International Journal on Advanced Science Engineering Information Technology

Vol.14 (2024) No. 5
ISSN: 2088-5334

The Development and Application of a MCPS (Motivation and Creative Problem Solving) Instructional Model in Computing Liberal Arts for Non-Majors

Hee Jung Park [a], Yong Ju Jeon [b,*]

[a] *Department of Creativity Software, Andong National University, Gyeongdongro 1375, Andong, GyeongBuk, Republic of Korea*
[b] *Department of Computer Education, Andong National University, Gyeongdongro 1375, Andong, GyeongBuk, Republic of Korea*
*Corresponding author: *yyongju@anu.ac.kr*

Abstract— **The purpose of this study is to develop and validate a learning motivation and creative problem-solving MCPS instructional model for non-majors in a computing liberal arts course. This study aimed to propose a specific method for conducting a computing liberal arts class tailored for non-majors. The research was divided into two phases: model development and model application studies. In the model development study, we formulated the MCPS instructional model and lesson design by integrating Papert's constructionism learning principles, addressing the challenges in computing education for non-majors, and incorporating various teaching methods identified from related research. In addition, to improve the completeness of the instructional model, we developed an evaluation rubric that considered both content and evaluation aspects through expert review. In the model application study, we validated the effectiveness of this instructional model by implementing it in a university class. We employed various research analysis methods to derive further insights and implications. Research suggests that the traits of Papert's constructionism learning theory can enhance learner engagement and foster creative problem-solving in computing education when utilizing the MCPS instructional model. This model, as proposed in the study, can effectively serve as a pedagogical approach for delivering semester-long or more extensive computing liberal arts courses tailored for non-majors in college. We anticipate that the outcomes of this study will contribute to the establishment of robust computing education programs in universities, particularly at a time when the demand for computing education among non-majors is increasing, and its significance is growing.**

Keywords— **Motivation; creative problem solving; MCPS instructional model; computing liberal arts; non-majors; computing education; learning motivation; papert's constructionism.**

Manuscript received 5 Dec. 2023; revised 18 Mar. 2024; accepted 2 Jul. 2024. Date of publication 31 Oct. 2024.

I. INTRODUCTION

The world is undergoing a digital transformation driven by software and artificial intelligence. Accordingly, many countries worldwide, including Korea, are proactively fostering creative talent that utilizes digital competencies to solve problems in the digital era, marking it as a central policy priority [1], [2]. For instance, the Korean government has prepared a comprehensive plan for fostering digital talents and aims to foster 1 million digital talents [1], [2]. Additionally, within university education, initiatives such as the software-centered university project initiated in 2015 underscore the importance of computing education. This mandate extends beyond computing majors, encompassing non-majors and professionals in the field, and is tailored to reflect the specific characteristics of each academic major and department [3].

In contrast to elementary and secondary education, where computing education is deemed essential and systematically integrated into the curriculum [4], universities often face challenges in effectively delivering computing education to non-majors. These challenges stem from a lack of experience and expertise in catering to the diverse needs of non-major students. Consequently, ineffective education practices may arise, such as implementing uniform curricula irrespective of students' majors or capabilities or overly focusing on programming grammar without adequately assessing students' proficiency levels and interests [5]. In the non-majors cases, if computing education is conducted without motivation, class participation and educational effectiveness decrease [5], [6]. Unlike majors, it isn't easy to expect results regarding learning effectiveness with the same teaching methods as majors because they are not familiar with computer use. Therefore, it is necessary to research computing teaching methods for non-majors [7], [8].

In this regard, Shin [9] argued for the necessity of a computing education model to stimulate learners' motivation and cultivate creative problem-solving skills. Kim [10], Park and Choi [11] proposed that, in computing education for non-majors, it's crucial to emphasize discovering functions through self-exploration or peer-based learning grounded in constructionist principles rather than simply teaching command usage.

In particular, to improve the computational thinking of non-majors, it is necessary to focus on enhancing creativity and problem-solving skills rather than acquiring computing skills [11]. Creative problem-solving skills and learning motivation are pivotal variables in non-majors competency [12]. Hence, it is necessary to apply an instructional model that integrates various variables related to creativity, such as motivation and problem-solving skills, to enhance the creativity of college students in educational innovation [13]. Furthermore, Papert [14] transformed learning into an exciting and enjoyable experience through programming tools. His insights, proposed over 40 years ago, remain relevant today more than ever amidst large-scale educational innovations. They are closely related to our reality, particularly in the current context of the educational revolution.

Based on Papert's constructionism learning principles and relevant research on computing education, this study presented specific methods to enhance learning motivation and creative problem-solving skills in a computing literacy course for non-majors. We developed an instructional model for non-major computing literacy courses (MCPS, Motivation, and Creative Problem Solving) and validated its applicability through expert review and field application.

II. Material and Method

A. Studies Related to Computing Liberal Arts Courses for Non-majors

The term "computing" encompasses the academic discipline and is also employed globally in similar contexts, such as informatics or information and communication technology (ICT). CC2020 proposes using the term computing education to mitigate concerns regarding terminological interpretation and promote universalization. Additionally, it proposes a global standard curriculum for computing education at the higher education level [15].

Domestic and foreign universities also recognize the importance of cultivating the computing thinking skills necessary for university students to live in the modern era. Computing education is being implemented across various countries to develop future competencies for all students, although the curriculum and education forms are different [14]. In the case of domestic universities, computing education for non-majors is being established and gradually expanded in alignment with the global trend, starting with the software-centered university project. We reviewed research on computing education for non-majors to glean insights into effective educational methodologies and challenges encountered in computing classes.

Park and Choi[11] suggested that block programming, which has easy grammar and is simple to handle, is utilized as a computing liberal arts class in universities. It aims to improve creativity and problem-solving skills rather than acquiring computing skills. Also highlighted the primary challenge in implementing such curricula: lacking motivation for computing education. Kim [16] asserted that while the number of students exposed to computing education is growing, it often lacks relevance to their majors and is challenging to access. Therefore, concerted efforts are necessary to promote understanding and recognition of the importance of computing. Additionally, Kim [10] stated that students frequently encounter difficulties with programming terminology and principles, and preliminary research should be conducted before implementing computing education. Kang [17] advocated a need for computing education and exposure to computational thinking regardless of major. Still, there are relatively few opportunities to encounter such systems compared to majors. In the educational field, following pre-prepared code differs from student-centered problem-solving learning [18].

Oh et al. [19] analyzed the effectiveness and changes in perception after conducting a Scratch programming class for non-majors. The results showed that it is essential to make non-majors aware of the need for programming classes before conducting them. In addition, in a study on computing education models for non-majors, Kim [20] developed a test-driven problem-solving learning model (CT-TDPS) that applies agile development methods to address the limitations of the waterfall model from a software engineering perspective. Sohn [21] developed a design-based software education model (DBSEM) to improve the difficulties encountered in the process of acquiring programming languages, and Jeon [22] developed a SW-AI teaching and learning model (SAGE) in which learning stages are designed around changes in learner schemas to improve the creative convergence capabilities of preservice teachers. In addition, Choi [23] developed a computational thinking-centered multiple project-based computing education model (CT-MPB) by deploying three stages of projects. Choi emphasized that most existing studies in this domain are biased toward elementary and secondary education, highlighting the necessity for more extensive cases and research focused on universities.

While numerous universities have implemented courses to enhance the effectiveness of computing education, there remains a critical need for research on computing education models tailored to address operational challenges in computing liberal arts classes for non-majors. Specifically, there is a pressing need to develop models that foster creative problem-solving skills among students.

B. Papert's Constructionism

Various studies and examples of computing education have been published recently, but most of them lack an exploration of the pedagogical background, and a discussion of constructionism approaches [24]. The educational paradigm of the knowledge information society emphasizes the value of individual learners' needs and characteristics and learner-centered learning [25]. Constructionism learning theories have been gaining traction in school settings due to changing perceptions of the nature of learning [26].

A mathematician, computer scientist, and educator, Papert expanded on developmental psychologist Piaget's Constructivism educational philosophy to formulate the Constructionism learning theory. Papert developed an early educational programming language called the computer language Logo [27] and used the analogy of teaching a new language to a Logo turtle to introduce the programming concept. Later, the MIT Media Lab, led by his student Resnick, developed Scratch based on Papert's constructionism.

Commonalities emerge when comparing Papert's constructionism to the previously established constructivism of Piaget and Vygotsky. These include knowledge inseparable from context and real-world challenges, constant reflection, and reflexivity, essential as learners build on their own experiences. Collaboration with other learners to solve new problems is also necessary. However, they are differentiated by a spirit of continuous challenge without fear of failure, encouragement and emphasis on failure as a productive opportunity to improve problems through the concepts of "bugs" and "debugging," enjoyment to achieve challenging goals, and proper use of time to accomplish significant things [27], [28], [29], [30], [31]. In the constructionism approach, a paradigm of education in the information age, Papert emphasized that expressing inner feelings and ideas into concrete objects is critical to learning [32], [33].

C. Learning Motivation and Creative Problem Solving

Learning motivation is "the force that initiates learning, determines its direction, and determines the continuity and intensity of learning." A learner's motivation is a significant variable to consider in a teaching-learning situation because it is the driving force for learning to occur [12]. In other words, learners who are not motivated in a teaching-learning situation will lose direction, so motivation is essential for learner behavior [14]. Creative problem-solving is "a process in which creative thinking occurs through the interaction of divergent and convergent thinking in three stages: understanding the problem, generating ideas, and planning and executing actions" [32]. Learning creative problem-solving is essential in computing education because Korean students are proficient at solving problems. Still, they often struggle with identifying and exploring issues independently. Generating solutions through computational thinking involves active and creative work, offering an opportunity to enhance traditional passive forms of education [34].

Studies on the relationship between creativity, problem-solving, and motivation in college students have shown that problem-solving and learning motivation are essential for increasing overall creativity. Therefore, for practical computing classes for college students, it is necessary to develop a class model and research various instructional methods to foster computational thinking skills, learning motivation, and creative problem-solving.

D. Effective Teaching and Learning Methods for Non-majors

Although interest in computing education has expanded to include non-majors, it isn't easy to expect non-majors unfamiliar with computers to learn effectively using the same teaching methods as existing majors [7], [8]. Various instructional methods are needed to facilitate understanding because of the lack of basic programming knowledge and its connection to significant subjects [35].

The most popular instructional method is a universally recognized way of conveying knowledge to students, in which the lecturer demonstrates knowledge while the students listen and learn. This method can effectively transfer linguistic information and cognitive knowledge [32]. Cognitive apprenticeship theory is an instructional method that involves a process of modeling in which the teacher performs a task, and then the learner observes and imitates. It can be applied in all areas of learning, but it is especially helpful in primary areas that require complex cognitive strategies, such as science, math, and information [32].

Problem-based learning is a constructive instructional method that simultaneously teaches subject matter knowledge, skills, and problem-solving strategies by presenting complex, real-world, unstructured problems that require learners to find meaningful solutions to problems or tasks [32], [36]. Project-based learning is a method of performing multiple learning activities in which the learner's mind creates concrete and tangible outputs in the process of solving problems related to real life [37]. When applied to computing education, this method addresses the challenge of teaching simple programming languages and grammar that lack relevance to real-life contexts [38]. In team projects, students with different programming abilities can help each other to complete more challenging tasks. Constructive evaluation activities that allow for sharing opinions and feedback can lead to improved outputs [39].

The CT-CPS (Computational Thinking-based Creative Problem Solving) instructional model integrates each element of computational thinking into the creative problem-solving phase. When applied to computing education, it offers a learning experience where students discover problems on their own and creatively and actively solve them using computers [34]. Therefore, an effective instructional method for non-majors should differ from that of majors and employ various processes according to the context of computing education.

E. The Development of the MCPS Instructional Model

This study suggested a specific method of computing liberal arts classes for non-majors to improve their learning motivation and creative problem-solving skills. To this end, the Motivation and Creative Problem Solving (MCPS) model for non-majors was developed based on Papert's constructionism learning principles, the challenges encountered in computing classes for non-majors, and the insights gleaned from other instructional models. The model was then reviewed and refined through expert evaluation to ensure its appropriateness, leading to the development of a final version.

1) Principles of Design for Developing Instructional Model: Some principles of the instructional model design developed in this study. First, we explored the approach of constructionism, which is an educational paradigm in the information age, and designed based on Paper's learning principles of constructionism [16], [24], [25], [26], [27], [29],

[30], [31], [32], [33]. Second, we analyzed the issues of computing classes for non-majors through previous studies and devised a solution to enhance them by employing the modeling process [5], [10], [11], [16], [18], [23], [31], [38]. Third, to enhance the learning effectiveness of computing education for non-majors, we applied various teaching methods for each stage to foster interest in learning, encourage active class participation, and develop skills for creative problem-solving [32], [34], [36], [37], [38], [39]. Table 1 shows the details of the design principles for developing models for instruction.

TABLE I
DESIGN PRINCIPLES AND DETAILS OF THE MCPS INSTRUCTIONAL MODEL.

Principles of design	Details
Reflection on Papert's constructionism learning principles	A. Use technology, digital technology, as a tool for thinking [27], [33].
	B. Help students learn how to learn rather than teach them everything they need to know [27].
	C. Use programming to organize thinking and problem-solving [27].
	D. Allow them to learn through experience [16], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33].
	E. Allow them to explore what interests them most [27], [28], [33].
	F. Allow learners to bring out their inner feelings and ideas [27], [29], [30], [31].
	G. Allow them to learn from failure [27], [33].
	H. Allow them to express their ideas concretely [28], [32].
	I. Make ideas materialize [28], [32].
	J. Go through a process of proofreading their ideas [27], [28], [32].
	K. Evaluate each other's ideas. Share ideas so that they can communicate with each other [27], [28], [32].
	L. The result is an activity of constructing knowledge based on concrete objects [27], [28], [32], [33].
	M. They can engage in an iterative process of developing their tools and media [27], [28], [32].
	N. Allow learners to manage their learning process [28], [32], [33].
	O. Help them understand how ideas are shaped and transformed when they are expressed through different media and when they are realized in specific contexts [27], [28], [32].
	P. Making learning easy, fun, and engaging [27].
Challenges encountered in computing classes	a. Lack of motivation in computing education [5], [11], [38].
	b. Investigation of learners' characteristics and difficulties before computing education [10], [16].
	c. Prerequisite understanding of computers [5], [11], [10], [16].
	d. Difficulty with programming languages and terminology [10], [11], [16].
	e. A variety of learner-centered curricula and teaching and learning methods [11],[31].
	f. Spreading educational systems and organizing educational environments to provide various computing experiences and opportunities [11], [18], [23].
Reflection of other instructional models	① Principles of lecture-based learning [32].
	② Principles of cognitive apprenticeship [32].
	③ Problem-based learning principles [32], [36].
	④ Project-based learning principles (individual, team) [37], [38], [39].
	⑤ Principles of computational thinking-based creative problem solving (CT-CPS) [34].

2) Draft of MCPS Instructional Model: To develop a draft of the instructional model, we used the principles and details of instructional design in Table 1 to establish the stages of a class, considering when they can be applied. We developed a draft instructional model, as shown in Table 2.

TABLE II
DRAFT OF THE MCPS INSTRUCTIONAL MODEL

Instructional models		Step-by-step design principles		
Steps	**Details**	**Papert's constructionism learning principles**	**Challenges encountered in computing classes**	**Other instructional models**
Step 1. motivation and preparation	1.1 Learning motivation (relevance to major, need for computing education) 1.2 Identifying learner's characteristics and difficulties in computing education (online survey) 1.3 Identifying the antecedents of computing education	A, B	a, b, c	① ,②
Step 2. building of basic knowledge	2.1 Learning motivation (find related current IT issues) 2.2 Demonstration (explain programming grammar) 2.3 Support (using real-life examples) 2.4 Stop helping (extend practice)	A, B, C, D, P	a, b, c, d	
Step 3. application of basic knowledge	3.1 Learning motivation (build expectations by introducing computing skills) 3.2 Problem settings 3.3 Derive a solution to the problem 3.4 Share solutions	A, B, C, D, E, F, G, P	a, b, c, d, e	② ,④

Instructional models		Step-by-step design principles		
Step 4. planning outputs and execution	4.1 Exploring ideas 4.2 Ideation 4.3 Design and share ideas 4.4 Implementing ideas 4.4.1 Proofread ideas 4.4.2 Share Concrete Outputs 4.5 Evaluation	A, B, C, D, E, F, G, H, I, J, K, L, M, N, P	a, b, c, d, e	③ ,⑤
Step 5. development	5.1 Learning motivation (identify challenges and confidence boosts) 5.2 Applying digital technology 5.3 Express ideas through various media	A, O, P	a, f	online/offline

In step 1, 'Motivation and Preparation,' we acknowledged the relevance of the major and the need for computing education to address the issue of insufficient learning motivation. We addressed this by integrating it across all stages, utilizing the identification of learner characteristics and comprehension of computers to sustain motivation. In step 2, 'Building of Basic Knowledge,' expert instructors demonstrate the problem-solving process to assist learners with programming grammar and terminology difficulties. This is further enhanced with real-world examples, gradually reducing support as learners independently solve problems. In Step 3, 'Application of Basic Knowledge,' a learner-centered, problem-based learning environment is established to alleviate the burden on learners and allow them to practice before engaging in team projects as part of a mini-project. In step 4, 'Planning and producing products,' we emphasize the process of correction and communication through intra- and inter-team sharing activities so that intangible ideas can be materialized. It is organized to manage the learning process by using the experience of failure as a learning opportunity. Step 5, 'Development,' is designed to provide an environment and opportunities for learners to develop their sense of challenge and confidence as an extension of the classroom rather than ending with producing and evaluating concrete objects.

3) MCPS Instructional Model Appropriateness Review: To ensure the appropriateness of the proposed MCPS instructional model, the first Delphi survey was conducted to assess the Content Validity Ratio (CVR), followed by the second Focus Group Interview (FGI) for expert review. In the first expert review, a validity review was conducted on the appropriateness of the instructional model, such as the purpose of development of the model, class steps, and applicability. A Likert 5-point scale was used for responses, with five indicating 'very appropriate' and one indicating 'not at all appropriate.' Suggestions for the MCPS instructional model were freely written in an open format.

The validity analysis method was calculated using the CVR of Lawshe (1975), as shown in Equations [40].

$$\mathrm{CVR} = \frac{N_e - \frac{N}{2}}{\frac{N}{2}} \qquad (1)$$

Number of "essential" respondents

Total number of respondents

The experts involved in the review comprised 27 individuals, including university professors and lecturers specializing in computer education and educational technology, as well as in-service teachers and researchers. Based on the number of participating experts, the criterion for ensuring content validity was established at a CVR of .407 [40]. Table 3 presents detailed results of the expert evaluation, including response outcomes and research items, from the initial expert review.

TABLE Ⅲ
CVR VERIFICATION RESULT FOR EACH QUESTIONNAIRE

Division	Contents of question		Number of Responses (N=27)					M	SD	CVR
			5	4	3	2	1			
Purpose of development	1	Do you think the 'purpose of development' of the MCPS instructional model is appropriate?	12	15	0	0	0	4.4	0.51	1.00
Steps of class	2	It is appropriate to base the instructional steps of the study model on Papert's constructionism learning principles and the problem of improving computing classes.	16	9	1	1	0	4.5	0.75	0.85
	3	The instructional steps in the model are appropriate.	14	11	1	1	0	4.4	0.75	0.85
	4	The detailed instructional steps in step 1, 'Motivation and preparation,' are appropriate.	14	9	3	1	0	4.3	0.83	0.70
	5	The detailed lesson steps for Step 2, 'building Basic knowledge,' was set appropriately.	12	14	1	0	0	4.4	0.57	0.93
	6	the class steps for Step 3, 'apply basic knowledge,' are set appropriately.	12	13	2	0	0	4.4	0.63	0.85
	7	The class steps for Step 4, 'Planning Outputs and Execution,' are set appropriately.	15	11	1	0	0	4.5	0.58	0.93
	8	The class steps for 'Step 5 'development' are set appropriately.	15	10	2	0	0	4.5	0.64	0.85
Applicability	9	The study model is appropriate for use in a project-based programming class for non-majors.	10	14	2	1	0	4.2	0.75	0.78
	10	The study model would help design computing course content for non-majors.	14	11	2	0	0	4.4	0.64	0.85
Overall average										4.5

The results of the CVR validation showed that all the survey items were above .407, which ensured the overall validity of the items. The overall average was 4.5, which was a favorable evaluation result. However, the applicability (4.2) of item 9 showed a slightly lower response result. As an open-ended opinion related to this, it was observed that it could be difficult for non-majors and that step integration and simplification were necessary. It was also observed that this can be very effective for non-majors in a macroscopic aspect if the class difficulty and task level are well adjusted and operated by analyzing the learners' characteristics in step 1.

The validity of the MCPS instructional model was investigated through a first-order Delphi study—the results of descriptive opinions and open-ended responses to items were valid for all items. In addition, interviews were conducted with four university professors and researchers majoring in educational engineering and related to teaching-learning theory and instructional design to gather more in-depth and professional opinions from an educational engineering perspective on whether the development of the instructional model and the composition of the instructional design in this study were organized correctly to meet the goals.

To summarize the first and second expert opinions, it is necessary to distinguish the model development from existing models by clearly presenting the instructional model and instructional design components separately and providing detailed activities and strategies for each step. Additionally, it was noted that the study would be enhanced if the instructional design clearly outlined the instructional goals and evaluation methods and if these were reflected in the field application. Therefore, in this study on the development of the MCPS instructional model, an additional investigation was conducted to create a learner evaluation rubric based on the MCPS instructional model to enhance its comprehensiveness. This rubric considers content and evaluation aspects by explicitly outlining the lesson objectives.

4) Developing a Finalized Instructional Model: The instructional model was modified to reflect its validity verification, expert open review, and FGI survey results. The final version of the MCPS instructional model was developed, as shown in TABLE 4, by supplementing the teaching and learning activities and strategies.

TABLE IV
MAPS INSTRUCTIONAL MODEL

Steps	Details
Step 1. Motivation and preparation	1.1 Learning motivation (relevance to major, need for computing education)
	1.2 Identifying learner's characteristics and difficulties in computing education (online survey)
	1.3 Identifying the antecedents of computing education
Step 2. Building of basic knowledge	2.1 Learning motivation (find related current IT issues)
	2.2 Demonstration (explain programming grammar)
	2.3 Support (using real-life examples)
	2.4 Stop helping (extend practice)
Step 3. Application of basic knowledge	3.1 Learning motivation (build expectations by introducing computing skills)
	3.2 Problem settings
	3.3 Derive a solution to the problem
	3.4 Share solutions
Step 4. Planning outputs and execution	4.1 Exploring ideas
	4.2 Ideation
	4.3 Design and share ideas
	4.4 Implementing ideas
	4.4.1 Proofread ideas
	4.4.2 Share Concrete Outputs
	4.5 Evaluation
Step 5. Development	5.1 Learning motivation (identify challenges and confidence boosts)
	5.2 Applying digital technology
	5.3 Express ideas through various media

5) Differentiation from the Existing Instructional Model: The MCPS instructional model developed in this study has the following characteristics compared to previous studies.

Firstly, based on Papert's constructionism learning theory, learner level and pace were adjusted. The preparatory step was organized based on learner characteristics to provide learners with many opportunities to make choices and reflect on them so that non-majors can feel satisfaction and achievement.

Secondly, most of the existing instructional models only motivate students initially; the MCPS instructional model fosters continuous motivation throughout all steps. Furthermore, it offers activities such as finding relevance and providing related information from stage 2 to stage 5. By enabling non-major students to sustain and enhance their learning motivation at every stage—through reminders of their goals and progress checks—the significant issue of lack of motivation has been addressed and improved.

Thirdly, while most existing instructional models are designed for short-term (or single-session) classes, the MCPS instructional model is designed for semesters or longer by applying practical learning principles for non-majors.

Fourthly, in traditional computing classes, the product project typically follows the learning of programming, serving as a task or a means of evaluation. In contrast, the MCPS instructional model is designed to facilitate systematic learning by incorporating the entire process of planning and producing applications and project outputs within the framework of essential knowledge formation in the instructional model.

Fifthly, integrating extracurricular activities fosters a sense of challenge and self-confidence, with the instructional model extending beyond the mere production and evaluation of existing outputs. Moreover, it can be flexibly configured and operated to align with school events and schedules.

F. The Development of an Evaluation Rubric for the MCPS Instructional Model

1) Principles of Developing an Evaluation Rubric for the MCPS Model: Based on related studies [23], [34], [41], [42], [43], [44], [45], there are some principles for developing an evaluation rubric for the MCPS model. First, include cognitive and definitional domains, encompassing theory and practice, as well as individual and collaborative outputs. Second, based on process-oriented evaluation comprises formative evaluation (classroom practice, assignments), summative evaluation (short answer, narrative, practical evaluation, product plan, and report), and collaborative evaluation. Third, it is not limited to computational thinking skills. The process of producing output by solving real-life (major) related problems by themselves is organized and evaluated as an educational goal. Fourth, the scores are subdivided to enable reliable evaluation. Fifth, clear criteria should be provided based on the MCPS instructional model's step-by-step objectives.

2) Evaluation rubric factors: The steps of the MCPS lesson model were matched with the evaluation factors from the relevant research in columns. Rows are the objectives for each step in the MCPS instructional model. The evaluation factors for each step in the MCPS instructional model included knowledge, skill, and attitude domains, resulting in the evaluation factors shown in Table 5.

TABLE V
RESULTS OF DERIVING EVALUATION FACTORS FOR AN EVALUATION RUBRIC

MCPS Instructional Model Steps [6]	**Evaluation Factors**	**Knowledge**	**Skills**	**Attitudes**
Step 1. Motivation and preparation	1.1 Basic principles and concepts of computers	○		
	1.2 Recognition of the relevance of computing to major fields	○		○
	1.3 Design real-world computing algorithms	○	○	○
Step 2. Building of basic knowledge	2.1 The concept of programming Elements	○		
	2.2 Executing code with programming elements	○	○	
	2.3 Executing code to solve real-world problems	○	○	○
Step 3. Application of basic knowledge	3.1 Analyzing code to solve real-world(major-related) problems	○		
	3.2 Execute code to solve a real-world(major-related) problem	○	○	
	3.3 Expressing a Solution	○	○	○
Step 4. Planning outputs and execution	4.1 Exploring real-world problems	○		
	4.2 Appropriateness of data collection and analysis	○	○	
	4.3 Decompose the problem into solvable units	○	○	
	4.4 Design a solution	○	○	
	4.5 Implement the solution	○	○	
	4.6 Refine through testing and debugging	○	○	
	4.7 Feedback through sharing and collaboration	○	○	○
	4.8 Representing the results of collaborative output	○	○	○
Step 5. development	5.1 various expressions of ideas	○	○	○
	5.2 share your impressions of participating in computing classes	○	○	○

3) Rubric appropriateness review: A validity study was conducted with experts on the draft MCPS instructional model learner evaluation rubric. The validity analysis method was based on the CVR calculation formula [Equation (1)], and the CVR was set at .538 (number of experts: 13) according to the number of responding experts. We conducted a CVR review of the learner evaluation rubric development's evaluation factors and evaluation levels. First, we checked the validation results of the evaluation elements, and all of them were above .538, which ensured the validity of all items. However, after discussing the borderline value of Step 5.2, "Sharing experiences and reflections on participation in computing classes (0.54)", it was removed because it is part of the "Expressing ideas in a variety of ways" step in Step 5.1. The overall rating was 4.50, which is a positive result. The results of the expert review on the appropriateness of the evaluation level based on the evaluation factors showed that some items had a slightly lower CVR value of 0.54. Still, all survey items had a CVR value of .538 or higher, ensuring the validity of all items. Although the overall rating of 4.53 was positive, the ratings of Step 1.2, "Recognizing the relevance of computing to the major field (4th Industrial Revolution)," and Step 1.3, "Designing computing algorithms in real life," were somewhat lower than the other items. However, the ratings of 4.2 were positive; it can be observed that it is essential that non-majors have activities that are not only related to their major but also related to real life to motivate them. In addition, we observed that the evaluation of outputs should include the process and set evaluation criteria and that evaluation and feedback from instructors and evaluation of self-evaluation and peer evaluation are needed.

4) Develop a finalized version of the evaluation rubric: Based on the expert review comments above, the draft learner evaluation rubric was revised and refined to develop a finalized version. Step 1 was to evaluate learners' attitudes before programming classes and their ability to recognize the need for computing education and to write real-life algorithms. In Step 2, the students were asked to understand the concepts of programming elements, follow the code of real-life examples, and evaluate whether they could write and extend them by adding their ideas. Step 3 is constructing a solution to a real-world or major-related problem and presenting a finalized solution. Step 4, the output process was evaluated on whether the team explored and analyzed the idea, broke it down into solvable units, designed and implemented it, and evaluated the output together to provide feedback. Step 5 was organized to assess the attitude of reaffirming the need for computing and gaining confidence through sharing experiences and feelings about various expressions of ideas. The learner evaluation rubric for the MCPS instructional model developed in this study will offer a more objective and reliable assessment by furnishing a precise evaluation criterion that assesses both the process of learning programming and the completion of creative output.

G. Application of MCPS Instructional Models

1) Purpose and hypothesis: We developed and implemented a computing course utilizing the MCPS instructional model to enhance learning motivation and foster creative problem-solving skills in a college-level computing liberal arts class for non-majors. Subsequently, we evaluate its efficacy through result analysis. To validate the effectiveness via a comparison between the group utilizing the MCPS teaching model and the group undergoing traditional computing classes in liberal arts for non-majors. The research hypotheses for application and effectiveness analysis of the MCPS instructional model are as follows:

[Learning Motivation – Inter-group]

a. [Hypothesis A1] In a college-level computing liberal arts class for non-majors, the MCPS class model group will exhibit significant differences in intrinsic motivational goals compared to the group undergoing traditional computing classes.
b. [Hypothesis A2] In a college-level computing liberal arts class for non-majors, the MCPS class model group will demonstrate significant differences in extrinsic motivational goals compared to the group undergoing traditional computing classes.
c. [Hypothesis A3] In a college-level computing liberal arts class for non-majors, the MCPS class model group will show significant differences in the control of learning beliefs compared to the group undergoing traditional computing classes.
d. [Hypothesis A4] In a college-level computing liberal arts class for non-majors, the MCPS class model group will display significant differences in self-efficacy regarding learning motivation compared to the group undergoing traditional computing classes.

[Creative Problem-solving Skills – Inter-group]

a. [Hypothesis B1] In a college-level computing liberal arts class for non-majors, the MCPS class model group will present significant differences in specific areas of creative problem-solving skills such as knowledge, thinking, skills, and proficiency compared to the group undergoing traditional computing classes.
b. [Hypothesis B2] In a college-level computing liberal arts class for non-majors, the MCPS class model group will demonstrate significant differences in expansive thinking regarding creative problem-solving skills compared to the group undergoing traditional computing classes.
c. [Hypothesis B3] In a college-level computing liberal arts class for non-majors, the MCPS class model group will exhibit significant differences in critical and logical thinking regarding creative problem-solving skills compared to the group undergoing traditional computing classes.
d. [Hypothesis B4] In a college-level computing liberal arts class for non-majors, the MCPS class model group will manifest significant differences in motivational factors concerning creative problem-solving skills compared to the group undergoing traditional computing classes.

2) Application design: This study was conducted with non-major students enrolled in the computing course for non-majors at the University of A in South Korea. The participants comprised 91 students from four divisions, with 43 students in the experimental group and 48 in the control group. The intervention period they lasted from March 1st to December 21st, 2022, spanning 15 weeks each for the first and second semesters. Before the experiment, pre-tests for creative problem-solving skills and learning motivation were conducted to confirm the homogeneity of the experimental and control groups. Following the pre-tests, the experimental group received a computing course based on the MCPS class model, while the control group received traditional computing instruction. Traditional computing instruction refers to lecture-based teaching focusing on grammar and end-of-semester project-centered instruction. Post-tests were administered immediately after the course's completion, and this application's design is outlined in Table 6.

TABLE VI
APPLICATION DESIGN OF THE RESEARCH

G1	O1	X1	O3
G2	O2	X2	O4

G1: control group
G2: experimental group
O1, O2: pre-test (creative problem-solving ability test, learning motivation test)
X1: MCPS-based computing course
X2: traditional computing course
O3, O4: post-test (creative problem-solving ability test, learning motivation test)

3) Test Tool of Learning Motivation: For the learning motivation test, a test tool modified by Kang [46] based on the Motivation Strategies for Learning Questionnaire (MSLQ) produced by Pintrich and his colleagues [47] was used. The validity of this test paper was re-verified by one teacher and two educational engineering experts [12]. The sub-factors of the learning motivation test paper consist of 'Internal goals,' 'External goals,' 'Control of learning beliefs', and 'Self-efficacy.' There are 16 items, four for each sub-factor, and each item is presented on a 5-point Likert scale. Table 7 shows the sub-factors and number of items in the learning motivation test tool [15].

TABLE VII
SUB-FACTORS AND ITEMS OF THE LEARNING MOTIVATION TEST TOOL

Sub-Factor	**Question**	**Number of Questions**
internal goals	1, 10, 13, 14	4
external goals	5, 7, 9, 16	4
control of learning beliefs	2, 6, 11, 15	4
self-efficacy	3, 4, 8, 12	4
total	16	

4) Test Tool of Creative Problem-Solving Ability: In this study, the 'Simple Creative Problem-Solving Ability Test Paper' developed by the MI Research Team [48] of the Psychological Lab at Seoul National University was used [44]. This tool was developed based on the 'Simple Creative Problem-Solving ability test development study' [49] published by the Korea Educational Development Institute to measure creative problem-solving ability [34]. The sub-factors of the tool consist of 'knowledge in a specific area, the function of thinking, skill, and mastery,' 'divergent thinking,' 'critical and logical thinking,' and 'motivational elements. There are 20 items, five for each sub-factor, and each item is presented on a 5-point Likert scale. Table 8 shows the sub-factors and number of items in the creative problem-solving ability test tool [15].

TABLE VIII
SUB-FACTORS AND ITEMS OF THE CREATIVE PROBLEM-SOLVING ABILITY TEST TOOL

Sub-Factor	Question	Number of Questions
Knowledge in a specific area, the function of thinking, skill, and mastery	1, 2, 3, 4, 5	5
Divergent thinking	6, 7, 8, 9, 10	5
critical and logical thinking	11, 12, 13, 14, 15	5
motivational elements	16, 17, 18, 18, 20	5
total	20	

5) Extraction of Constructionism Learning Characteristics Factors based on Papert: In this study, to conduct causal analysis considering the constructionism learning characteristics of Papert, we utilized the Class Evaluation Items developed by our university. To extract the constructionism learning characteristics of Papert, we analyzed the validity of the factors by selecting 12 items out of a total of 15 items in the class evaluation questionnaire, excluding three self-evaluation items, as the subject of factor analysis. Statistical analysis was performed using SPSS 26, and exploratory factor analysis was conducted on the completed items through expert validation. The extraction method employed was Principal Component Analysis, and the rotation method used was varimax for factor analysis [50].

In the third round of factor extraction, following the first and second rounds, the Kaiser-Meyer-Olkin (KMO) measure yielded a value of .850 (> .5), indicating adequacy, and Bartlett's test of sphericity resulted in an approximate chi-square value (significance probability) of 705.178 (.000), confirming suitability. A total of 6 items were identified, with each factor comprising three items, demonstrating the most appropriate conclusion. Additionally, high reliability was ensured for Factor 1 and 2, with values of .987 and .990, respectively.

Considering the content and evaluation areas of the items ultimately derived from exploratory factor analysis, and after review by three experts, the factors were named ‘interest-based problem-solving’ and ‘communication using digital technology.’ Furthermore, a Cronbach's alpha value of .991 was obtained through reliability analysis, indicating very high reliability. The constructivist learning characteristics factors and item composition utilized in this study are outlined in Table 9.

TABLE IX
CHARACTERISTICS AND ITEMS OF PAPERT'S CONSTRUCTIONIST LEARNING

Sub-Factor	Question	Number of Questions	Cronbach-α
interest-based problem-solving	11, 14, 15	3	.991
communication using digital technology	4, 9, 10	3	
total	20		

6) Instructional Implementation: In this study, to validate the MCPS instructional model's effectiveness, we developed a computing course based on the MCPS model and conducted expert validation. We intend to apply for this course to the experimental group. The control group was designed to follow the traditional university computing course format. To compare the courses between the experimental and control groups, we outlined the weekly course designs for each group, as shown in Table 10.

TABLE X
COMPARISON OF LESSON PLANS BY WEEK BETWEEN THE EXPERIMENTAL GROUP AND THE CONTROL GROUP

Experimental Group (N=43) (MCPS-based Computing Course)		Week (3 h)	Control Group (N=48) (Traditional Computing Course)	
Application of MCPS Principles	Course Topic (App Inventor Programming)		Course Topic (App Inventor Programming)	Teaching Method
[Step 1] • Maintain overall motivation • Identify learner characteristics	Orientation and 4th industrial revolution, (online survey)	1	Orientation and 4th Industrial Revolution, introduction to App inventor	‣ lecture method+ basic examples
• Pre-existing understanding of computing	Basic Examples: understanding App inventor's essential functions and understanding of variables (understanding computing)	2	Understanding app inventor's essential functions and understanding of variables	
[Step 2] • Find motivation and relevance	Understanding conditions	3	Understanding conditions	
• Adaptation of learner characteristics from step 1 -> adjusting learning pace	Understanding lists	4	Understanding lists	
and level	Understanding loops	5	Understanding loops	
• Explanation of syntax followed by practice	Understanding functions	6	Understanding functions	
• Demonstrating real-life examples	Understanding other functions	7	Understanding other functions	
• Extend with additional ideas	Understanding level check (midterm exam)	8	Understanding level check (Midterm exam)	
[Step 3] • Find motivation and relevance • Set everyday life (major-related) problems • Find solutions independently • Share solutions	Solving everyday life and major-related problems	9	Solving everyday life and major-related problems	‣ Lecture method + advanced examples (including real-life related topics)
[Step 4]	Creating apps using maps (brainstorming ideas)	10	Creating apps using sensors	

Experimental Group (N=43) (MCPS-based Computing Course)		Week (3 h)	Control Group (N=48) (Traditional Computing Course)	
Application of MCPS Principles	**Course Topic (App Inventor Programming)**		**Course Topic (App Inventor Programming)**	**Teaching Method**
• Form teams based on learner opinions • Implementing creative outputs by exploring ideas and solving problems • [Example: 2nd class, output production 1st class conducted together or example: 1st class, output production 2nd class] • [Submit step-by-step plans and reports]	Creating apps using maps creating quiz apps (design and share ideas, for example)	11	creating apps using maps	
	creating quiz apps (design and share ideas, for example)	12	Creating quiz apps	
	Creating apps for convenience in daily life (Implementing ideas)	13	Creating apps for convenience in daily life	
	Evaluate outputs (instructor evaluation, peer evaluation, self-evaluation)	14	Apps for college students (Output production)	‣ Problem-based learning ‣ Project-based learning (team)
[Step 5] • IoT demonstration • Participation in campus exhibitions • Computing competitions, etc.	Spreading ideas	15	Evaluate outputs (instructor evaluation, peer evaluation, self-evaluation)	

In the experimental group's initial session, an online survey was conducted to analyze learner characteristics. Based on the results of this analysis, adjustments were made to learner level, learning speed, and task difficulty starting from MCPS class stage 2. Additionally, a team was formed incorporating learner feedback, and the class environment was tailored to address learner difficulties and preferences before each session [51].

Comparing the characteristics of the two groups, during the first and second weeks, the experimental group discussed the necessity of computing and computer comprehension as part of a motivation and preparation step. An online survey was also conducted to analyze learner characteristics. In contrast, the comparison group proceeded directly to the main class after an orientation session.

From weeks 3 to 8, both groups focused on learning programming grammar. The experimental group adjusted their approach based on learner characteristics, presenting topics that overlapped with consideration of learning levels, speeds, and task difficulties. Additionally, they engaged in activities before each class to connect with current IT issues. The instructional approach involved principles of lecture methods for explaining and summarizing grammar and cognitive apprenticeship principles for demonstration and extended activities based on real-life examples. The comparison group followed a more traditional grammar-centered lecture method with basic examples.

Weeks 9 to 14 saw the experimental group engaging in practical exercises to solve real-life or significant problems. They explored ideas, designed, implemented, and evaluated solutions independently, fostering creativity through team collaboration, opinion sharing, and feedback. Meanwhile, the comparison group focused on in-depth examples followed by producing output, mainly through problem-based learning sessions addressing end-of-semester projects.

The final 15th week it involved a developmental stage and idea dissemination activities for the experimental group (such as school software exhibitions, idea contests, IoT demonstrations, etc.). In contrast, the comparison group underwent project evaluation.

III. Results and Discussion

A. *Comparison of Pre-Post Tests*

Data analysis and processing in this study were conducted using IBM SPSS version 26. To verify the hypothesis of this application, a t-test was performed to confirm the homogeneity of the experimental and control groups' learning motivation and creative problem-solving ability before the experiment. The pre-test results for learning motivation and innovative problem-solving ability showed that the control and experimental groups had similar average values, and the significance levels for each factor in the two groups were all above .05, showing no significant differences between the groups. Therefore, it was confirmed that the control and experimental groups were homogeneous in terms of learning motivation. Table 11 shows the results of the Learning Motivation pretest.

TABLE XI
Results of pre-test for learning motivation(experimental n=43, control n=48)

Sub-Factor	Group	M	SD	T	P
Internal goals	Experimental group	3.56	.557	-1.032	.305
	Control group	3.69	.626		
External goals	Experimental group	3.69	.638	-.147	.884
	Control group	3.71	.758		
Control of learning beliefs	Experimental group	3.70	.423	-.486	.628
	Control group	3.76	.571		
self-efficacy	Experimental group	3.01	.612	.246	.806
	Control group	2.97	.819		

After conducting the pre-test for creative problem-solving ability, it was observed that the control group had a slightly higher overall average value than the experimental group. However, the significance level for each factor was above .05, indicating no significant differences between the groups. Consequently, it was confirmed that the control and experimental groups were homogeneous in terms of creative problem-solving ability. The results of the creative problem-solving ability pre-test are presented in Table 12.

TABLE XII
Results of the pre-test for creative problem-solving ability(experimental n=43, control n=48)

Sub-Factor	Group	M	SD	T	P
Knowledge in a specific area, the	Experimental group	2.71	.597	-.896	.373

function of thinking, skill, and mastery	Control group	2.83	.648		
divergent thinking	Experimental group	2.73	.716	-.074	.941
	Control group	2.74	.745		
Critical and logical thinking	Experimental group	3.47	.453	-1.378	.172
	Control group	3.62	.576		
Motivational elements	Experimental group	3.17	.493	-.617	.539
	Control group	3.25	.684		

1) Comparison of Post-Test Results between Groups: To confirm whether there are significant differences in learning motivation and creative problem-solving ability between the experimental group and the control group after the experimental treatment, a post-test was conducted, and the results comparing the post-test of the experimental group and the control group using a g independent t-test are shown in Table 13.

TABLE XIII
COMPARISON OF POST-TEST RESULTS OF THE EXPERIMENTAL AND CONTROL GROUPS (EXPERIMENTAL N=43, CONTROL N=48)

Sub-Factor		Paired	M	SD	T	P
Learning motivation	internal goals	experimental group	4.02	.733	2.111	**.038***
		control group	3.68	.777		
	external goals	experimental group	3.70	.856	-.747	.457
		control group	3.83	.735		
	control of learning beliefs	experimental group	4.02	.486	.370	.712
		control group	3.97	.617		
	self-efficacy	pre	3.43	.867	.573	.568
		post	3.33	.746		
Creative problem-solving ability	knowledge in a specific area, the function of thinking, skill, and mastery	experimental group	3.14	.591	.507	.613
		control group	3.07	.690		
	divergent thinking	experimental group	3.27	.765	.765	.446
		control group	3.15	.778		
	critical and logical thinking	experimental group	3.82	.541	-.857	.394
		control group	3.91	.503		
	motivational elements	experimental group	3.51	.658	-.181	.857
		control group	3.53	.727		

As a result of the post-test conducted between groups, the average score of the experimental group was generally higher than that of the control group. However, this difference was statistically significant only in the internal goal area among the sub-factors of learning motivation, where the p-value was below the significance level of .05. This finding indicates a significant difference in learning motivation, specifically in the internal goal area among students who participated in MCPS model-based computing classes compared to those who took traditional computing classes. Nevertheless, no significant changes were observed in other learning-motivation sub-factors and in the creative problem-solving ability sub-factors based on the post-test results between groups. Consequently, to further examine the average scores within the experimental group across different factors, pre- and post-tests within the group were compared, and additional analysis was conducted on the average score of the experimental group.

2) Comparison of Pre-Post Test Results Within Groups: As previously mentioned, to examine the changes in the sub-factors of learning motivation and creative problem-solving ability within each group of the control and experimental groups, additional paired sample t-tests were conducted. In this regard, the following research hypothesis was formulated to verify the effectiveness through comparison within the group applying the MCPS instructional model in computing general education classes for non-majors.

[Learning Motivation – Within-group]

a. [Hypothesis C1] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in internal goal orientation of learning motivation.
b. [Hypothesis C2] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in external goal orientation of learning motivation.
c. [Hypothesis C3] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in perceived control of learning beliefs.
d. [Hypothesis C4] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in self-efficacy of learning motivation

[Creative Problem-solving Skills – Within-group]

a. [Hypothesis D1] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in specific areas of knowledge, thinking, skills, and expertise related to creative problem-solving ability.
b. [Hypothesis D2] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in divergent thinking of creative problem-solving ability.
c. [Hypothesis D3] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in critical and logical thinking of creative problem-solving ability.
d. [Hypothesis D4] The group of non-majors in university computing liberal arts classes where the MCPS instructional model is applied will exhibit a significant difference in motivational factors of creative problem-solving ability.

First, paired sample t-tests were conducted on the control group's pre-test and post-test results, which are shown in Table 14. The comparison of pre-test and post-test results for each evaluation within the control group revealed a slight increase in mean values for all sub-domains of the learning motivation evaluation, excluding the internal goal orientation. Additionally, within the sub-domains of learning motivation, a statistically significant change was observed in the self-efficacy domain, and within the sub-domains of creative problem-solving ability evaluation, significant changes were observed in the divergent thinking and critical/logical thinking domains, with p-values of .033, .039, and .033, respectively, indicating significance at the .05 level. The results of the paired sample t-tests for the experimental group are presented in Table 15.

TABLE XIV
COMPARISON OF PRE-TEST AND POST-TEST RESULTS IN THE CONTROL GROUP (CONTROL N=48)

Sub-Factor		Paired	M	SD	T	P
Learning motivation	Internal goals	pre	3.74	.633	.453	.653
		post	3.68	.777		
	External goals	pre	3.73	.774	-.628	.533
		post	3.83	.735		
	Control of learning beliefs	pre	3.80	.599	-1.644	.107
		post	3.97	.617		
	self-efficacy	pre	3.03	.837	-2.191	**.033***
		post	3.33	.746		
Creative problem-solving ability active	Knowledge in a specific area, the function of thinking, skill, and mastery	pre	2.90	.670	-1.323	.192
		post	3.07	.690		
	Divergent thinking	pre	2.83	.779	-2.118	**.039***
		post	3.15	.778		
	Critical and logical thinking	pre	3.67	.615	-2.191	**.033***
		post	3.91	.503		
	Motivational elements	pre	3.30	.707	-1.468	.149
		post	3.53	.727		

TABLE XV
COMPARISON OF PRE-TEST AND POST-TEST RESULTS IN THE EXPERIMENTAL GROUP (CONTROL N=43)

Sub-Factor		Paired	M	SD	T	P
Learning motivation	internal goals	pre	3.59	.582	-2.771	**.008****
		post	4.02	.733		
	external goals	pre	3.71	.659	.034	.973
		post	3.70	.856		
	control of learning beliefs	pre	3.77	.471	-2.437	**.019***
		post	4.02	.486		
	self-efficacy	pre	3.08	.644	-2.237	**.031***
		post	3.43	.867		
Creative problem-solving ability	knowledge in a specific area, function of thinking, skill, and mastery	pre	2.73	.587	-3.224	**.002****
		post	3.14	.591		
	divergent thinking	pre	2.77	.706	-2.981	**.005****
		post	3.27	.765		
	critical and logical thinking	pre	3.47	.458	-3.288	**.002****
		post	3.82	.541		
	motivational elements	pre	3.19	.458	-2.294	**.027***
		post	3.51	.658		

The comparison of pre-test and post-test results for each evaluation in the experimental group revealed a significant increase in mean values across all domains compared to the control group, excluding the external goal orientation domain of the learning motivation evaluation. Furthermore, within the sub-domains of the learning motivation evaluation, excluding the external goal orientation, statistically significant changes were observed in three areas with p-values of .008, .019, and .031, respectively, indicating significance at the .05 level. Additionally, within the sub-domains of the creative problem-solving ability evaluation, significant changes were observed across all domains with p-values of .002, .005, .002, and .027, respectively, indicating significance at the .05 level.

Although no significant difference was found in the post-hoc comparison analysis between groups for creative problem-solving ability, the pre-test and post-test comparison within the experimental group revealed a significant increase in mean values across all sub-domains, confirming significant changes.

B. *Interpretation of Test Comparison Results and Discussion*

The comparison results between groups for learning motivation and creative problem-solving ability evaluation, as well as factors that showed statistically significant improvements in pre-test and post-test comparisons within groups, were analyzed.

Firstly, focusing on learning motivation, the analysis of comparisons for each sub-factor between groups revealed significant improvements in the internal goal orientation area in the post-test comparison. Furthermore, in the pre-test and post-test comparison within groups, statistically significant improvements were observed in all sub-domains, excluding the external goal orientation, for the experimental group.

Similarly, when examining the results for creative problem-solving ability, while no statistically significant differences were found in the post-test comparison between groups, significant improvements were observed in all sub-domains within the experimental group in the pre-test and post-test comparison.

Therefore, [Hypothesis A1] was accepted, indicating that the group of non-majors in university computing general education classes where the MCPS instructional model was applied showed a significant difference in internal goal orientation of learning motivation compared to the group where traditional computing classes were used. However, [Hypotheses A2-A4] and [Hypotheses B1-B4] were rejected, indicating no significant differences. Additionally, upon further analysis of the changes in sub-factors of learning motivation and creative problem-solving ability within groups, [Hypothesis C2] was rejected, suggesting no significant difference, while [Hypotheses C1], [C3-C4], [D1-D4] were all accepted, indicating substantial differences.

In summary, the MCPS instructional model-based computing education for non-majors in computing general education classes demonstrated effectiveness in internal goal orientation of learning motivation, perceived control in learning beliefs, self-efficacy, and all sub-factors of creative problem-solving ability according to within-group comparison results. However, although the post-test comparison between groups revealed somewhat more significant results in the experimental group's sub-factors compared to the control group, with only the internal goal orientation factor of learning motivation showing substantial difference, indicating the need for further analysis with diversified methods to understand the relationship among factors within the experimental group.

C. *Analysis of Relationships between Each Factor in the Application of the MCPS Instructional Model*

1) First-order correlation analysis between subfactors of each test: This study verified the effectiveness of the MCPS instructional model for operating computing general education classes for non-majors in university settings. In addition to assessing effectiveness, the study aimed to derive various implications by conducting in-depth analyses of correlations and causality among sub-factors. The sub-domains of the learning motivation and creative problem-solving ability evaluation tools used in this study were analyzed, focusing on seven factors: three sub-factors of learning motivation (excluding the external goal orientation) and four sub-factors of creative problem-solving ability, where statistically significant differences were not observed in the experimental group. Pearson correlation coefficients were calculated to analyze the primary correlations among these sub-factors. Upon examining the evaluation results, it was observed that there were significant correlations between two sub-factors of learning motivation (internal goal orientation, self-efficacy) and four factors of creative problem-solving ability (specific area knowledge/thinking/skills and expertise, divergent thinking, critical/logical thinking, motivational factors). Therefore, additional analysis was conducted to investigate the relationships between these sub-factors further.

2) First-order causal analysis between each sub-factor: To examine the relationship between the sub-factors of learning motivation and creative problem-solving ability, a primary causal relationship analysis was conducted. Two sub-domains of learning motivation (internal goal orientation, self-efficacy) were set as independent variables, while four sub-domains of creative problem-solving ability were set as dependent variables [12], [52], [53], [54], [55], [56]. The following research hypotheses were formulated to determine whether there exists any causal relationship between learning motivation and creative problem-solving ability in the group of non-majors in university computing general education classes where the MCPS instructional model was applied.

[Primary Causal Analysis - Multiple Regression Analysis]

a. [Hypothesis E1]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between internal goal orientation of learning motivation and specific area knowledge/thinking/skills and expertise of creative problem-solving ability.
b. [Hypothesis E2]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between internal goal orientation of learning motivation and divergent thinking of creative problem-solving ability.
c. [Hypothesis E3]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between internal goal orientation of learning motivation and critical/logical thinking of creative problem-solving ability.
d. [Hypothesis E4]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between internal goal orientation of learning motivation and motivational factors of creative problem-solving ability.
e. [Hypothesis E5]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between self-efficacy of learning motivation, specific area knowledge/thinking/skills, and expertise of creative problem-solving ability.
f. [Hypothesis E6]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between self-efficacy of learning motivation and divergent thinking of creative problem-solving ability.
g. [Hypothesis E7]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between self-efficacy of learning motivation and critical/logical thinking of creative problem-solving ability.
h. [Hypothesis E8]: The MCPS instructional model-based computing education for non-majors in university computing classes will have a causal relationship between self-efficacy of learning motivation and motivational factors of creative problem-solving ability.

To validate the research model, stepwise regression analysis was employed to distinguish factors influencing the dependent variable and to present only the independent variables that had an effect. The aim was to analyze the impact of two areas of learning motivation, set as independent variables, on creative problem-solving ability.

The results were examined after conducting multiple regression analyses on the influence of learning motivation on creative problem-solving ability. The independent variables, internal goal orientation, and self-efficacy, significantly influenced the dependent variables: specific area knowledge/thinking/skills and expertise, divergent thinking, critical/logical thinking, and motivational factors, with p-values of .000 ($p < .05$). All other variables were deemed insignificant and were consequently eliminated through backward elimination. This regression equation demonstrates explanatory power of 33%, 47.1%, 22.8%, and 23.9% for motivational factors.

Thus, analyzing the causal relationships between the seven factors of learning motivation and creative problem-solving ability (as described in hypotheses E1 to E8), it was found that internal goal orientation influenced specific area knowledge/thinking/skills and expertise, as well as divergent thinking. Conversely, critical/logical thinking and motivational factors were influenced by self-efficacy.

3) Secondary correlation analysis between each sub-factor: As observed earlier, the causal relationships between the two factors of learning motivation (internal goal orientation, self-efficacy) and the four factors of creative problem-solving ability (specific area knowledge/thinking/skills and expertise, divergent thinking, critical/logical thinking, motivational factors) were examined in this study. To determine whether Papert's constructionism learning characteristics, consisting of two factors (problem-solving based on interest and understanding, digital-based communication), mediate these relationships, Pearson correlation coefficients were calculated to analyze the secondary correlations among the eight factors.

The results indicated significant correlations between Papert's constructionism learning characteristics, one factor of learning motivation (internal goal orientation), and three factors of creative problem-solving ability (specific area knowledge/thinking/skills and expertise, divergent thinking, motivational factors). Hence, additional analysis was conducted to investigate the relationships between these sub-factors further.

4) Secondary causal analysis between each sub-factor: In the earlier first-order causal analysis, a causal relationship was established between learning motivation (internal goals, self-efficacy) and creative problem-solving ability (specific area knowledge/thinking/skills, divergent thinking, motivational factors). Subsequently, a mediation regression analysis was conducted to examine whether there is a mediating effect of Papert's constructionism learning characteristics (problem-solving based on interest and understanding, digital-based communication) on the causal relationship between learning motivation and creative problem-solving ability.

The hypotheses set to examine the influence of Papert's constructionism learning characteristics (problem-solving based on interest and understanding, digital-based communication) are as follows:
[Secondary Causal Analysis - Mediation Regression Analysis]

a. [Hypothesis F1]: In the MCPS instructional model-based computing education for non-majors, the statistical impact of internal goal orientation of learning motivation on specific area knowledge/thinking/skills and expertise of creative problem-solving ability will be mediated by problem-solving based on interest and understanding.
b. [Hypothesis F2]: In the MCPS instructional model-based computing education for non-majors, the statistical impact of internal goal orientation of learning motivation on divergent thinking of creative problem-solving ability will be mediated by problem-solving based on interest and understanding.
c. [Hypothesis F3]: In the MCPS instructional model-based computing education for non-majors, the statistical impact of internal goal orientation of learning motivation on specific area knowledge/thinking/skills and expertise of creative problem-solving ability will be mediated by digital-based communication.
d. [Hypothesis F4]: In the MCPS instructional model-based computing education for non-majors, the statistical impact of internal goal orientation of learning motivation on divergent thinking of creative problem-solving ability will be mediated by digital-based communication.

The results of the mediation regression analysis indicate that the mediator variable 2, ‘digital-based communication, has a complete mediating effect on the dependent variable 1, specific domain knowledge, thinking, skills, and mastery. The Barron and Kenny approach distinguishes between partial and complete mediation effects and is commonly used by many researchers [57]. However, it is weak in that it does not directly test the significance of the indirect effects but rather infers based on a series of tests. Hayes' bootstrapping method was employed to verify the mediator variables' indirect effects. Bootstrapping is a technique for estimating the distribution of parameters based on sample data when the population distribution is unknown. Indirect effects are considered significant at the 5% significance level if the 95% confidence interval does not include zero. Therefore, it can be interpreted that all paths have mediating effects [58]. The results of the indirect effects verification are shown in Table 16.

TABLE XVI
INDIRECT EFFECT VERIFICATION RESULTS (BOOTSTRAPPING)

Route	Indirect Effect			
	Effect	BootSE	Boot LLCI	Boot ULCI
Internal goal → interest-based problem solving → specific domain knowledge, thinking, skills, and mastery	.2563	.2242	.0439	.8640
Internal goal → interest-based problem solving → dispersive thinking	.3021	.3012	.0276	1.1198
Internal goal → digital-based communication → specific domain knowledge, thinking, skills, and mastery	.3026	.1470	.0181	.6179
Internal goal → digital-based communication → dispersive thinking	.5050	.1921	.0583	.8481

Therefore, these results indicate that MCPS-based computing education influences specific domain knowledge, thinking, skills, mastery, and divergent thinking through higher internal goals via interest-based problem-solving and digital-based communication.

D. *Interpretation of causal relationship analysis results and Discussion*

The causal relationship analysis results among motivation, creative problem-solving ability, and Papert's constructionism Figure 1 summarizes learning characteristics factors. In summary, the internal goal of motivation partially mediates the influence of interest-based problem-solving on specific domain knowledge, thinking, skills, mastery, and divergent thinking within creative problem-solving. Additionally, digital-based communication fully mediates this relationship. Moreover, motivation self-efficacy was observed to affect the critical and logical thinking and motivational aspects of creative problem-solving.

Therefore, the hypothesis testing results for the 1st causal analysis-multiple regression analysis are as follows: [Hypothesis E1], [Hypothesis E2] was accepted, indicating that in non-major students, there is a causal relationship between the internal goal of motivation and specific domain knowledge, thinking, skills, and mastery, as well as divergent thinking within creative problem-solving in MCPS-based computing education. Additionally, [Hypothesis E7] and [Hypothesis E8] were accepted, suggesting that in non-major students, there is a causal relationship between the self-efficacy of motivation and critical, logical thinking and motivational aspects of creative problem-solving in MCPS-based computing education. Furthermore, [Hypotheses E3 ~ E6] were all rejected, indicating no significant difference.

The results of the 2nd causal analysis - mediation regression analysis are as follows: [Hypothesis F1], [Hypothesis F2] was accepted, indicating that in non-major students, there is a statistical mediation effect (partial) of interest-based problem-solving on the influence of the internal goal of motivation on specific domain knowledge, thinking, skills, and mastery within creative problem-solving in MCPS-based computing education. Additionally, [Hypothesis F3] [Hypothesis F4] was accepted, suggesting that in non-major students, there is a statistical mediation effect (complete) of digital-based communication on the influence of the internal goal of motivation on specific domain knowledge, thinking, skills, and mastery, as well as divergent thinking within creative problem-solving in MCPS-based computing education.

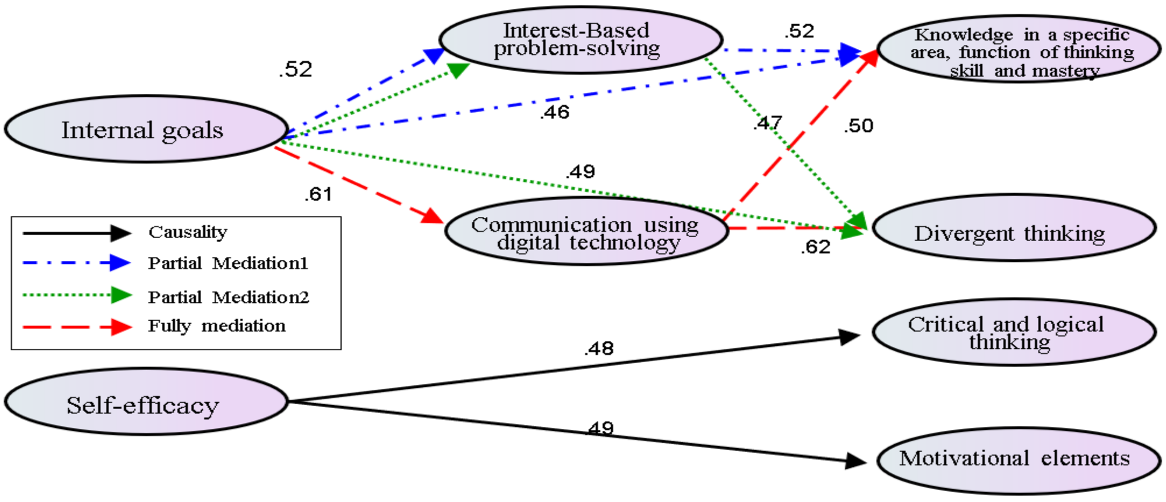
Interest-Based problem-solving
Knowledge in a specific area, function of thinking skill and mastery
Internal goals
Communication using digital technology
Divergent thinking
Critical and logical thinking
Self-efficacy
Motivational elements
Causality
Partial Mediation1
Partial Mediation2
Fully mediation
.52
.52
.46
.47
.50
.49
.61
.62
.48
.49

Fig. 1 Results of causal relationship analysis between sub-factors

To analyze this specifically, to foster specific domain knowledge, thinking, skills, mastery, and divergent thinking, students should be encouraged to choose challenging tasks and persistently perform them, considering learning itself as a reward, thus allowing for the cultivation of intrinsic goals. The learning environment should be structured to respect students' opinions and emphasize learning activities based on their interests and understanding, utilizing digital technology for communication processes. Such an environment can be more effective in fostering these skills.

Moreover, teaching and learning methods and environments should be designed to cultivate critical and logical thinking and motivational factors to provide satisfaction, interest, successful experiences, and positive feedback to develop self-efficacy. Additionally, in Papert's constructionism evaluation-based computing education, active problem-solving based on interest and understanding, along with smooth communication through digital tools, can effectively mediate motivation and creative problem-solving abilities [50], [59], [60].

Therefore, creating a learning environment that encourages intrinsic goals, respects students' opinions, utilizes digital technology for communication, provides challenging tasks, offers positive feedback, and facilitates active problem-solving can be effective in fostering specific domain knowledge, thinking, skills, and mastery, as well as divergent thinking, critical and logical thinking, and motivational factors in computing education.

E. *Rubric evaluation results of the experimental group*

In this study, we aim to examine the impact of MCPS (Modeling, Computing, Problem-Solving) instructional framework-based computing classes on non-major students' motivation and creative problem-solving abilities. We will analyze the evaluation rubric results of the experimental group to evaluate their performance. The evaluation rubric results of the experimental group are presented in Table 17.

TABLE XVII
EVALUATION RUBRIC RESULTS OF THE EXPERIMENTAL GROUP

MCPS Steps[6]	**Evaluation Type**	**Evaluation Average**	**Evaluation average**	**Points**
Step 1. Motivation and preparation	Midterm (Written)	1.1 Basic principles and concepts of computers	4.19	5
	Assignments & Feedback	1.2 Recognition of the relevance of computing to major fields	4.25	5
	Training	1.3 Design real-world computing algorithms	4.07	5
Step2. Building of basic knowledge	Midterm exam (written + practical)	2.1 The Concept of Programming Elements	4.70	5
		2.2 Executing code with programming elements	4.08	5
		2.3 Executing code to solve real-world problems	**3.84**	5
Step 3. Application of basic knowledge	Lab sessions + assignments	3.1 Analyzing code to solve real-world(major-related) problems	7.52	10
		3.2 Execute code to solve a real-world(major-related) problem		

MCPS Steps[6]	**Evaluation Type**	**Evaluation Average**	**Evaluation average**	**Points**
		3.3 Expressing a Solution	**3**	5
Step 4. Planning outputs and execution	Project evaluation (proposal, report)	4.1 Exploring real-world problems	4.89	5
		4.2 Appropriateness of data collection and analysis		
		4.3 Decompose the problem into solvable units	4.89	5
		4.4 Design a solution		
		4.5 Implement the solution	4.03	5
		4.6 Refine through testing and debugging.		
		4.7 Feedback through sharing and collaboration	4.33	5
		4.8 Representing the results of collaborative output	90.20	100
Step 5. Development	Assignments	5.1 Various expressions of ideas	4.	5
Overall Average			4.27	

Upon reviewing the evaluation rubric results of the experimental group, it was found that the overall average score was 4.27 out of 5 points, indicating positive outcomes. However, differences were observed across different stages. Among them, the score for Stage 2, '2.3 Execution of Code for Real-life Problem Solutions (3.84 points)', was the lowest, followed by Stage 3, '3.3 Expression of Problem Solutions (3.95 points)'.

This suggests that while students grasp programming concepts, they may encounter difficulty executing code or expressing solutions. Nevertheless, excluding Stage 4, '4.8 Expression of Collaborative Outputs', the average score was 4.54 points, indicating overall positive outcomes. This can be attributed to continuous communication and feedback among team members while collaborating to express outputs. Furthermore, the average scores for real-life applications and problem solutions improved as stages progressed, indicating a positive influence of the course stages on expressing problem solutions. The lower score for 'Diverse Expression of Ideas (4.0 points)' in Stage 5 was attributed to non-submissions, resulting in a slightly lower average score.

F. Analysis of Lecture evaluation Results and Project Impressions

1) Analysis of Course Evaluation Results: After conducting the computing education, the student's overall satisfaction with the course evaluation results was higher than the overall course average in all categories. Additionally, the ratings from the experimental group were higher than those from the control group across all items. Notably, the items regarding the 'consistency and progress of course planning and content,' 'instructor's preparation and teaching ability,' and 'communication between instructor and learners' scored above 4.50, indicating high satisfaction levels.

This indicates that, compared to other general education courses at the university, computing education received higher overall learner satisfaction. Furthermore, it suggests that MCPS-based computing education had a more positive impact on learner satisfaction. Additionally, it reaffirms the importance of teaching methods and the role of instructors in learner-centered constructionism-based computing education courses.

2) Keyword Analysis and Word Cloud of Project Impressions: Next, for qualitative analysis, a word cloud was generated using visualization techniques to easily visualize the key themes extracted from the reflections in the project reports. To achieve this, a part-of-speech tagging module was employed in Python programming to assign parts of speech, considering the meaning and context of the content. Only nouns were extracted, and word frequency was calculated to represent the occurrence of each word.

Among the extracted keywords, 'coding' appeared most frequently, with 30 occurrences, followed by 'class' (29 occurrences), 'project' (25 occurrences), and 'thought' (21 occurrences). Word frequency was calculated, and the word cloud generation results were visualized for straightforward interpretation. As shown in Figure 2, keywords such as coding, project, and program were central, surrounded by terms like interest, solution, experience, interest, opportunity, collaboration, achievement, and difficulty, reflecting the sentiments expressed in the reflections.

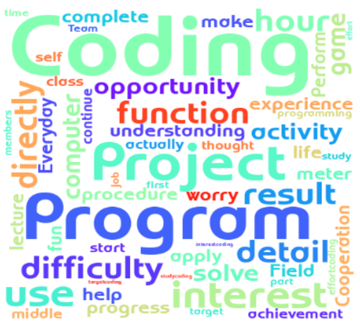

Fig. 2 Word visualization

3) Topic analysis of project impressions: Topic modeling was employed to extract and analyze topics from the reflections on the project outcomes and subjective opinions on course evaluations of the experimental group. Topic Modeling is a technique used to extract critical themes (topics) from text-based document data, classify (cluster), and analyze documents based on the extracted topics. Latent

Dirichlet Allocation (LDA) topic modeling provided by the gensim package in Python was used for topic analysis. LDA is a prominent machine learning-based technique for topic modeling that utilizes the Dirichlet distribution to infer latent topics present in the given documents [61].

In this study, the results of analyzing the reflections on the project outcomes and subjective opinions on course evaluations led to the extraction of topics. The number of topics to be extracted (k=4) and the number of key words to constitute each topic (word=15) were set. Consequently, the main 15 words constituting each topic were output along with their relevance scores. Table 18 presents the results of the topic analysis obtained using the lda_model.print_topics() function and the visualization results from pyLDAvis, providing the final topic analysis results regarding the topic labels.

TABLE XVIII
COMPREHENSIVE TOPIC ANALYSIS RESULTS

Topic Analysis Results (lda_model.print_topics) and Visualization Results by pyLDAvis			
Topic number	**Topic-specific words**	**Token Distribution Ratio**	**Final topic label**
A	Thinking, coding, program, part, project, problem, utilization, programming, first, use, function, computer, though, class, course, etc.	41%	Digitally based idea programming
B	Problem solving, assignment, help, topic. start. learning. creative. apply. study. role. progress. subject. Thinking, flow, technology, etc.	25%	Helps with creative problem-solving skills
C	Activity, motivation, cooperation, knowledge, answer, question, understanding, environment, improvement, lesson, solution, student, progress, fun, etc.	18.1%	Increased motivation
D	logic, once, structure, again, game, difficulty, collaboration, everyone, for, understanding, existence, bug, part, mood, teammate, interest, etc.	15.9%	Solving difficulties through collaboration and classes based on interest and understanding

In this study, four topics were extracted from the reflections on project reports and course evaluations submitted by students who took the computing class based on the MCPS teaching model. The final topic labels provide insights into students' perceptions and evaluations of the course.

First, the topic related to ‘Digital-based idea programming’ includes words such as thoughts, coding, programming, computer, problem, solution, utilization, and functionality. It was observed that initially, students might find it challenging to express their ideas through programming, but over time, they become more familiar with digital tools and find them helpful for learning more accurately and deeply.

Second, the topic associated with ‘Enhancing creative problem-solving skills’ includes words like creativity, problem-solving, assistance, programming, learning, thoughts, study, progress, and thinking. Students expressed that exploring and implementing solutions to real-life problems independently, finding and applying unfamiliar functions, and continuously improving the app even after completion enhanced their fluid thinking and provided a refreshing experience.

Third, the topic related to ‘Improving motivation’ includes words such as class, activities, collaboration, progress, solution, motivation, enhancement, outcomes, and utilization. Students indicated that they gained awareness of the importance of computing, demonstrated a sense of accomplishment and determination in producing outputs, and expressed expectations for further development by integrating computing into their majors, leading to a desire to learn more computing courses in the future.

Fourth, the topic concerning ‘Collaborative problem-solving and interest-based learning’ includes words like logic, team members, solution, difficulties, problem, outcomes, thoughts, understanding, interest, etc. Students mentioned that through collaboration and communication with team members, they were able to overcome initial difficulties and complete improved outputs, experiencing a different learning environment and a sense of achievement compared to traditional learning methods.

IV. CONCLUSION

In this study, we aimed to propose specific instructional model for operating computing liberal arts courses tailored for university non-majors to enhance learning motivation and creative problem-solving skills. Drawing from research in computing education and inspired by Papert's constructionist learning principles, we addressed the shortcomings typically found in computing courses designed for non-majors. Through expert review and application in educational settings, we validated the applicability of this course model and derived implications.

The research findings confirm that the MCPS instructional model in computing liberal arts classes for non-majors enhances creative problem-solving skills by fostering learning motivation. Additionally, learning motivation serves as a crucial factor in creative problem-solving abilities, acting as both the driving force and the sustainer of perseverance throughout the problem-solving process. Intrinsic motivation, which involves enjoying the creative problem-solving process and considering it rewarding, is an essential factor for creative problem-solving abilities. Cognitive self-efficacy has also been shown to influence creative thinking by encouraging individuals to tackle challenges with enthusiasm. These results can be seen in the same context as the results of existing research conducted on prospective teachers [62].

The MCPS instructional model proposed in this study could be utilized for systematic computing education in non-major computing liberal arts courses. Observing that learners exhibit motivation and confidence only when they find learning easy and enjoyable, adapting the instructional model to suit learners' needs by simplifying or appropriately modifying it could lead to more effective computing education. The more professors dedicate themselves to refining their teaching methods, the more likely it is for learners to experience successful teaching. Therefore, this study is expected to serve as foundational material for university computing liberal arts education, as the demand for computing education continues to rise.

REFERENCES

[1] Joint Ministries of South Korea, "Comprehensive Plan for Digital Talent Development," Aug. 22, 2022. [Online]. Available: https://www.moe.go.kr/boardCnts/viewRenew.do?boardID=72769&boardSeq=92573&lev=0&searchType=null&statusYN=W&page=1&s=moe&m=0315&opType=N.

[2] Ministry of Science and ICT press release, "The Ministry of Science and ICT. Digital 1 million talent training starts in earnest," Jan. 19, 2023. [Online]. Available: https://www.msit.go.kr/bbs/view.do?sCode=user&nttSeqNo=3182645&pageIndex=1&searchTxt=%EC%86%8C%ED%94%84%ED%8A%B8%EC%9B%A8%EC%96%B4&searchOpt=ALL&bbsSeqNo=94&mId=113&mPid=238.

[3] Ministry of Science and ICT, "Announcing the 2022 Software Centers of Excellence", No. 2022-0092, Jan. 27, 2022. [Online]. Available: https://www.msit.go.kr/bbs/view.do?sCode=user&mId=129&mPid=128&bbsSeqNo=100&nttSeqNo=3177544.

[4] S. Y. Hong et al., "Exploratory study on the model of the software educational effectiveness for non-major undergraduate students," *Journal of The Korean Association of Information Education,* vol. 23, no. 5, pp. 427–440, 2019. DOI: 10.14352/jkaie.2019.23.5.427.

[5] J. E. Nah, "Software Education Needs Analysis in Liberal Arts," *Korean Journal of General Education*, vol. 11, no. 3, pp. 63–89, 2017.

[6] J. Y. Seo, "A case study on programming learning for non-majors to nurture SW convergence talents," *Journal of Digital Convergence*, vol. 15, no. 7, pp. 123–132, 2017. doi: 10.14400/JDC.2017.15.7.123.

[7] M. J. Lee, "Exploring the Effect of SW Programming Curriculum and Content Development Model for Non-majors College Students: Focusing on Visual Representation of SW Solutions," *Journal of Digital Contents Society*, vol. 18, no. 7, pp. 1313–1321, 2017. doi:10.9728/dcs.2017.18.7.1313.

[8] H. J. Choi, "The Programming Education Framework for Programming Course in University," *The Journal of Korean Association of Computer Education,* vol. 14, no. 1, pp. 69–79, 2011.

[9] J. C. Shin, "The Effects of Computational Thinking-based Liberal Education on Problem Solving Ability," *Journal of the Korea Institute of Information and Communication Engineering*, vol. 14, no. 4, pp. 83–95, 2020.

[10] H. S. Kim, "Analysis of Non-Computer Majors' Difficulties in Computational Thinking Education," *The Journal of Korean Association of Computer Education,* vol. 18, no. 3, pp. 49–57, 2015.

[11] G. J. Park and Y. J. Choi, "Exploratory Study on the Direction of Software Education for the Non-major Undergraduate Students," *The Journal of Education & Culture,* vol. 24, no. 4, pp. 273–292, 2018.

[12] D. I. Park, "Relationship between learning motivation, metacognition, and creative problem-solving skills in social studies classes in elementary school based on the Creative Problem Solving model," M.S. thesis, Dept. Educational Technology, Ewha Womens Univ., Seoul, Korea, 2007.

[13] H. W. Kim, "A Study on the Design and Operation of Liberal Arts College for Improving Creativity of University Students," *The Journal of Creativity Education*, vol. 18, no. 4, pp. 91–14, 2018. doi:10.36358/JCE.2018.18.4.91.

[14] H. J. Park, "The Development and Application of a MCPS (Motivation and Creative Problem Solving) Instructional Model for Computing Liberal Arts for Non-Majors," Ph.D. dissertation. Dept. of Creativity Software, Andong National Univ., Andong, Korea, 2023.

[15] ACM and IEEE-CS, "Computing Curricula 2020," A Computing Curricula Series Report, Dec, 31, 2020.

[16] S. H. Kim, "A Survey on the Needs of Non-Major University Students for Coding Education Programs Status," Master thesis. Sungsil Univ., Counseling Education Psychology, Seoul, Korea, 2021.

[17] E. S. Kang, "Structural Software Education Model for Non-majors - Focused on Python," *Digital Contents Society*. vol. 20, no. 12, pp. 2423–2432, 2019. doi: /10.9728/dcs.2019.20.12.2423.

[18] M. S. Lim, "A Study on the Design of Creative Coding Educational Platform. Digital Contents Society," *Journal of Digital Contents Society*, vol. 21, no. 2, pp. 439–444, 2020. doi:10.9728/dcs.2020.21.2.439.

[19] M. J. Oh, "Analysis of Effects of Scratch Programing Education to Improve Computational Thinking," *Korea Association for Educational Information and Media*. vol. 24, no. 2, pp. 255–275, 2018.

[20] Y. J. Kim, "The Development and Application of Computational Thinking based Test Driven Problem Solving(CT-TDPS) Learning Model for Problem Solving Programming Education," Ph.D. Dissertation. Korea National Univ., Cheongju, Korea, 2021.

[21] W. S. Sohn, "A Developing a Teaching-Learning Model of Software Education for Non-major Undergraduate Students," *Korean Institute for Practical Engineering Education,* vol. 9, no. 2, pp. 107–117, 2017. doi: 10.14702/JPEE.2017.107.

[22] H. J. Jeon, "Development and Application of SW·AI Teaching·Learning Model SAGE to Improve the Creative Convergence Competency of Preservice Teachers," Ph.D. Dissertation. Computer Korea National Univ., Cheongju, Korea, 2022.

[23] S. K. Choi, "The Development of Multiple Project Based Coding Education Model focused on Computational thinking," Ph.D. Dissertation. Kyung Hee Univ., Seoul, Korea, 2019.

[24] H. R. Kim, "A Study on the meaning of Social Constructionist Approaches to Coding Education," *Journal of the Korean Association of Information Education,* vol. 25, no. 1, pp. 217–226, 2021.

[25] I. A. Kang, "Why constructivism?" Munumsa, Yongin, Gyeonggi-do, Korea: 1997.

[26] J. G. Sung, "An Exploratory Study of the Utilization of Virtual Reality as the Expansion of Maker Education Space," Ph.D. Dissertation, Dept. of Education, Seoul National Univ., Seoul, Korea, 2017.

[27] S. Papert, "Mindstorms: Children, computers, and powerful ideas," Basic Books, Inc., 1980.

[28] E. Ackermann, "Piaget's Constructivism, Papert's Constructionism: What's the difference?" *Future of learning group publication,* vol. 5, no. 3, pp. 438, 2001.

[29] S. Papert and I. Harel, "Situating constructionism," *Constructionism,* vol. 36, no. 2, pp. 1–11, 1991.

[30] A. Alanazi, "A Critical Review of Constructivist Theory and the Emergence of Constructionism" *American Research Journal of Humanities and Social Sciences,* vol. 2, pp. 1-8, 2016. doi:10.21694/2378-7031.16018.

[31] S. Papert, "Constructionism: A new opportunity for elementary science education. Massachusetts Institute of Technology," Media Laboratory, Epistemology and Learning Group, 1986.

[32] H. J. Choi and Y. J. Jeon, "Informatics Education, 3rd edition," Hanbit Academy, Seoul, Korea: 2023. ISBN: 9791156646471

[33] S. Gary, "An Investigation of Constructionism in the Maine Youth Center," Ph.D. dissertation, The University of Melbourne, Australia, 2006.

[34] Y. J. Jeon, "The Development and Application of a CT-CPS (Computational Thinking-based Creative Problem Solving) Instructional Model for the Software Education of New Curriculum," Ph.D. dissertation, Dept. of Gifted Information Education, Korea National Univ., Cheongju, Korea, 2017.

[35] H. W. Jung, "A study on basic software education applying a step-by-step blinded programming practice," *The Society of Digital Policy & Management*, vol. 17, no. 3, pp. 25–33, 2019. doi:10.14400/JDC.2019.17.3.025.

[36] S. I. Park et al., "Understanding the pedagogy of instructional methods," Company *of Educational Science*, Paju, Gyeonggi-do, Korea: 2011.

[37] I. A. Kang, H. J. Yoon and J. W. Hwang, "Maker Education: Constructivism Reunited in the Age of the 4th Industrial Revolution," Naeha Publishing House, Seoul, Korea: 2017.

[38] S. J. Jun, "Design and Effect of Development-Oriented Model for Developing Computing Thinking in SW Education," *Journal of the Korean Association of Information Education*, vol. 21, no. 6, pp. 619–627, 2017.

[39] K. S. Oh and S. J. Ahn, "A study on the development of educational contents about computational thinking," *The Journal of Korean Association of Computer Education,* vol. 19, no. 2, pp. 11–20, 2016.

[40] C. Lawshe, "A quantitative approach to content validity," *Personnel Psychology*, vol. 28, no. 4, 563–575, 1975.

[41] M. H. Shin, "A Study on the Development and Application of Rubrics for Performance Assessment in Terms of Promoting Program Learning Outcomes," *Journal of Engineering Education Research*, vol. 15, no. 5, pp. 108–118, 2012.

[42] K. Brennan & M. Resnick, "New framework for studying and assessing the Development of Computational Thinking," *Paper presented at annual American Educational Research Association meeting*, Vancouver, BC, Canada, 2012.

[43] CollegeBoard, "AP Computer Science Principles: Course and exam description," Feb, 10, 2023.

[44] M. J. Kim, G. Yoo, H. Kim: "Development of a scoring rubric based on Computational Thinking for evaluating students' computational artifacts in a programming course," *The Journal of Korean Association of Computer Education*, vol. 20, no. 2, pp. 1–11, 2017.

[45] S. H. Kim, "The development of an assessment Rubric of App-inventor program based on process-focused assessments," M.S. thesis, Dept. Computer Education, Korea Univ., Seoul, Korea, 2019.

[46] D. L. Gang, "Effects of process-oriented and result-oriented learning motivation methods on learning motivation and academic achievement," Master's thesis, Dept. of Curriculum Major, Korea National Univ., Cheongju, Korea, 1996.

[47] Pintrich, P. R., & De Groot, E. V, "Motivational and self-regulated leaning components of classroom academic performance." *Journal of Educational Psychology*. Vol. 82, pp. 33- 40. 1990.

[48] S. H. Cho, "Research on the development of a simple creative problem-solving test (I), "Korea Educational Development Institute, No. CR2001-33, ISBN: 8983884843

[49] S. J. Hwang, T. H. Jung & H. J. Lim, "Effect of Programming Education using App Inventor on Informatics Gifted Elementary Students' Creative Problem Solving Ability and Learning Flow," Master's thesis, Dept. of Gifted Information Education, Korea National Univ., Cheongju, Korea, 2015.

[50] H. J. Park, H. Kim, J. Choi, Y. Jeon, "Development of Teaching Efficacy Instrument in Informatics (Software and AI) Subject," *The Journal of Korean Association of Computer Education*, vol. 24, no. 4, pp. 39–52, 2021.

[51] H. J. Park and Y. J. Jeon, "A Design and Application of Software Liberal Arts Course based on CT-CPS Model for Developing Creative Problem-Solving Ability and Learning Motivation of Non-software Majors," IJIV., Vol. 6, No. 2, 2022. doi: 10.30630/joiv.6.2.996.

[52] E. A. Locke, E. Frederick and E., Lee, C., & P. Bobko, "Effect of self-efficacy, goals, and task strategies on task performance," *Journal of Applied Psychology,* vol. 69, no. 2, pp. 241, 1984.

[53] S. W. Kim and Y. J. Lee, "Effects of Software Education Using Robots in the Creative Problem-Solving Ability of Middle School Students," *The Journal of Korean Association of Computer Education,* vol. 23, no. 5, pp. 13–22, 2020, doi:10.32431/kace.2020.23.5.002.

[54] K. R. Park and M. R. Park, "Development and Application of Artificial Intelligence Job Creation Career Education Program focused on Constructivist Learning Environments for Elementary School Students.", *The Journal of Korean Association of Computer Education*, vol. 26, no. 5, pp. 13–30. 2023. doi:10.32431/kace.2023.26.5.002.

[55] M. W. Lee and S. S. Kim, "The Effect of Maker Education Program Utilizing Virtual Reality Creation Platform on Creative Problem Solving Ability and Learning Flow," *The Journal of Korean Association of Computer Education,* vol. 23, No. 5, pp. 65–72. 2020 doi:10.32431/kace.2020.23.2.007.

[56] W. Y. Chang, The Effects of Online Judge System on Motivation and Thinking in Programming Education: Structural Relationships between Factors," Vol. 24, No. 5, pp. 1–16, doi:10.32431/kace.2021.24.5.001.

[57] G. S. Noh, "Statistical analysis SPSS & AMOS for writing papers with proper knowledge," Seoul, Korea: Hanbit Academy, 2019.

[58] B. R. Bae, "Theory and practice of controlling and mediating effect analysis using SPSS/PROCES," Seoul, Korea: Cheongnam Book Publishing, 2021.

[59] M. J. Kim and J. M. Kim, "An Analysis of the Current Status of SW-Centered Universities' Informatics Curriculum in Korea based on Japan's Standards for Liberal Arts Informatics Curriculum," *The ,* Vol. 23, No. 2, pp. 65–72. 2020, doi:10.32431/kace.2020.23.2.007.

[60] Y. M. Go and H. S. Kim, "Analysis of the Effectiveness of an Artificial Intelligence Literacy Education Program for High School Students Based on NDIS Model," *The Journal of Korean Association of Computer Education,* Vol. 26, No. 3, pp. 57–66. 2023.

[61] J. Y. Lee, "Python big data analysis based on data science," Seoul, Korea: Hanbit Academy, 2020.

[62] Y. J. Jeon and T. Y. Kim, "Suggestions of Instructional Strategy in the Affective Aspect through the Analysis of Causality between the Computer Learning Attitude Factors of the Non-Major Students in the Software Education Class of the Teacher Training College," Journal of Korean Association of Computer Education, vol. 19, no. 6, pp. 15–23, 2016.