Journal of Theoretical and Applied Information Technology
30th June 2017. Vol.95. No 12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ASSESSING THE EFFECTIVENESS AND USABILITY OF USING PAIR PROGRAMMING TO IMPROVE PROGRAMMING LANGUAGE LEARNING, PRODUCTIVITY, AND CODE QUALITY

[1]ZAID T. ALHALHOULI, [2]FARHAN M Al OBISAT, [3]TAMARA E. ALSHABATAT,
[4]TAMADOR I. ALRAWASHDEH

[1, 2, 3, 4] Department of Computer and Information Technology, Tafila Technical University, Jordan

E-mail: [1] zaid_halhouli@yahoo.com , [2] fobisat@ttu.edu.jo, [3] talshbat@gmail.com,
[4] alrawashdeh555@gmail.com

ABSTRACT

Programming is a major challenge faced by universities students at different levels. A significant learning outcome that emerged in recent years is working in teams, wherein two programmers can engage in pair programming to work on a single task as a group. This study examined the usability and effectiveness of using pair programming. We also evaluated the effects of pair programming on student results, time consumed, and number of errors, Big-O notation, and time complexity. The mixed method approach was applied by formulating a questionnaire and three different projects. We applied a novel strategy for creating pairs of students from different levels and courses in Tafila Technical University. Results indicated that pair programming is feasible and effective for educational purposes. Positive results were also obtained for programming learning, time, performance, and code quality.

Keywords: *Pair Programming Experience, Collaborative Learning, Teaching Methods, Usability, and Team Performance.*

1. INTRODUCTION

A common problem encountered by computer science students is computer programming. Many researchers have attempted to address the programming problem by testing different strategies and techniques, such as pair programming; however, the programming trend remains problematic. Pair programming is an extreme programming practice and an active software development method [1, 2]. Pair programming is a programming pattern that enables two programmers to work together on the same computer during a single duty; in this pattern, the leader works on the computer, whereas the observer checks for errors and provides useful suggestions [3].

Recent literature indicates that student programmers who participate in team learning and pair programming tend to perform better on coding projects and are more likely to succeed in early code development than those who are working on their own [4, 5]. Nonetheless, this approach also presents a number of drawbacks, such as differences in skills, technical competence, ability to work, and share knowledge within a team; improving coordination among the skills of students; and improving productivity with the shortest time and the slightest defects [6]. The most prominent challenge is cultural issues related to society when programmers are paired randomly [7].

This research aims to evaluate the effect of using pair programming technique on the student productivity from deferent perspectives and on the code quality by evaluates the use of this technique in three courses by examining various factors that may affect and enhance problem solving and code quality. The remainder of this paper is organized as follows. In Section 2, we describe the general background and enumerate empirical studies on pair programming. Section 3 briefly explains our methodology. The results and discussion are presented in Section 4. Section 5 explores research directions for future works and states the research limitations. Finally, the conclusion is provided in Section 6.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2. LITERATURE REVIEW

Many studies have been conducted to measure the effect of pair programming on the quality of the produced code, retention of students in programming courses, completion of a programming course, time of building and delivering programming tasks, and student grades in programming courses [4, 8, 9, 10, and 11] These studies highlighted the efficient use and advantages of using pair programming in the industrial and academics fields.

Powell et al. [4] declared that pair programming could improve overall productivity and software quality through collaboration. Cockburn et al. [12] indicated that compared with solo programmers, pair programmers produced higher quality code in approximately half the time. This result is consistent with the findings of Kitchen [13], who have determined that using pair programming will produce code with fewer defects and good quality in half the time, develop good knowledge, and build teamwork. The rotation of partners also helps programmers familiarize themselves with the overall system. Moreover, pair programming promotes programming productivity and knowledge transfer among software developers [14]. VanDeGrift [15] asserted that pair programming could improve software design, reduce code deficiencies, enhance technical skills, improve team communication, and create an enjoyable working environment.

Numerous guidelines must be followed in pairing students to achieve high effectiveness. Students should also be supervised by staff members or instructors who can resolve any problem that may arise between pairs. Student partners should also be rotated to help them develop their communication skills and prevent the development of intolerance to the same partner [16]. The educational benefits gained from applying the pair programming technique include superior results on graded assignments, increased satisfaction/reduced frustration among students, increased confidence of students regarding their project results, and reduced workload for the teaching staff [17, 18].

Several studies have measured the different aspects of using pair programming in various fields; accordingly, various methods have been implemented to form groups. Faja [19] argued that team composition was a crucial factor that would affect the collaboration and performance of teams. In the literature, various methods have been implemented to form pairs. These methods can be categorized as follows: 1) matching pairs versus random pair assignment and 2) same partners throughout the course versus changing partners. Howard [20] applied various pair formation approaches; he assigned pairs via random selection, paired students according to different Myer–Briggs personality types, and matched students on the basis of instructor observation. McDowell et al. [16] paired students based on their preferences or randomly if they did not have any preference and wished to remain with the same partner. Choi et al. [21] and Zacharis [22] assigned students pairs according to their grades. Braught [23] rotated pairs every three or four laboratory sessions. Bevan [24] suggested pairing students based on their skill levels. Studies have shown that if students were paired with less proficient partners, then they might consider the experience a waste of time and might complete the assignment on their own but submit it as a combined effort. Faja [19] and Chaparro et al. [25] found that matching skill levels and programming tasks would strongly influenced collaboration between pairs. Katira et al. [26] suggested that the compatibility of pair programmers would significantly affect their work productivity.

Braught et al. [23] found that gender had no effect on the confidence of the completed work. By contrast, Choi et al. [21] posited that pair groups with the same gender displayed higher levels of satisfaction, communication, and compatibility.

Freeman et al. [27], Salleh et al. [17], and Owolabi et al. [28] found no significant difference between pair and solo students in terms of academic performances (quizzes, final exams, and course grades) and time spent on a project. On the contrary, McDowell et al. [29] discovered that paired students performed significantly better than individual students in final exams and course

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

grades. Canfora et al. [30] reported that the time spent by pair programmers in programming was decreased compared with that spent by solo programmers, thereby refuting the claim made by Freeman. Agrawal et al. [31] found that pairs selected based on team ability, swapping flexibility, compatibility level, and application domain knowledge exhibit better performance than randomly selected pairs. Sfetsos et al. [32] discovered better performance and collaboration viability for pairs whose members exhibited heterogeneous personalities and temperaments.

Salge et al. [33] and Akour et al. [3] concluded that compared with individual leaning, pair programming was more effective in improving student learning of programming concepts.

Previous studies have indicated that using the pair programming technique presents numerous advantages in the academic field, and that various ways of forming pairs can be adopted. Accordingly, this study measures new aspects from different perspectives to observe the effects of pair programming on academic achievement. In the next section, we discuss the research methodology and the methodology used for group formation.

3. RESEARCH METHODOLOGY

A systematic approach was adopted for data collection and analysis [34]. A mixed method approach (combination of qualitative and quantitative methods) was applied in this study [35] based on the research objectives. The first objective was measured using a questionnaire at end of the research tests. This questionnaire evaluates the usability and effectiveness of using the pair programming technique. The second objective was tested by evaluating and analyzing the results of three student projects.

This study selected 18 questions from the Usability and User Experience (USE) questionnaire [36, 37] to measure the usability and effectiveness of using the pair programming technique. A total of 62 questionnaires were distributed to students enrolled in three different courses, namely, Computer Skills 2 (C++), Internet Programming, and Advanced Internet Programming. Only 58 questionnaires were returned. In addition, a set of factors were selected to measure their effect on a programming pair. These factors were duration time, design, problem solving, gender, and creativity in production. Other methods were also applied to form groups, such as random pairing and optional pairing, in which a student can choose his/her group. The research results are explained in detail in the next section.

4. FINDINGS AND DISCUSSION

This study explored and measured pair programming and its application in the academic field from different perspectives. The results shows that students benefit from having a partner for programming projects because partners help answer questions, share ideas and skills to the group, and help with debugging and problem solving. The results of the survey indicated that the majority of the students were interested to work in pairs on their projects. Most of the students believed that pair programming improved their grades. A detailed description is provided in the following sections.

4.1. Measuring the effect of adopting pair programming on students and code

We designed three projects for each course. The students were divided into pairs and individuals according to a method previously described in the research methodology. The experimental results indicated that adopting pair programming in the academic field enhanced learning programming languages and optimized the strength of a program, including its performance, Big-O notation, and time complexity.

Compared with solo programming, pair programming can produce shorter programs, implement better designs, contain fewer defects, and typically require less competence to complete a task. As shown in Figure 1, design effectiveness was measured based on project duration, effort, and quality. The programming efforts were distributed among individuals, thereby reducing the workload of an individual programmer. Consequently, a work package is completed within a shorter period, and coding does not require storage capacity. For example, when the students were asked to produce

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

a drop-down list of expected output with JavaScript, those that were produced using HTML code demonstrated that production under pair programming was better than that under individual programming. These findings are consistent with the results of many previous works, such as [10, 38].

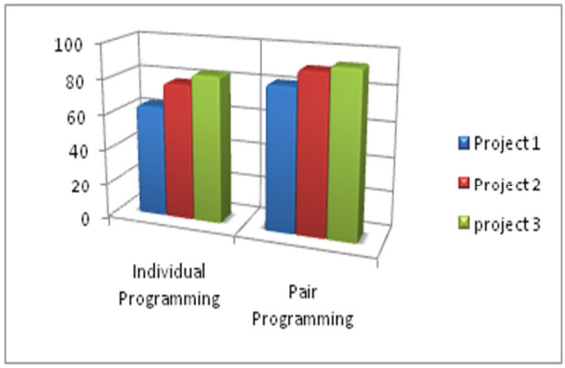

Figure 1: Rate of production in both projects

The total time for all the experiments (i.e., the time spent at work) with pair programming (estimated between 3 h and 3.5 h, Table1) was considerably less than that with individual programming. This finding is similar to the results obtained by Bipp et al. in 2008 [39].

Table 1: Time to Complete Each Project (in Person)

Programming type	Project1 (h)	Project2 (h)	Project3 (h)
Individual programming	4.37	4.20	3.30
Pair programming	3.16	3.05	2.20

The defect density was used to compare the effectiveness of pair programming and individual programming [40, 41]. Each line of code is a problem that should be solved using the pair programming technique.

Defect density = Lines of code/defects (1)

The code of any program contains errors. The results indicated that fewer mistakes were discovered when pair programming, rather than solo programming, was used. This finding is consistent with the result of [42]. In addition, pair programming can immediately solve problems and produce the best designs, thereby suggesting that having a pair with whom one can discuss ideas and address problems can significantly improve the quality and quality of work [43]. Table 2 provides the statistical results.

Table 2: Density Defect for Each Project (in Person)

Programming type	Line of code	Defects	Density defect
Individual programming	259	13	0.05 defect per LOC
Pair programming	172	6	0.03 defect per LOC

The resulting code had approximately 15% fewer defects [44]. For the test cases, the students passed each program, which was essentially the percentage of the instructor's test cases passed. Pair programming worked more effectively than individual programming twice and was closer to the course completion rate [45].

Project evaluation illustrates that paired students consider time as much as possible in completing work packages by searching for a code that takes less time to produce high-quality work and looking for inexpensive means without changing the course of action. In addition to working without errors, solutions should also reduce time complexity. Although individual programming can produce a code that can solve the problem without requiring a long implementation, a larger number of code lines is used in single programming compared with that in pair programming (Figure 2).

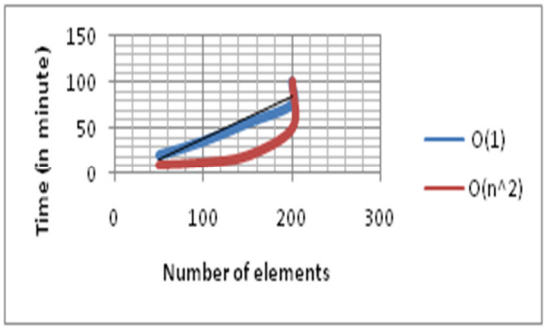

Figure 2: Big-O Complexity

4.2. Measuring the usability and effectiveness of pair programming

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SPSS v.21 was used in this study as the main data analysis technique to assess the usability and effectiveness of the pair programming technique. The students were presented with a brief illustration of the benefits of pair programming and how the features of this technique could be utilized to accomplish tasks without any problem and within the least time. This explanation was presented to the respondents, which mainly consisted of students from Tafila Technical University. The opinions of the students with regard to using pair programming were collected via a perception survey, which allowed the authors to discover factors that enhanced and motivated the students to use pair programming. First, equivalence test and reliability test were conducted to confirm the assumption of equal variances for the scores. The variance equivalence test for each group was confirmed, where $p < 0.05$ was recorded for each group. The reliability of the test results matches all the scales required for the Cronbach's alpha (0.70 and above), thereby indicating that the results are reliable [46], as shown in Table 3.

Table 3: Reliability Test of the Measurement Scales for Usability and effectiveness I

Variables	N of items	Cronbach's α
Usefulness (U)	4	0.843
Ease of Use (EU)	5	0.759
Ease of Learning (EL)	7	0.886
Satisfaction (S)	2	0.964

The analysis result of the students provides good insight into the opinions and positive feedback regarding the use of the pair programming technique. The survey results are shown in Table II. The results indicate that this technique is usable and effective, and thus, should be launched and used in the academic field, as suggested by the value of the mean for each of group in the study questionnaire, which is greater than 3.7, as shown in Table 4.

Table 4: Mean for System Usability Scale Questions

Variables	Mean	STD	Std. Error
Usefulness	3.7586	0.76515	0.10047
Ease of Use	3.8207	0.64638	0.08487
Ease of Learning	3.8916	0.77479	0.10173
Satisfaction	3.931	0.84005	0.11030

This study also assessed the usability and effectiveness of using pair programming from different perspectives and criteria, such as gender, course, specialization, and student level. The results show that females exhibit better mean values than males. In addition, Applied Physics and Geological Engineering have mean values greater than 4. All other specializations present a good mean; among which, the smallest value is 3.10, as shown in Table 5. Moreover, by comparing the results of the questionnaire according to student level, the researchers found that students at the fourth year level were more willing to use pair programming than that those in other levels. Furthermore, the mean values for all the involved students in all the courses were convergent, as shown in Table 6.

Table 5: Mean for System Usability Scale Questions According to Gender and Specialization

(at the end of this paper)

Table 6: Mean for System Usability Scale Questions According to Student Level and Course Name

(at the end of this paper)

5. CONCLUSION AND LIMITATIONS

This study found an apparent willingness among students to use pair programming in the academic field to enhance programming learning. Pair programming positively affected code quality, time, and student results. These results are congruent with those of other studies that have explored and studied the outcomes of and benefits from adopting pair programming in learning. This study expands knowledge on learning programming languages in different disciplines, fills in level gaps among students in the same class, enhances willingness to exchange knowledge, and promotes cooperation while working in groups.

Nevertheless, this study is hindered by a number of limitations. First, the small sample size may affect the perception of students about the importance of group work. Second, this work did not consider group size in assessing the outcomes of and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

benefits from using pair programming. Moreover, a set of limitations appeared during research and data collection, such as the non-desire to work in a group, difficulty of coordination among members, communication problems, and societal issues (e.g., mixed gender).

REFERENCES:

[1] Ally, M., Darroch, F. and Toleman, M. (2005) 'A framework for understanding the factors influencing pair programming success', Proceedings of the XP 2005 Conference, pp.82–91.

[2] Schmidt, C., Kude, T., Heinzl, A., & Mithas, S. (2014). How Agile practices influence the performance of software development teams: The role of shared mental models and backup.

[3] Akour, M., Al-Radaideh, K., Alazzam, I., & Mahmoud Alsmadi, I. (2013). Effective pair programming practice: toward improving student learning in software engineering class. International Journal of Teaching and Case Studies, 4(4), 336-345.

[4] Powell, L. M., & Wimmer, H. (2015). Evaluating the Effectiveness of Student Group Work for Mobile Application Development Learning, Productivity, Enjoyment and Confidence in Quality. In Proceedings of the EDSIG Conference (p. n3456).

[5] Williams, L., & Upchurch, R. L. (2001, February). In support of student pair-programming. In ACM SIGCSE Bulletin (Vol. 33, No. 1, pp. 327-331). ACM.

[6] Goel, S., & Kathuria, V. (2010). A novel approach for collaborative pair programming. Journal of Information Technology Education, 9, 183-196.

[7] Radhakrishnan, P., & Kanmani, S. (2012). Student's Opinion on Adopting Pair Programming as a Teaching and Learning Tool. International Journal of Computer Applications, 60(4).

[8] Powell, L. M., &Wimmer, H. (2016). Evaluating Students' Perception of Group Work for Mobile Application Development Learning, Productivity, Enjoyment and Confidence in Quality. Information Systems Education Journal, 14(3), 85.

[9] Braught, G., MacCormick, J., & Wahls, T. (2010, March). The benefits of pairing by ability. In Proceedings of the 41st ACM technical symposium on Computer science education (pp. 249-253). ACM.

[10] Praveen, H., Muralidhar, T., Vinod, Y.V., Padmanabhuni, K. and Madina, S. (2012) 'Effective pair programming practice an experimental study', Journal of Emerging Trends in Computing and Information Sciences, Vol. 3, No. 4, pp.471–479.

[11] Hannay, J.E., Arisholm, E., Engvik, H. and Sjoberg, D.I.K. (2010) 'Effects of personality on pair programming', Software Engineering, IEEE Transactions on, Vol. 36, No. 1, pp.61–80.

[12] Cockburn, A., & Williams, L. (2000). The costs and benefits of pair programming. Extreme programming examined, 223-247.

[13] Kitchenham, B. (2004). Procedures for performing systematic reviews. Keele, UK, Keele University, 33(2004), 1-26.

[14] Haungs, J. (2001). Pair Programming on the C3 Project. Computer, 34(2), 118-119.

[15] VanDeGrift, T. (2004, March). Coupling pair programming and writing: learning about students' perceptions and processes. In ACM SIGCSE Bulletin (Vol. 36, No. 1, pp. 2-6). ACM.

[16] McDowell, C., Werner, L., Bullock, H.E., Fernald, J. (2006). Pair Programming Improves Student Retention, Confidence, and Program Quality. Communication of the ACM, 49(8), 90-95.

[17] Salleh, N., Mendes, E., & Grundy, J. (2011). Empirical studies of pair programming for CS/SE teaching in higher education: A systematic literature review. IEEE Transactions on Software Engineering, 37(4), 509-525.

[18] Williams, L. A., & Kessler, R. R. (2000, March). The effects of" pair-pressure" and" pair-learning" on software engineering education. In Software Engineering Education & Training, 2000. Proceedings. 13th Conference on (pp. 59-65). IEEE.

[19] Faja, S. (2011). Pair programming as a team based learning activity: a review of research. Issues in Information Systems, 12(2), 207-216.

[20] Howard, E. V. (2007). Attitudes on Using Pair- Programming. Journal of Educational Technology Systems, 35(1), 89-103.

[21] Choi, K.S.., Deek., F.P. &Im., I. (2009). Pair Dynamics in Team Collaboration. Computers in Human Behavior, 25(4), 844-852.

[22] Zacharis, N. Z. (2011). Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course. IEEE Transactions on Education, 54(1), 168-170.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[23] Braught, G., Wahls, T., &Eby, L. M. (2011). The Case for Pair Programming in the Computer Science Classroom. ACM Transactions on Computing Education, 11(1), Forthcoming.

[24] Bevan, J., Werner, L., & McDowell, C. (2002). Guidelines for the use of pair programming in a freshman programming class. In Software Engineering Education and Training, 2002.(CSEE&T 2002). Proceedings. 15th Conference on (pp. 100-107). IEEE.

[25] Chaparro, E. A., Yuksel, A., Romero, P., & Bryant, S. (2005, June). Factors affecting the perceived effectiveness of pair programming in higher education. In Proc. PPIG (pp. 5-18).

[26] Katira, N., Williams, L., Wiebe, E., Miller, C., Balik, S., &Gehringer, E. (2004, March). On understanding compatibility of student pair programmers. In ACM SIGCSE Bulletin (Vol. 36, No. 1, pp. 7-11). ACM.

[27] Freeman, S. F., Jaeger, B. K., & Brougham, J. C. (2004). Pair programming: More learning and less anxiety in a first programming course. age, 8, 1.

[28] Owolabi, J., Adedayo, O. A., Amao-Kehinde, A. O., &Olayanju, T. A. (2013). Effects of Solo and Pair Programming Instructional Strategies on Students' Academic Achievement in Visual-Basic. Net Computer Programming Language. GSTF Journal on Computing (JoC), 3(3), 108.

[29] McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. ACM SIGCSE Bulletin, 34(1), 38-42.

[30] Canfora, G., Cimitile, A., &Visaggio, C. A. (2005, June). Empirical study on the productivity of the pair programming. In International Conference on Extreme Programming and Agile Processes in Software Engineering (pp. 92-99). Springer Berlin Heidelberg.

[31] Tripathi, M., Agrawal, A., Maurya, L. S., & Hora, H. (2013). Parametric Pair Programming-A Way towards Optimum Output. International Journal of Scientific and Research Publications, 506.

[32] Sfetsos, P., Stamelos, I., Angelis, L., &Deligiannis, I. (2009). An experimental investigation of personality types impact on pair effectiveness in pair programming. Empirical Software Engineering, 14(2), 187- 226.

[33] de Lima Salge, C. A., & Berente, N. (2016, January). Pair Programming vs. Solo Programming: What Do We Know After 15 Years of Research?. In System Sciences (HICSS), 2016 49th Hawaii International Conference on (pp. 5398-5406). IEEE.

[34] Dybå, T., Arisholm, E., Sjøberg, D. I., Hannay, J. E., & Shull, F. (2007). Are two heads better than one? On the effectiveness of pair programming. IEEE software, 24(6), 12-15.

[35] Creswell, J. W. (2013). Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications.

[36] Lund, A. M. (2001). Measuring Usability with the USE Questionnaire12.". Usability interface, 8(2), 3-6.

[37] Hassan, Z. B. H., & Alhalhouli, Z. T. (2014, November). The stakeholders satisfaction about the using social networks in Jordanian hospitals for knowledge sharing. In Information Technology and Multimedia (ICIMU), 2014 International Conference on (pp. 163-168). IEEE.

[38] Chong, J., Plummer, R., Leifer, L., Klemmer, S. R., Eris, O., & Toye, G. (2005, June). Pair programming: When and why it works. In Proc. The 17th Workshop of the Psychology of Programming Interest Group PPIG17 (pp. 43-48).

[39] Bipp, T., Lepper, A., & Schmedding, D. (2008). Pair programming in software development teams–An empirical study of its benefits. Information and Software Technology, 50(3), 231-240.

[40] Padberg, F., & Muller, M. M. (2003, September). Analyzing the cost and benefit of pair programming. In Software Metrics Symposium, 2003. Proceedings. Ninth International (pp. 166-177). IEEE.

[41] Rahman, F., Khatri, S., Barr, E. T., & Devanbu, P. (2014, May). Comparing static

[42] bug finders and statistical prediction. In Proceedings of the 36th International Conference on Software Engineering (pp. 424-434). ACM.

[43] Frentiu, M. I. L. I. T. O. N. (2005). Correctness: a very important quality factor in programming. Studia Universitas Babes-Bolyai, Seria Informatica L (1), 11-20.

[44] Anjum, S., Batul, H., & Sirshar, M. (2015). A Comparative Sudy of Ensuring Quality in Pair Programming.

[45] Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

case for pair programming. IEEE software, 17(4), 19.

[46] Waller, L. R., & Tietjen-Smith, T. (2009). A national study of community college retention rates segmented by institutional degree of urbanization. Academic Leadership, 7(1), 1-3.

[47] Hair, J. F. (2009). Multivariate data analysis.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 5: Mean For System Usability Scale Questions According To Gender And Specialization

Criteria / Factors	Gender		Specialization									
	Male	Female	Computer Information Systems	Geological Engineering	Electrical Power Engineering	Mechatronics	Chemistry Technology	Civil Engineering	Chemical engineering Industries	Mechanical Engineering	Applied Physics	Mathematics
N	35.00	23.00	33.00	1.00	2.00	2.00	4.00	5.00	1.00	6.00	3.00	1.00
Usefulness	3.71	3.84	3.73	4.25	4.00	3.88	3.13	3.55	4.00	4.25	3.75	4.00
Ease of Use	3.62	4.13	3.84	4.20	4.00	3.80	3.10	3.52	3.80	4.00	4.47	3.80
Ease of Learning	3.66	4.25	3.90	4.43	3.14	3.50	4.29	3.43	4.00	3.76	4.81	4.00
Satisfaction	3.86	4.04	3.97	4.00	3.00	4.50	4.50	3.70	4.00	3.58	4.00	4.00

Table 6: Mean For System Usability Scale Questions According To Student Level And Course Name

Criteria / Factors	Student level				Course name		
	1st year	2nd year	3rd year	4th year	Internet programming	Advanced Internet programming	Computer skills 2 C++
N	1.00	20.00	26.00	11.00	17.00	14.00	27.00
Usefulness	4.25	3.81	3.63	3.93	3.59	3.86	3.81
Ease of Use	4.00	3.87	3.75	3.89	3.78	3.90	3.81
Ease of Learning	2.29	3.70	4.02	4.09	4.04	3.84	3.83
Satisfaction	2.00	3.70	4.08	4.18	4.29	3.71	3.81