M. Felleisen, R. B. Findler, M. Flatt, and S. Krishnamurthi, "The TeachScheme! project: Computing and programming for every student," *Computer Science Education*, vol. 14,

no. 1, pp. 55–77, 2004.

[6] K. Jones, "What happens when students design and run their own simulations?," *Simulation & Gaming*, vol. 29, no. 3, pp. 342–347, 1998.

[7] C. Kelleher and R. Pausch, "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers," *ACM Computing Surveys (CSUR)*, vol. 37, no. 2, pp. 83–137, 2005.

[8] J. M. Keller, "Development and use of the ARCS model of instructional design," *Journal of instructional development*, vol. 10, no. 3, p. 2, 1987.

[9] K. Kiili, "Digital game-based learning: Towards an experiential gaming model," *The Internet and higher education*, vol. 8, no. 1, pp. 13–24, 2005.

[10] P. R. Kleinginna Jr and A. M. Kleinginna, "A categorized list of motivation definitions, with a suggestion for a consensual definition," *Motivation and emotion*, vol. 5, no. 3, pp. 263–291, 1981.

[11] A. Nanjappa and R. Van Eck, "Educational games: Learners as creators," in *Seminar in Instructional Design and Technology, The University of Memphis*, 2001.

[12] S. Papert, *Children, computers, and powerful ideas*. Harvester, 1980.

[13] S. A. Papert, *Mindstorms: Children, computers, and powerful ideas*. Basic books, 2020.

[14] M. Prensky, "The games generations: How learners have changed," *Digital game-based learning*, vol. 1, no. 1, pp. 1–26, 2001.

[15] E. F. Provenzo Jr, "The video generation.," *American School Board Journal*, vol. 179, no. 3, pp. 29–32, 1992.

[16] A. Rollings and E. Adams, *Andrew Rollings and Ernest Adams on game design*. New Riders, 2003.

[17] R. M. Ryan and E. L. Deci, "Intrinsic and extrinsic motivations: Classic definitions and new directions," *Contemporary educational psychology*, vol. 25, no. 1, pp. 54–67, 2000.

[18] S. A. Shafto, "Programming for learning in mathematics and science," *ACM SIGCSE Bulletin*, vol. 18, no. 1, pp. 296–302, 1986.

[19] J. A. Valente, "Logo as a Window into the Mind," *Logo Update*, vol. 4, no. 1, pp. 1–4, 1995.

[20] A. Watt and S. Maddock, "Computer games technology and higher education," *Virtual Reality*, vol. 5, pp. 185–194, 2000.