J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

The Study on Correlation of Cognition on Software Education with Improvement of Computational Thinking☆

Oakyoung Han[1] Jaehyoun Kim[2*]

ABSTRACT

The interest in the Fourth Industrial Revolution along with the development of ICT makes the software get the attention of the world. This phenomenon naturally leads to the concern for software education. Learning software coding is not easy for students whose major is in humanities or social sciences. This paper is a study of how cognition on software education affects to education of computational thinking. For research method, moderator variables were adopted on the proposed research model to prove that positive cognition can derive good influence on improvement of computational thinking. To find out moderator variables of the research model, we have conducted the questionnaire over three years for total of 928 students who took the software coding courses. As the result of the study, we proved that the positive cognition on software education can make the better improvement of computational thinking within proper moderator variables.

☞ keyword : Computational Thinking, Software Coding, Problem Solving, Cognition on Software, Multiple Regression Analysis, Moderator Model Analysis

1. Introduction

As entering the era of the Fourth Industrial Revolution, software education has taken an important position. Students are required to understand software coding to prepare for the future, and they have to improve the level of computational thinking abilities through software education to be a member of software society. Computational thinking is considered essential as a basic skill taught in schools along with 3Rs which are reading, writing, and arithmetic. Therefore, it is obvious that improving the level of computational thinking abilities is important.

Many universities in Korea offer software coding classes that include computational thinking in curriculums. Some of them have been designated as mandatory courses for every student. However, it is not easy to take coding classes for students majored in humanities or social sciences. Since the students are having difficulties in understanding the course materials, they think negatively on either software or computational thinking. Such dissatisfaction on courses lowers the effectiveness of the education as well as the achievement of the courses[1]. Negative attitude toward software courses affects the students to establish negative perceptions for software. It is very important for students who are future workers to have positive perceptions about the software because the bad perceptions can make a bad impact when working in the future.

We need an study to let students have positive cognition on software education. If students have the positive cognition on software education, it would improve the students' level of computational thinking abilities and it can make students have open minds toward software. In the near future, when the students work as a member of software centered society, open minds toward software and effect of the education can make students have a confidence on dealing with software.

In this study, three factors for positive cognition of

1 Sungkyun Software Education Institute, Sungkyunkwan University, Seoul, 03063, Korea
2 Dept. of Computer Education, Sungkyunkwan University, Seoul, 03063, Korea
* Corresponding author (jaekim@skku.edu)
[Received 31 December 2018, Reviewed 23 January 2019(R2 9 May 2019), Accepted 31 May 2019]
☆ This research was supported by a research grant from MSIP & IITP for software-oriented university (2015-0-00914)
☆ Apreliminary version of this paper was presented at APIC-IST 2018.

software education were used as independent variables for regression analysis, and the improvement of computational thinking was used as a dependent variable for the research model. The three independent variables for positive cognition on software education were 'SWImprvMe', 'NeedSW', and 'SWMkBetter'. After confirming that dependent variable was dependent on independent variables, we confirmed that moderator variables have a moderating effect on the relationship between independent variables and a dependent variable. The survey was conducted for three years to find out the moderator variables. We collected the questionnaire answer from all 928 students who took the software coding classes. As a result of the questionnaire, three moderator variables could be applied to the research model. The three moderator variables were 'SmllGrp', 'LabRelated', and 'MajorRel'.

2. Research Background

2.1 Improvement of CT

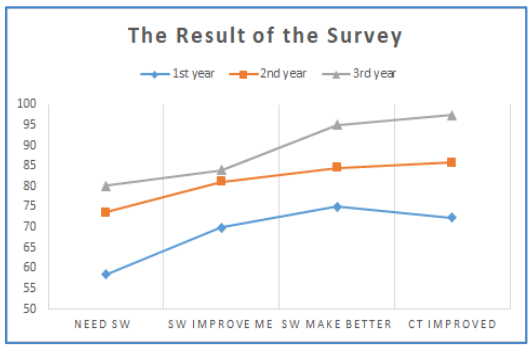
A survey was conducted for three years and the 928 students who took software coding classes were participated for the questionnaire. The result of questionnaire showed that computational thinking(CT) scores were higher when they had a better positive cognition on software.

(Table 1) The Result of the Survey

	1st Year	2nd Year	3rd Year
Need SW	180(57.9%)	162(73.7%)	317(79.8%)
SW Improve Me	211(69.8%)	178(80.9%)	333(83.9%)
SW Make Better	233(74.9%)	187(85%)	377(95.0%)
CT Improved	224(72.0%)	189(85.9%)	385(97.0%)
# of respondents	311	220	397

The composition of the survey participants was as shown in Table 1. 311 students were participated at the first year, 220 students at the second year, and 397 students at the third year. Hence, the total participants were 928 students.

Figure 1 indicates that not only the cognition on software was improved every year, but the scores of the computational thinking were also getting higher.



(Figure 1) Better Cognition, better CT

The first question related positive cognition on software education was 'Do you need software education even though you are majored in humanities or social sciences?' At the first year, only 180 students out of 311 that is 57.9% of the respondents answered positively. The second year was improved up to 69.8% of the respondents as positive answer, and the third year hit 79.8% as positive response.

The second question was 'Do we think learning software that enhances the thinking abilities can improve yourself?' In this question, the mentioned thinking abilities are meant to be comprehensive and generally refer to the computational thinking that is currently receiving intensive attention. At the first year, 211 students that is 69.8% showed positive response. At the second year, 80.9% of the respondents answered positively; and 83.9% of the respondents answered positively at the third year.

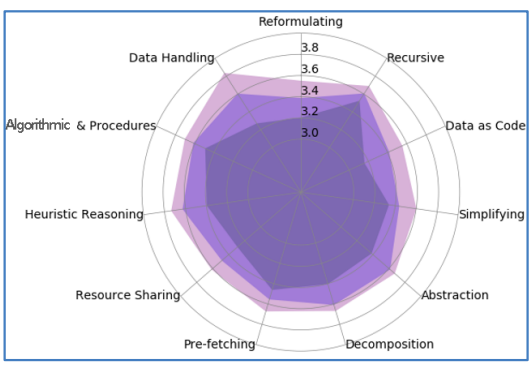
The third question was 'Do you think learning software can make better software centered society?' At the first year, 233 students that is 74.9% showed positive response. At the second year, 85% of the respondents answered positively; and 95.0% of the respondents answered positively at the third year.

The last question was 'Are you certain that your level of computational thinking abilities is improved in many aspects?' At the first year, 224 students that is 72.0% showed positive response. At the second year, 85.9% of the respondents answered positively; and 97.0% of the respondents answered positively at the third year.

2.2 Assessment for Computational Thinking

In Section 2.1, the question of whether the level of computational thinking abilities improved is needed to be verified. Hence, there were the questions to test computational thinking abilities in the survey. The questionnaire items about computational thinking were composed with 11 different areas[2,3,4,5,6,7,8,9].

Figure 2 shows the result of the radiation graph for computational thinking abilities. In Figure 2, the innermost radiograph shows the score of computational thinking abilities before taking no software coding class. The middle radiation graph in Figure 2 corresponds to the scores of computational thinking abilities of students who took one basic software coding class. The outermost radiation graph corresponds to the scores of computational thinking abilities of students who had taken two mandatory software coding classes.



(Figure 2) Improvement of Level of CT

2.3 Descriptive Questionnaire

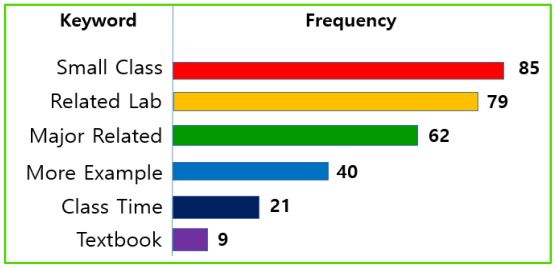
The 397 students who participated the third year of questionnaire survey were conducted a descriptive questionnaire on what they wanted for software coding classes to improve their computational thinking abilities. We analyzed the students' answers and found the main keyword.

The most highly mentioned keyword was 'Small Class.' The students wanted a small group class instead of a large-scale class. If there are too many students, students had difficult time to pay attention on the class.

The second most mentioned keyword was 'Related Lab.' It seemed that students had hard times if the theoretical class and the practical lab were not related. Even though some theoretical classes were not available to design a proper related practical lab, students wanted to see the connection between the theoretical class and the practical lab.

The third frequently mentioned keyword was 'Major Related.' Instead of learning the famous problem, students wanted problems which were related to their major for problem-based learning. Since the students took a class because it is mandatory for the graduation, they wanted useful problems for lab assignment rather than common problems.

Other keyword were 'More Example,' 'Class Time,' and 'Textbook.' We only selected the top 3 from the keyword to apply as the moderator variables to improve the computational thinking abilities.



(Figure 3) Keyboards for Better Class

3. Literature Review

3.1 Software Literacy

A researcher studied on providing a smart learning environment to improve the efficiency of education[10]. To prepare for the education in the era of the Fourth Industrial Revolution, applying high technology in the educational field is efficient than the traditional learning process to improve learning effect. To get advantage of using high technology in the classroom, students need to understand the concept of software because many class materials supplied as software products. For better education and learning, it is obvious that everyone involved in the education should be aware of software literacy.

There is a growing need for software education such as computational thinking education for convergence with other scholarships in universities. It is due to the advent of the 4th Industrial Revolution[11]. Software literacy is an imperative to understanding everyday 21st century contexts where the focus is on software. Up to high school, students can only be a consumer of software, but when students enter an university, students must be a consumer and also a producer. To produce a software, students must learn the basic concepts of software and practice software coding.

3.2 Computational Thinking

Information technologies are the base of the world infrastructure. Nowadays, the effort has been oriented mainly to convert students into users of computer tools. This has gone from being necessary to being insufficient, because the use of software applications means to manage a digital language that is obsolete in a time that is not proportional, in effort, to the time that has been invested in acquiring these skills. That is, instead of teaching students only the syntax of a changing language, they should be instructed in the rules that allow them to know how the digital language is constructed. Thus, computational thinking emerges as a paradigm of work, and the programming is established as the tool to solve problems[12].

Improving the computational thinking abilities is essential for everyone not just for understanding digital language. Computational thinking will be a fundamental skill, which can be used by everyone in future to strengthen and enhance learning[13]. It is a thought process that involves formulating problems so that solutions can be represented as computational steps and algorithms. In other words, computational thinking is problem solving ability, not simply software understanding.

3.3 Cognitive-Based Education

Existing programming education often leads to learner's cognitive burden. The traditional programming education is based on the programming method of teaching or project-based method in which students are required to find solutions depending on their ways of thinking and some guidance by teachers. The programming ability can be improved by various factors. The emotional intelligence is a concept that can complement existing cognitive-based education[14]. When a learner participates a programming class with positive cognitive learning, the programming ability can be empowered.

The primary missions of educational institutions, from elementary to graduate and professional schools. are to impart knowledge and to teach cognitive skills. One of the most important cognitive skills is no doubt problem-solving ability. Problem solving is of course predominantly involved in basic courses such as mathematics and science and in professional areas such as medicine. engineering, and architecture. But problem solving pervades almost all areas of instruction; reading and writing have important problem-solving components, for example. Even such a rudimentary process as retrieving information stored in long-term memory can be viewed as a problem-solving activity[15]. Hence, learning for problem solving is bond with cognitive skills.

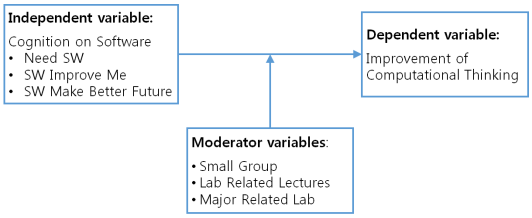
4. Method

4.1 Research Model

After analyzing the three-year survey, how students think about learning software affects improvement of computational thinking. The results of the analysis proved when the students had a positive cognition on purpose for the software coding class, the level of computational thinking abilities is improved more than others who had a negative cognition. Therefore, we adopted the improvement of computational thinking as a dependent variable, and applied factors of cognition on software education as independent variables. Figure 4 presents our research model, and it has independent variables part, dependent variable, along with moderator variables.

The independent variables for cognition on software education are 'NeedSW,' 'SWImprvMe,' and 'SWMkBetter' those are explained as the research background in Section 2.1.

For statistical analysis, IBM SPSS package is used. Through the statistical analysis, we verified that the proposed research model was correctly suggested.



(Figure 4) Moderator Model

4.2 Multiple Regression Analysis

The multiple regression is applied to check the relationship between several independent variables and a dependent variable. Three independent variables and one dependent variable were applied for the multiple regression analysis for the research model as Figure 5.

Variables Entered/Removed[a]

Model	Variables Entered	Variables Removed	Method
1	SWImprvMe, NeedSW, SWMkBetter[b]	.	Enter

a. Dependent Variable: CTImprv
b. All requested variables entered.

(Figure 5) Multiple Regression

As Figure 5, all requested independent variables are entered for the dependent variable 'CTImprv.'

4.3 Moderator Model Analysis

In statistics and regression analysis, moderation occurs when the relationship between two variables depends on a third variable. The third variable is referred to as the moderator variable. For the suggested research model, we applied 3 variables as moderator variables as shown in Figure 6.

Variables Entered/Removed[a]

Model	Variables Entered	Variables Removed	Method
1	SWImprvMe, NeedSW, SWMkBetter[b]	.	Enter
2	SmllGrp, LabRelated, MajorRel[b]	.	Enter

a. Dependent Variable: CTImprv
b. All requested variables entered.

(Figure 6) Moderator Variables

The rationale for the selection of moderator variables is described in Section 2.3. The moderator variables are 'SmllGrp,' 'LabRelated,' and 'MajorRel.'

5. Result

The results of the statistical analysis using SPSS are as follows.

Model Summary[b]

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Durbin-Watson
1	.455[a]	.207	.201	.55560	2.040

a. Predictors: (Constant), SWImprvMe, NeedSW, SWMkBetter
b. Dependent Variable: CTImprv

(Figure 7) Model Summary for R^2

The correlation between the independent variable and the dependent variable is highly correlated as 0.455 as Figure 7. The value of adjusted R square is 20.7%, and it shows the total explanatory power of dependent variable of independent variables. The value of Durbin-Watson is 2.040. Since the numerical value of Durbin-Watson is close to 2 and not close to 0 or 4, there is no correlation between the residuals; hence, the regression model is interpreted as appropriate.

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	31.607	3	10.536	34.131	.000[b]
	Residual	121.314	393	.309		
	Total	152.922	396			

a. Dependent Variable: CTImprv
b. Predictors: (Constant), SWImprvMe, NeedSW, SWMkBetter

(Figure 8) F-test for Research Model

The linear regression's F-test has the null hypothesis that there is no linear relationship among the variables. With F = 34.131 and 396 degrees of freedom the test is highly significant, thus we can assume that there is a linear relationship between the variables in our model as shown in Figure 8.

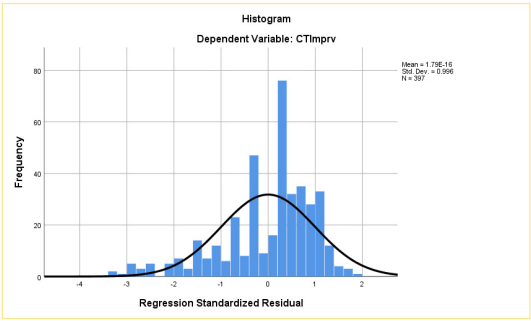
In Figure 9, the linear regression's t-value for the influencing relationship of 'NeedSW' toward 'CTImprv' is 2.943 which satisfies the condition over ±1.96, and significance probability is .003 that satisfies p < 0.05. Therefore, 'CTImprv' is dependent on 'NeedSW' in our model. The t-value of 'SWMkBetter' toward 'CTImprv' is 2.449 which satisfies the condition over ±1.96, and significance probability is .015. Therefore, 'CTImprv' is also dependent on 'SWMkBetter' in the research model.

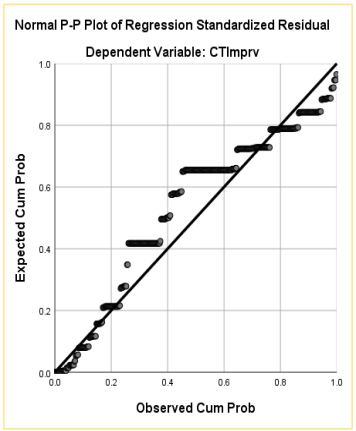
Coefficients[a]

Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
1	(Constant)	3.433	.126		27.218	.000
	NeedSW	.116	.039	.184	2.943	.003
	SWMkBetter	.110	.045	.172	2.449	.015
	SWImprvMe	.111	.048	.159	2.319	.021

a. Dependent Variable: CTImprv

(Figure 9) t-value for Research Model





(Figure 10) Histogram & Plot

Model Summary[c]

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics: R Square Change	F Change	df1	df2	Sig. F Change
1	.455[a]	.207	.201	.55560	.207	34.131	3	393	.000
2	.890[b]	.793	.790	.28503	.586	367.765	3	390	.000

a. Predictors: (Constant), SWImprvMe, NeedSW, SWMkBetter
b. Predictors: (Constant), SWImprvMe, NeedSW, SWMkBetter, SmllGrp, LabRelated, MajorRel
c. Dependent Variable: CTImprv

(Figure 11) Moderator Model Analysis

The t-value of 'SWImprvMe' toward 'CTImprv' is 2.319 which satisfies the condition over ±1.96, and significance probability is .021 that satisfies the condition p < 0.05. Therefore, 'CTImprv' is also dependent on the last moderator variable 'SWImprvMe' in the proposed research model.

The last thing we need to check is the homoscedasticity and normality of residuals. The histogram indicates that the residuals approximate a normal distribution. The plot shows that in our linear regression analysis there is no tendency in the error terms as shown in Figure 10.

The result of moderator model analysis is shown in Figure 11. Since the significance F change is .000, which is less than 0.05, it can be judged that there is a moderating effect on our model. It shows that 'Smllgrp', 'LabRelated' and 'MajorRel' are moderator variables for 'CTImprv' which is the dependent variable on the research model.

6. Discussion

This study started with the hypothesis that the positive cognition on software education improves the effectiveness of getting computational thinking abilities. For three years, the questionnaires were conducted, and the total of 928 students who took software coding classes answered the questionnaires. The result of the questionnaires were analyzed to verify the hypothesis.

As the software coding classes are repeatedly opened for the mandatory courses, students' perceptions about the classes have changed positively and gradually agreed to the necessity of software education because of emphasis on the importance of software in various media. As the result, it derive the positive cognition on software education. Having positive attitude toward the class makes the students accomplish the better educational effect. The multiple

regression analysis was processed to prove the hypothesis, and it has proven that the improving computational thinking abilities is dependent on the positive cognition on software education.

Furthermore, to find out additional factors that can improve computational thinking abilities even more, we proposed a moderator model in the research model. In order to confirm the moderator variables, 397 students were asked to write a descriptive questionnaire. Then three moderator variables with the most frequent responses were selected. Thru statistical analysis, the selected 3 moderator variables were determined that they have moderating effects on the proposed research model.

This study concluded that positive cognition on software education can make the better improvement of computational thinking within proper moderator variables. Nevertheless, it should be noted that there are variables that can not be used in general situation, since this study is limited to specific circumstances of our institution.

Reference

[1] O. Han and J. Kim, “Examining the relationship between educational effectiveness and computational thinking in smart learning environment”, Journal of Internet Computing and Services(JICS), Vol.19, No.2, pp.57-67, 2018.
https://doi.org/10.7472/jksii.2018.19.2.57

[2] J. Wing, “Computational thinking”, Communications of the ACM, Vol.49, No. 3, pp.33-35, 2006.
https://doi.org/10.1145/1118178.1118215

[3] S. Jacob and M. Warschauer, “Computational thinking and literacy”, Journal of Computer Science Integration, Vol.1, No.1, 2018.
https://doi.org/10.26716/jcsi.2018.01.1.1

[4] A. Juškeviciene and V, Dagiene, “Computational Thinking Relationship with Digital Competence”, Informatics in Education, Vol.17, No.2, pp.265-284, 2018.
https://doi.org/10.15388/infedu.2018.14

[5] S. Grover and R. Rea, “Computational Thinking in K‐12 : A Review of the State of the Field”, Educational Researcher, Vol.42, No.1, pp.38-43, 2013.
https://doi.org/10.3102/0013189X12463051

[6] Google for Education. Exploring computational thinking. Retrieved from https://edu.google.com/resources/programs/exploring-computational-thinking/#!home

[7] D. Barr, J. Harrison, and L. Conery, “Computational Thinking: A Digital Age”, Learning and Leading with Technology, ISTE(International Society for Technology in Education), pp.20-23, 2011.
https://files.eric.ed.gov/fulltext/EJ918910.pdf

[8] D. Weintrop, E. Beheshti, M. Horn, K. Orton, K. Jona, L. Trouille, and U. Wilensky, “Defining computational thinking for mathematics and science classrooms”, Journal of Science Education and Technology, Vol.25, No.1, pp.127‐147, 2016.
https://doi.org/10.1007/s10956-015-9581-5

[9] K. Boom, M. Bower, A. Arguel, J. Siemon, and A. Scholkmann, “Relationship between computational thinking and a measure of intelligence as a general problem-solving ability”, ITiCSE 2018 Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, pp.206-211, 2018.
http://delivery.acm.org/10.1145/3200000/3197104/iticse18main-id73-p.pdf?ip=115.145.32.105&id=3197104&acc=ACTIVE%20SERVICE&key=0EC22F8658578FE1%2EB50D9BE1468BDDBD%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&__acm__=1557318565_92eb52c0b66db82eff07b70015b2bfad

[10] O. Han and J. Kim, “A Study on the Technology Utilization for Smart Education in the 4th Industrial Revolution Era”, Journal of Internet Computing and Services(JICS), Vol.19, No.4, pp.71-82, 2018.
https://doi.org/10.7472/jksii.2018.19.4.71

[11] W. S. Kim, “A Study on the Recognition of Freshman on Computational Thinking as Essential Course”, Culture and Convergence, Vol. 39, No. 6, pp.141-170, 2017.
http://www.dbpia.co.kr/Journal/PDFViewNew?id=NODE07294178&prevPathCode=

[12] F. J. García-Peñalvo, “Computational thinking”,

IEEE Revista Iberoamericana de Tecnologías del Aprendizaje (IEEE RITA), Vol. 13, No. 1, pp. 17-19, 2018.
http://dx.doi.org/10.1109/RITA.2018.2809939

[13] R. Mohanty and S. Bala Dasm, “A Proposed What-Why-How (WWH) Learning Model for Students and Strengthening Learning Skills Through Computational Thinking,” In Progress in Intelligent Computing Techniques: Theory, Practice, and Applications, pp. 135-141. Springer, Singapore, 2018. https://link.springer.com/chapter/10.1007/978-981-10-3376-6_15

[14] Y. Bae and W. Jun, “A Study on Design and Implementation of a Programming Teaching Model Using Emotional Intelligence”, Journal of Internet Computing and Services(JICS), Vol.19, No.6, pp.125-132, 2018.
https://doi.org/10.7472/jksii.2018.19.6.125

[15] N. Frederiksen, “Implications of cognitive theory for instruction in problem solving,” ETS Research Report Series, 1983(1), pp.363-407, 1983.
https://doi.org/10.1002/j.2330-8516.1983.tb00019.x

◐ 저 자 소 개 ◑

한 옥 영(Oakyoung Han)
1985년 The University of Kansas 대학교 Computer Science 학과 (B.S. 학사)
1989년 California 주립 대학교 대학원 Computer Science 학과 (M.S. 석사)
1999년 한국과학기술원(KAIST) 대학원 전산학과 (박사수료)
2012년 성균관대학교 대학원 컴퓨터교육학과 (박사)
1989년～1991년 삼성전자 컴퓨터부문 응용개발실
1991년～1999년 덕성여자대학교 전산과장, 연구교수
2000년～현재 성균관대학교 컴퓨터교육과 겸임교수
2015년～현재 성균관대학교 성균SW교육원
관심분야 : 소프트웨어교육, 컴퓨팅사고력, 자료구조, 알고리즘, 인공지능, SW+인문, 인터넷중독, etc.
E-mail : oakyoung@skku.edu

김 재 현(Jaehyoun Kim)
1988년 성균관대학교 수학과 졸업(학사)
1992년 웨스턴일리노이 주립대학교 대학원 전산학과 석사
2000년 일리노이공과대학교 대학원 전산학과 박사
2014년～현재 한국컴퓨터교육학회 부회장
2010년～현재 한국인터넷정보학회 부회장
2002년～현재 성균관대학교 컴퓨터교육과 교수
2016년～현재 성균관대학교 성균SW교육원 원장
2019년～현재 성균관대학교 사범대학장
관심분야 : 객체지향 소프트웨어공학, 컴퓨터교육, Computer Based LET etc.
E-mail : jaekim@skku.edu