I.J. Modern Education and Computer Science, **2017, 5, 1-8**
Published Online May 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2017.05.01

Factors Affecting Failing the Programming Skill Examination of Computing Students

Rex P. Bringula, Ariel D.V. Aviles, Ma. Ymelda C. Batalla, Ma. Teresa F. Borebor, Mark Anthony D. Uy, and Bernadette E. San Diego
College of Computer Studies and Systems, University of the East, Manila, 1008, Philippines
E-mail: rex_bringula@yahoo.com, co113@yahoo.com, may.cabrera02@gmail.com, aseret_f@yahoo.com, markanthony.uy@ue.edu.ph, ditas_sandiego@yahoo.com

***Abstract*—**This descriptive study utilized a validated instrument to determine the factors that affect failing a programming skill examination. Through this finding, the study attempted to provide solutions to address the concerns of the students. The top three reasons why students failed the programming skill examination were the insufficient time dedicated to programming courses, self-inefficacy in programming, and unmatched question-time allotment. Overall, respondents attributed their failed mark in programming skill examination to question-related factors. This was confirmed through the use of regression analysis. Hence, it was concluded that students failed the programming skill examination because the perceived ability of the students in the programming skill examination did not correspond with the degree of difficulty of the programming skill examination questions. Further, the null hypothesis stating that student-related concerns do not predict the number of times the programming skill examination would be taken was partially rejected. Hence, it was recommended that the programming skill examination questions be calibrated based on the ability of the students. Future research directions were also presented.

***Index Terms*—**Computing curriculum, difficulty in programming, programming, programming skill, tutoring.

I. Introduction

The demand of the industry, positive outlook towards the computing degree programs, and perceived job opportunities are some of the pulling factors that attract students to enroll in computing degree programs [4]. Programming is one of the core and desirable skills of computing students (e.g., Computer Science, Information Technology, and Information Systems) [3]. In fact, according to JobStreet.com [6], programmers topped the list of highest paid professionals for new graduates in the Philippines.

Unfortunately, learning to program is difficult [8,9,11]. Programming requires exceptional precision [7,9] since it involves correctness of logic, syntax, and semantics. Robins et al. [11] further commented that a strong foundation on the knowledge on computers, programming languages, programming tools and resources, and theory and formal methods are prerequisites to a skilled programmer. The difficulty of putting the solution to a language understood by the computer is a barrier for novice programmers. When confronted with programming errors, they feel anxious, panicky, and stressed [12]. They also experience grief and frustration [13]. Ultimately, they tend to shift to other degree programs when they cannot manage the difficulties [4,11,13].

It is worth mentioning that there were students who studied and worked hard to finish the degree. To test how much knowledge the students gained from their programming courses and to determine if they are ready to face the demand of the industry, a hands-on programming skill examination (PSE) may be given. It may be given at the later part of the course. The overall objective of this activity is to provide students who might need learning interventions. Unfortunately, it has been observed that students have difficulty in passing this activity despite the fact that the PSE is given at the end of the course. To understand this phenomenon and to propose solutions to the pressing concerns, this study was conceived. It aimed to answer the following research questions. 1) What are the reasons why students failed the PSE? 2) What are the possible solutions to address the reasons behind failure in the PSE?

The paper is subdivided into five main sections in order to answer the above questions. In Related Work section, studies conducted related to this research were presented. This section also served as basis in the formulation of the Research Framework of the study. The Methodology part presented how data were collected and analyzed. Afterwards, the findings of the study were presented in the Results Section. The findings were discussed in the Discussion section. The summary of the study was presented in the Conclusions and Recommendations.

II. Related Work

The following sections discussed systems theory which served as the theoretical basis of the current study. This section also presented similar studies about programming skills assessment and the challenges students faced when learning to program.

A. Systems Theory

Copyright © 2017 MECS

This study was guided by the systems theory or systems thinking. The systems theory is a "thought process that considers the interconnections and reactive relationships among the parts of a system" [15, p. 987]. It argues that parts of a complex system are interacting and that they are interdependent. If one of the parts changed, it may have a consequential impact on other parts. Hence, every part should be evaluated to determine the potential impacts at various levels [15].

This theory assumed that the organization would change based on the feedback provided by the stakeholders. In the context of educational setting, stakeholders may be teachers, administrators, and students. Feedbacks may be qualitative and quantitative in nature and they are deemed important for the improvements of an organization. Example, if students could not perform a long division, then there was flaw in the process of the organization. It may involve the curriculum, assessment procedures, teacher characteristics, and student characteristics. The feedback mechanism would identify the root causes of the problem and may serve as basis in changing the systems.

B. Programming Skill Assessment

Programming skill assessment is an evaluation of the cognitive skills of students in the area of programming. This skill may be evaluated through paper-based method (e.g., multiple choice exam) [7] or through hands-on exam. Kuechler and Simkin [7] said that multiple choice tests of programming skill assessment was more favorable because of the following: 1) advantage of machine scoring, volume grading, and easily computed statistical analyses of test results, 2) students were more confident since they could guess the correct answer, 3) students' dislike for tests where good writing skills compensate for the lack of factual recall, 4) higher chance of covering a wide range of course topics, 5) the perception that multiple choice tests are more objective, 6) compatibility with web-based courses, 7) easier resolution when test disputes arise due to higher referencing capabilities, and 8) more labor-intensive tasks inherent in grading constructed-response examinations.

On one hand, in 1988, Barger [2] tested the mastery of students on the BASIC programming language. The students had to solve a problem, encode the solution, debug the code, and save the file on a floppy disk. The teacher collected the disks and run the program. Whenever an error was encountered, the teacher inserted comments on how to correct the codes. The disks were then returned to the students to correct the mistakes. Another approach was proposed by Wang et al. [17] wherein students exchanged codes with other students and both acted as peer code reviewers. They shared ideas and made suggestions on how to improve the code. The teacher then gave scores to students based on their performance in writing, reviewing, and revising programs, and their abidance to the peer code review process.

C. Difficulty in Learning Programming

Novice programmers are confronted by different challenges in learning programming. These include weak problem solving strategies [5,11], insufficient programming knowledge [11], limited debugging skils [5], and incomplete understanding on how to transform human instructions to programming language syntaxes [19]. Abdel Rahman et al. [1] attributed the weakness of students in programming on lack of time to practice the programming language and on low skill of the English language.

The study of Suranauwarat [14] investigated the difficulties of non-Computer Science (CS) students who were taking up databases courses. The researcher disclosed that the non-CS students had difficulties in the hardware and software aspects of learning database management systems. The participants of the study were not skilled in using Microsoft Visio. This tool is used to draw entity relationship diagram (ERD). The author proposed that other tools such as ERDPlus, ER Assistant, and Gliffy be utilized instead of the Microsoft Visio because the first-three software tools are free and easy to use. The second problem was that students had difficulty in installing the relational database management systems (RDBMS). The researcher said that the time consumed in installing the RDBMS could have been used in learning the course. Lastly, there was a need for a new web development environment that is quick and easy to reproduce. The second and the last problems were addressed by shifting to cloud technologies, using of Vagrant, and no install version of MySQL.

The study of Mladenovic, Rosic and Mladenovic [8] compared the attitudes, motivation, and understanding of programming of Grade 7 elementary students (13-14 years old) on basic programming concepts. All participants of the study utilized Scratch and Logo but only differed in sequence. The first group utilized Logo followed by Scratch. The second group utilized Scratch followed by Logo. Both groups utilized both programming languages in a span of six weeks. The researchers revealed that students who used Scratch first had higher understanding on the complex concepts of programming structures than those who used Logo first. It was further disclosed that the former language was more positively accepted than the latter. Hence, students were more motivated to learn programming in the first programming language than the second programming language. The study concluded that it is advisable that students use Scratch programming language in order to engage students in programming.

Bringula et al. [3] studied the sources of programming errors of novice Java programmers (i.e., students). The authors found out that thought error was the main reason why novice programmers committed programming errors. This error is committed when there is a mismatched between correct coding syntax and acquired coding skills and can be "manifested by writing a syntax that they assumed to be a correct syntax but in reality it is syntactically incorrect" [3, p. 8]. It was also shown that novice Java programmers mostly committed errors that were related to symbols, keywords, and variable names. Since Java is a wordy programming language,

carelessness in typing the code was the strongest predictor of programming errors.

The difficulty of learning to program can be overcome by lecture attendance of students. Veerasamy et al. [16] analyzed the class attendance of students and its relationship with programming performance. Class attendance and in-class and homework formative tests were analyzed and correlated with programming performance. Programming performance was measured in terms of final examination. It was revealed that formative take home assessments had significant positive relationship with programming performance. On the other hand, negative correlation existed between lecture attendance and final exam scores.

In a similar study, Wilson and Shrock [18] determined the factors that contributed to the success of students to pass an introductory computer science course. They investigated twelve factors; namely, math background, luck, effort, difficulty of task, ability, self-efficacy, encouragement, comfort level in the course, work style preference, prior programming experience, prior non-programming computer experience, and gender. The study revealed that comfort level was the strongest predictor of success in the course. The level of mathematics skills of the students was the second most influential factor that affected success in the course. It is interesting to note that success in the course was also attributed to luck for success/failure. The researchers recommended that teachers should develop a classroom environment that is comfortable to students in order for them to pass their programming courses.

III. Research Framework, Operational Definition of Terms, and Hypothesis

Based on the foregoing discussion of theoretical bases, the research framework of this study was formulated. The research framework is shown in Figure 1. Based on this figure, the study considered the five organizational factors (hereafter referred as factors) which might influence the passing or failing of students in the PSE. The elements are defined below. Similarly, the factors were the independent variables of the study while number of times students took the PSE was the indicator of failure in the PSE – the dependent variable.

Question – It involves suitability of questions to student's level of programming skill, appropriateness of questions to its time allotment, and clarity of questions.

Faculty – It refers to the relevance of courses taught by the teachers as well as their dedication in teaching.

Curriculum – It includes items involving the appropriateness, organization, and sufficiency of courses in the curriculum.

Laboratory – It entails the adequacy of hardware and software of the college.

Self-efficacy – It determines the confidence towards programming and their level of preparedness in taking the PSE.

It is hypothesized that organizational factors, singly or in combination, do not predict failure in programming skill examination.

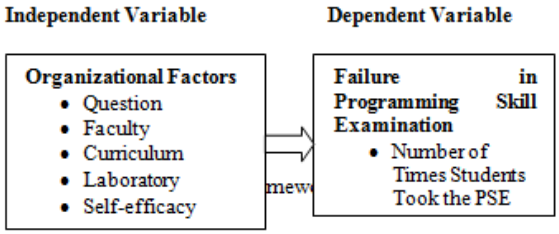

Fig.1. Research Framework of the Study

IV. Methodology

The study employed different processes in data collection. The first step undertaken was to identify the participants of the study. The research locale of the study was also identified at this section. The research design of the study was then discussed. The research instrument and how it was validated were also presented. Lastly, statistical tools utilized in the study were discussed.

A. Participants of the Study, Research Locale

This study was guided by the systems theory by soliciting feedback from the students. It was conducted during the first and second semesters of school year 2013-2014. There were 191 students who failed the PSE. There were 145 respondents who participated in the study which was about 76% of the total population. The respondents of the study were mainly composed of male students (f = 138, 74%), Information Technology students (f = 179, 96%), who took the PSE at least three times (f = 136, 73%). Majority of the participants developed a software on sale and inventory (f = 117, 81%).

The study was conducted at the College of Computer Studies and Systems of the University of the East. The college offers three degree programs; namely, Computer Science, Information Technology, and Information Systems. These degree programs offer capstone courses. At the end of these courses, students would take a programming skill examination (PSE). In the PSE, students were given an average of four-hour period to solve a programming problem. The problems that were given to the students were related to their developed software. This was conducted at the laboratories of the college after two weeks of oral capstone project defense.

B. Research Design, Research Instrument and its Validity and Reliability, Statistical Treatment of Data

This descriptive study utilized a validated instrument in the form of a questionnaire. The questionnaire had two parts. The first part gathered the reasons why the students failed the PSE. There were five factors considered. The questionnaire was pretested in one class section with 40 students. Respondents rated the items from 1 (Strongly Disagree) to 5 (Strongly Agree). Factor analysis and Cronbach's alpha analysis were utilized to determine the validity and reliability of the items. All factors with factor

Copyright © 2017 MECS

loadings of at least 0.50 and all items with Cronbach's alpha of at least 0.70 were retained. Since all factors and items met the criteria, the number of items was retained. The factors, number of items of each factor, factor loadings, and Cronbach's alpha are shown in Table 1.

The last part of the questionnaire was an open-ended question. This section allowed the participants to comment or to make suggestions on how to improve the process of PSE. Responses were coded and analyzed. The codes were presented to four faculty members of the Computing Department to determine their consensus on the coding. The codes were revised until an agreement was reached and established.

Table 1. Validity and Reliability of Factors that Might Explain Failure in the PSE

Student-Related Concerns	Factor loading
Question-related **(α = 0.826)**	
1. Questions are not suitable to students' level.	0.867
2. Questions are not appropriate to the allotted time.	0.835
3. Questions are difficult to understand.	0.895
Faculty-related **(α = 0.930)**	
4. CCSS faculty members did not teach students programming well.	0.936
5. CCSS faculty members did not teach students database well.	0.941
6. CCSS faculty members did not hone students' logic well.	0.909
7. CCSS faculty members gave students irrelevant programming problems which could not prepare them in PSE.	0.847
Curriculum-related **(α = 0.851)**	
8. The subjects in the curriculum are not enough to prepare the students in PSE.	0.876
9. The subjects in the curriculum are not well-organized to prepare the students in PSE.	0.891
10. The subjects in the curriculum are not appropriate to prepare students in PSE.	0.870
11. The hours dedicated to programming in the subjects are not enough.	0.711
Laboratory-related **(α = 0.880)**	
12. CCSS laboratories are not state-of-the-art to hone the programming skill of the students.	0.913
13. CCSS laboratories are not adequate to hone the programming skill of the students.	0.932
14. Programming languages at the CCSS laboratories and the languages used in the students' thesis are the not the same.	0.854
Self-efficacy-related **(α = 0.881)**	
15. I am not prepared for the PSE.	0.836
16. I do not have enough time to practice for the PSE.	0.662
17. I am not good in programming.	0.876
18. I am not good in logical thinking.	0.831
19. I do not have enough skill in database programming (e.g., database connection).	0.830
20. I do not have enough skill in programming.	0.744

Descriptive statistics such as frequency count, percentage, ranking, and mean were utilized to describe the data. Multiple regression analysis at 5% level of probability and 95% reliability was used to determine if organizational factors would significantly affect the failure of students in programming skill examination.

V. Results

Table 2 shows the reasons why students failed the programming skill examination. There were five factors investigated with 20 items. The item *"The hours dedicated to programming in the subjects are not enough"* got the highest mean rating of 3.37. This was followed by the item "I am not good in programming" with a mean rating of 3.18. The third highest mean rating was on the *"Questions are not appropriate to the allotted time to solve the problem"* (mean rating = 2.98). On the other hand, respondents had low mean rating on the items *"CCSS faculty members did not hone students' logic well"* (mean = 2.50), *"CCSS faculty members did not teach students database well"* (mean = 2.41), *and "Programming languages at the CCSS laboratories and the languages used in the students' theses are not the same"* (mean = 2.36).

If analyzed by factor, question-related factors (mean = 2.97) ranked first as the major reasons why students failed the PSE (See Table 3.). This was followed by the curriculum-related factors (mean = 2.94). The third reason was attributed to the respondents' programming self-efficacy factors (mean = 2.86). Faculty-related and laboratory-related factors ranked fourth and fifth, respectively.

Table 4 confirmed the results of Tables 2 and 3. Thirty-two percent (32%) of the respondents had concerns on the way the questions were written. Some of these concerns include *"Questions are unclear and tricky," "Make the questions easy,"* and *"Kindly, make the questions clear."* The manner of checking of the answers (f = 10, 16%), laboratory-related concerns (e.g., non-availability of specialized programming language) (f = 7, 11%), and time allotment to solve the problem (f = 6, 10%) were consistently found in the responses. Respondents asked for support (f = 8, 13%) from the college in terms of programming tutorials. Further, they requested diversity (f = 7, 11%) of PSE which caters to the other fields of computing. This may involve web design, photo manipulation, or networking.

Table 5 shows the regression of number of times the students took the PSE on reasons why students failed the PSE. The only significant factor that could predict the number of times the students took the PSE was the question-related factors (beta = 0.22, $p < 0.05$). The amount of variability in the number of times the students would take the PSE may be accounted to the difficulty of the questions given (Adj. R^2 = 2%).

Table 4 confirmed the results of Tables 2 and 3. Thirty-two percent (32%) of the respondents had concerns on the way the questions were written. Some of these concerns include *"Questions are unclear and tricky," "Make the questions easy,"* and *"Kindly, make the questions clear."* The manner of checking of the answers (f = 10, 16%), laboratory-related concerns (e.g., non-availability of specialized programming language) (f = 7, 11%), and time allotment to solve the problem (f = 6, 10%) were consistently found in the responses. Respondents asked for support (f = 8, 13%) from the college in terms of programming tutorials. Further, they requested diversity

(f = 7, 11%) of PSE which caters to the other fields of computing. This may involve web design, photo manipulation, or networking.

Table 5 shows the regression of number of times the students took the PSE on reasons why students failed the PSE. The only significant factor that could predict the number of times the students took the PSE was the question-related factors (beta = 0.22, $p < 0.05$). The amount of variability in the number of times the students would take the PSE may be accounted to the difficulty of the questions given (Adj. R^2 = 2%).

Table 2. Reasons Why Students Failed PSE

Student-Related Concerns	Mean	Rank
Question-related		
1. Questions are not suitable to students' level.	2.96	4
2. Questions are not appropriate to the allotted time.	**2.98**	**3**
3. Questions are difficult to understand.	2.99	5
Faculty-related		
4. CCSS faculty members did not teach students programming well.	2.51	17
5. CCSS faculty members did not teach students database well.	2.41	19
6. CCSS faculty members did not hone students' logic well.	2.50	18
7. CCSS faculty members gave students irrelevant programming problems which could not prepare them in PSE.	2.74	12
Curriculum-related		
8. The subjects in the curriculum are not enough to prepare the students in PSE.	2.92	6
9. The subjects in the curriculum are not well-organized to prepare the students in PSE.	2.77	10
10. The subjects in the curriculum are not appropriate to prepare students in PSE.	2.69	14
11. The hours dedicated to programming in the subjects are not enough.	**3.37**	**1**
Laboratory-related		
12. CCSS laboratories are not state-of-the-art to hone the programming skill of the students.	2.63	15
13. CCSS laboratories are not adequate to hone the programming skill of the students.	2.59	16
14. Programming languages at the CCSS laboratories and the languages used in the students' thesis are the not the same.	2.36	20
Self-efficacy-related		
15. I am not prepared for the PSE.	2.80	9
16. I do not have enough time to practice for the PSE.	2.72	13
17. I am not good in programming.	**3.18**	**2**
18. I am not good in logical thinking.	2.83	8
19. I do not have enough skill in database programming (e.g., database connection).	2.77	11
20. I do not have enough skill in programming.	2.86	7

Table 3. Reasons Why Students Failed PSE per Factor

Student-Related Concerns	Mean	Rank
Question-related	2.97	1
Curriculum-related	2.94	2
Self-efficacy-related	2.86	3
Faculty-related	2.54	4
Laboratory-related	2.53	5

Table 4. Results of Coded Responses

Concerns / Recommendations	*f* (n = 63)	%
Question-related	20	32
Manner of checking the PSE	10	16
Support	8	13
Diversity of PSE	7	11
Laboratory-related	7	11
Time allotment	6	10
PSE is OK.	5	8
Faculty-related	3	5
Abolish the PSE	2	3
Self-efficacy	2	3
Teaching	2	3
Grading the PSE	1	2
Curriculum-related	1	2

Table 5. Predictors of Failing the PSE

Student-Related Concerns	Beta	*p*-value
Question-related	**0.22**	**0.037**
Curriculum-related	0.03	0.777
Self-efficacy-related	0.18	0.095
Faculty-related	-0.14	0.244
Laboratory-related	0.09	0.370
Adjusted R^2 = 0.02		

VI. Discussion

This study attempted to determine the reasons why students failed a programming skill examination (PSE) – a final requirement of the capstone project course. It also determined the factors that could predict the number of times to take the PSE. The respondents disclosed that they failed the PSE due to limited programming hours dedicated in their curriculum. This attribution does not signify that the curriculum lacks the number of hours dedicated to programming. In fact, the curriculum of the college was based on the prescribed number of hours stipulated by the regulatory of higher education in the Philippines. This perceived inadequacy only shows that the prescribed number of hours may be revised. One possible solution is to revisit the curriculum and increase the number of units of programming courses.

Interestingly, respondents disclosed that they were not good in programming. They had low self-confidence in terms of programming. This finding is similar to the

study of Wilson and Shrock [18]. To compensate with their shortcomings in programming, they tended to shift their focus on other fields of computing. This was supported by the results shown in Table 4. They recommended other forms of skill evaluation, such as photo-manipulation, networking, web design, and documentation. On the other hand, this concern could have been avoided in the first place if students would develop program modules on their own while on their capstone project course. It must be noted that capstone courses require the students to develop their own software. If the tasks (i.e., development of different modules, database design, and coding) would be distributed among members, their programming skills would be polished and they would be more confident in taking the PSE.

This concern is a complex issue since it requires collaboration and participation of different stakeholders. First, the research adviser of the students must monitor the contributions of each group member. It must be emphasized that all members have to develop modules on their own. The role of the group leader is to synthesize and monitor the group members on each assigned task. Lastly, all members must participate actively specially during the integration phase.

The third reason for having a failed mark in PSE was the mismatched of level of difficulty of the question and the allotted time to solve it. The same result was found in the textual response. Respondents were consistent in their perceptions that the level of difficulty of the questions did not fit the designated time to solve them. The questions may not be problematic. Instead, their difficulty may not be suitable to the self-confidence and programming skills of the students. In other words, there is a gap between the level of difficulty of the questions and the level of programming skills of the students. In order to address this concern, it is recommended that the questions be calibrated based on the time allotment and programming skills of the students.

When the analysis was conducted per factor, it was revealed that question-related factor was the main reason why students failed the PSE. This supports the previous argument that there was a need to revisit the given programming problems. On the other hand, it is interesting to note that the students had low mean rating in terms of faculty- and laboratory-related factors. Though there were seven individuals who were not satisfied with the laboratories of the college, their perceptions were outweighed by the majority (f = 138) of the respondents. These findings were supported by the fact that they perceived that teachers taught them well in their programming courses and the software and hardware of the laboratories of the college were sufficient in conducting the PSE. These reflect the dedication of the faculty and IT infrastructure of the university.

The textual responses of the respondents also showed that they attributed their failing mark in the PSE to question-related factors (See Table 4.). It is worth noting that the way of checking the PSE was one of the reasons of a failed PSE. The 10 respondents argued that there should be a clear PSE rating system. They recommended that it should be stipulated in the policy the number of times the program should be checked, the assigned points per module, and a passing mark.

On the other hand, it was found that there were students asking for support in the form of tutorials. They perceived that they would be capable of passing the PSE provided that they were given a hands-on tutorial. This clarifies the previous finding in terms of confidence towards programming. They were not confident in programming but given an opportunity to undergo tutoring, their attitude might change and they would feel more prepared to take the PSE. Thus, a change in attitude and level of preparedness of the students may help them pass the PSE.

The regression analysis confirmed that question-related concerns could predict the failure of students in the PSE. The positive beta of this predictor indicates that it positively influences the number of times to take the PSE (i.e., an indication that students failed the PSE). In other words, as the question gets more difficult, it could be expected with certainty that the number of takes would also increase. This finding consistently revealed that failure (or passing) of the students depends on the level of difficulty of the questions.

Meanwhile, it was revealed that question-related factor was accounted to 2% in the variability in the number of times the PSE would be taken. Thus, there were other variables that were not included in the present study. It is recommended that level of preparedness and attitude towards programming be included in future studies.

VII. Conclusions and Recommendations

On the basis of the findings presented, the null hypothesis stating that student-related concerns do not predict the number of times the PSE would be taken was partially rejected. Specifically, the passing or failing of the students depends on the level of difficulty of the questions. It was found out that respondents consistently attribute their failure in this factor. Therefore, the gap between the difficulty of the problem and the programming skills of the students on hand was the reason why students failed the PSE.

Further, it can be concluded that the solutions to the concerns of the students warranted both short and long-term solutions. Short-term solutions include calibration of questions to meet the level of skill of the students and provision for tutoring. On one hand, long-term solutions consist of encouragement of students to develop their own modules during capstone courses. Furthermore, curricular revisions are encouraged to address the concerns of the students. Lastly, it is recommended that the proposed solutions and interventions be evaluated to determine their effectiveness.

Acknowledgements

The authors are indebted to Dr. Ester A. Garcia, Dr. Linda P. Santiago, Dr. Olivia C. Caoili, Dean Rodany A.

Merida, Dr. Socorro R. Villamejor, Dr.Mark Fabella, and to all participants of the study. This paper is funded by the University of the East.

REFERENCES

[1] S. M. AbdelRahman, B. AL-Syabi, E. Al Sharji, and S. S. Al Kaabi,"The Reasons behind the Weakness of some Students in Programming Courses in the College of Applied Science, Ibri", *International Journal of Modern Education and Computer Science(IJMECS)*, vol. 8, no.1, pp.48-54, 2016.DOI: 10.5815/ijmecs.2016.01.07

[2] R. N. Barger, "On-line evaluation and remediation of programming skills", in *Proceedings of the Annual Meeting of the International Association for Computing in Education*. (L.A., California, USA, Apr. 5-9, 1988). International Association for Computing in Education, New Orleans, USA, 1-8.

[3] R. P. Bringula, G. M. A. Manabat, M. A. A. Tolentino, and E. L. Torres, "Predictor of errors of novice Java programmers", *World Journal of Education*, vol. 2, no. 1, pp. 3-15, 2012. doi:10.5430/wje.v2n1p3.

[4] R. P. Bringula, R. Torres, and R. A. Merida, "Push and Pull of Institutional Image Indicators and Computing Degree Programs Viewed Through the Lens of Shifters and Transferees at the University of the East", *Management Education: An International Journal*, vol. 16, no. 3, pp. 13-27, 2016.

[5] A. Carbone, J. Hurst, I. Mitchell, and D. Gunstone, "An exploration of internal factors influencing student learning of programming", in Md. J. Nordin, K. Jumari, M. S. Zakaria, & Suwarno (Eds.). *Computing Education 2009: Proceedings 11th Australasian Computing Education Conference (ACE 2009), Conferences in Research and Practice in Information Technology (CRPIT), 95* (pp. 25–34). Sydney, Australia: Australian Computer Society, Inc.

[6] Jobstreet.com. (2015). *The 2015 jobs and salary report for fresh graduates*. Retrieved from http://www.jobstreet.com.ph/career-resources/2015-jobs-salary-report-fresh-graduates

[7] W. L. Kuechler, and M. G. Simkin,"How well do multiple choice tests evaluate student understanding in computer programming classes?", *Journal of Information Systems Education*, vol. 14, no. 4, pp. 389-399, 2003.

[8] M. Mladenović, M. Rosić, S. Mladenović,"Comparing Elementary Students' Programming Success based on Programming Environment", *International Journal of Modern Education and Computer Science(IJMECS)*, vol.8, no.8, pp.1-10, 2016.DOI: 10.5815/ijmecs.2016.08.01

[9] M. O. Pendergast, "Teaching introductory programming to IS students: Java problems and pitfalls", *Journal of Information Technology Education*, vol. 5, 491-595, 2006.

[10] D. N. Perkins, and F. Martin "Fragile knowledge and neglected strategies in novice programmers", In E. Soloway & S. Iyengar (Eds.), *Empirical studies of programmers, First Workshop* (pp. 213–229). Norwood, NJ: Ablex, 1986.

[11] A. Robins, J. Rountree, and N. Rountree, "Learning and teaching programming: A review and discussion", *Computer Science Education*, vol. 13, no. 2, pp. 137–172, 2003.

[12] C. Rogerson, and E. Scott, "The fear factor: How it affects students learning to program in a tertiary environment", *Journal of Information Technology Education*, vol. 9, pp. 147-171, 2010.

[13] S. Shuhidan, M. Hamilton, and D. D'souza, "A taxonomic study of novice programming summative assessment", in *Computing Education 2009: Proceedings 11th Australasian Computing Education Conference: Conferences in Research and Practice in Information Technology (CRPIT)* (Wellington, New Zealand, Jan. 20-23, 2009). ACE 2009. Australian Computer Society, Inc., Sydney, Australia, 147–156, 2009.

[14] S. Suranauwarat,"An Approach to Solving Technical Difficulties Facing Non-CS Students in a Database Class", *International Journal of Modern Education and Computer Science(IJMECS)*, vol. 9, no. 2, pp.14-26, 2017.DOI: 10.5815/ijmecs.2017.02.02

[15] B. Thornton, "Systems Theory/Thinking", in *Encyclopedia of Educational Leadership and Administration* (Fenwick W. English, ed., Vol. 2, pp. 987-988). Thousand Oaks, CA: SAGE Reference, 2006.

[16] A. K. Veerasamy, D. D'Souza, R. Lindén, E. Kaila, M.-J. Laakso, and T. Salakoski,"The Impact of Lecture Attendance on Exams for Novice Programming Students", *International Journal of Modern Education and Computer Science(IJMECS)*, vol.8, no.5, pp.1-11, 2016.DOI: 10.5815/ijmecs.2016.05.01

[17] Y. Wang, H. Li, H., Y. Feng, Y. Jiang, and Y. Liu, "Assessment of programming language learning based on peer code review model: Implementation and experience report", *Computers & Education*, vol. 59, pp. 412-422, 2012. doi:10.1016/j.compedu.2012.01.007.

[18] B. C. Wilson and S. Shrock "Contributing to success in an introductory computer science course: a study of twelve factors." *ACM SIGCSE Bulletin*. vol. 33. no. 1, pp/ 184-188, 2001.

[19] L. E. Winslow, "Programming pedagogy – A psychological overview", *SIGCSE Bulletin*, vol. 28, pp. 17-22, 1996.

Authors' Profiles

Rex P. Bringula is a professor at the University of the East (UE) College of Computer Studies and Systems. He received his BS Computer Science degree from UE as a Department of Science and Technology scholar. He received his Master in Information Technology and Ph.D. in Technology Management in Technological University of the Philippines. He is active in conducting school- and government-funded research projects, and in participating in local and international conferences. His research interests are in computer science/IT education, affective computing, Internet studies, cyber-behavior, web usability, and environmental issues.

Ariel D.V. Aviles is the chairperson of the Department of Information Systems of the College of Computer Studies and Systems (CCSS), University of the East. He holds a bachelors degree in computer science, master's degree in Business Administration and a master's degree in Information Management. He obtained all of these from UE.

Ma. Teresa F. Borebor holds a bachelor's degree in Computer Science from the University of the East. She obtained her master's degree in Information Technology from the Technological University of the Philippines (TUP) as a Tan Yan Kee scholar. She is currently pursuing her Doctor of Technology at TUP. She is now the associate dean of the College of Computer Studies and Systems, UE.

Bernadette E. San Diego is currently the college secretary of the UE-CCSS. She has been teaching for more than 20 years at the said department. Her research interests are on human behavior and computing education. She holds two master's degrees (Business Administration and Information Management). She is now on her dissertation writing for the completion of the degree of doctor of education.

Mark Anthony D. Uy is an Information Technology graduate of UE-CCSS. He is a full-time faculty of the said college. He obtained his master's in Information Management at the same institution. He was a student leader during his college years.

Ma. Ymelda C. Batalla is the chairperson of the Entertainment and Multimedia Computing of UE-CCSS. She obtained her bachelor's degree in Computer Science and master's degree in Information Technology both in TUP.

How to cite this paper: Rex P. Bringula, Ariel D.V. Aviles, Ma. Ymelda C. Batalla, Ma. Teresa F. Borebor, Mark Anthony D. Uy, Bernadette E. San Diego," Factors Affecting Failing the Programming Skill Examination of Computing Students", International Journal of Modern Education and Computer Science(IJMECS), Vol.9, No.5, pp.1-8, 2017.DOI: 10.5815/ijmecs.2017.05.01

Copyright © 2017 MECS