JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 19, No. 9, pp. 1663-1669, Sep. 2018

프로그래밍 실습수업에서의 짝 프로그래밍: 학생들의 수용성(受容性)을 중심으로 본 효과와 한계, 운영 방안

정 충 교
강원대학교 컴퓨터학부

Pair Programming in Programming Lab: The Effects, Limits, and Guidelines Based on the Student Receptivity

Choong-Kyo Jeong
Department of Computer Science and Engineering, Kangwon National University, 24341, Korea

[요 약]

짝 프로그래밍은 두 사람이 하나의 컴퓨터에서 코드를 작성하는 소프트웨어 개발 방법이다. 한 사람은 코드를 입력하고 다른 사람은 옆에서 훈수를 두는데 두 사람은 자주 역할을 바꾼다. 프로그래밍 실습 수업에 짝 프로그래밍을 적용하면 학습 성과 향상, 협동 작업 연습, 교류 증진 등 여러 가지 이득을 기대할 수 있다. 이 연구에서는 대학 프로그래밍 실습에 짝 프로그래밍을 적용하고 설문 조사를 통해 학생들이 짝 프로그래밍을 얼마나 잘 받아 들이는지, 짝 프로그래밍을 받아들이기 어렵게 하는 요소가 무엇인지를 조사하였다. 조사 결과를 바탕으로 짝 프로그래밍을 도입할 때 고려할 사항을 가이드라인으로 제시하였다. 이를 요약하면, 학생이 짝 프로그래밍 참여 여부를 선택할 수 있게 해야 할 것이며, 역할 교대를 방해하는 요인들을 제거해야 하고, 짝 배정에 세심한 노력을 기울여야 한다는 것이다.

[Abstract]

Pair programming is a software development technique in which two programmers work together at one computer. One writes code while the other reviews the code, and they switch roles frequently. Pair-programming practice in school programming lab is expected to improve the learning performance, provide collaboration experience, and promote interactions between students. This work finds out how students accept pair-programming, what make students reluctant to join pair-programming by repeated questionnaire surveys in a college programming lab class. Based on these findings aome guidelines for school pair-programming are provided. First, students should be allowed to choose to do pair-programming or not. Second, various obstacles that make students hesitate to switch roles should be removed. Third, the pair matching should be made with great care.

색인어: 협동작업, 가이드라인, 프로그래밍 실습, 짝 프로그래밍, 역할교대

Key word: Collaboration, Guideline, Programming lab, Pair-programming, Role change

http://dx.doi.org/10.9728/dcs.2018.19.9.1663

Received 24 August 2018; **Revised** 15 September 2018
Accepted 27 September 2018

Corresponding Author; Choong-Kyo Jeong

Tel: +82-33-250-6325
E-mail: ckjeong@kangwon.ac.kr

Ⅰ. 서 론

프로그래밍이 보편적으로 필요한 역량으로 인식되면서 초중등학교와 대학에서 프로그래밍 교육이 확대되고 있다. 그러나 프로그래밍을 처음 배우는 학생들 중 상당수는 프로그래밍을 이해하고 그 기술을 습득하는 데 큰 어려움을 느낀다. 짝 프로그래밍은 이 어려움을 완화하려는 노력으로 교육 현장에서 시도되는 교육 방법 중 하나이다.

원래 짝 프로그래밍은 소프트웨어 개발 현장에서 한 개발자가 작성한 코드를 다른 개발자가 검토하는 작업인 "코드 리뷰"를 더욱 적극적으로, 실시간으로 수행하는 개발 방법론의 일종이다[1]. 이 방법에서는 두 사람이 짝을 이뤄 프로그램을 작성한다. 한 사람은 운전사(driver) 역할을 맡아 주도적으로 프로그램을 작성한다. 다른 한 사람은 항해사(navigator) 역할을 맡는데 항해사는 프로그램 코드를 입력하는 대신 운전사가 입력하는 코드를 살펴보면서 문제점을 발견하고 대안을 제시하는 등 '훈수'를 둔다. 그리고 협업의 효과를 높이기 위해 주기적으로 (약 20분 정도의 주기) 두 사람의 역할을 바꾼다. 두 사람이 함께 고민할 만한 복잡도의 프로젝트에서 두 사람의 실력 수준이 비슷하고 성격 궁합이 잘 맞는 경우 팀웍 관리, 프로젝트 소요시간, 결과물의 품질 등에서 이득이 있는 것으로 알려져 있다[2].

프로그래밍 실습수업에 짝 프로그래밍을 적용하는 경우 여러 가지 이득을 기대할 수 있다. 학생들은 짝 프로그래밍을 통해 프로그래머의 핵심 덕목인 개방, 협업, 소통, 공유, 배려 등의 가치를 실천적으로 체득할 기회를 갖게 된다. 학생들은 프로그래밍을 처음 배울 때 여러 시행착오를 겪으면서, 도구 사용법으로부터 논리 구성 요령까지 갖은 종류의 경험적 지식을 쌓게 되는데, 이런 경험을 둘이 공유하면서 학습 속도와 학습 수준이 높아질 수 있다. 문제에 부딪혔을 때 둘이 협의함으로써 혼자 끙끙댈 때에 비해 좌절하지 않고 문제를 극복할 힘을 얻게 되기도 된다. 그리고 이런 긍정적 요소들이 학생들의 학업 성취도에도 영향을 주어 전반적인 성과를 기대할 수 있다.

그러나 이런 기대는 그야말로 이론적인 기대일 뿐, 실제로 이런 효과가 나타나는지 확인할 필요가 있다. 짝 프로그래밍의 효과를 확인하기 위해 구체적으로 살펴볼 필요가 있는 것은 크게 두 가지이다. 하나는 학습성과, 즉 짝 프로그래밍 방식으로 실습을 하면 그렇지 않은 경우에 비해 학생들이 프로그래밍을 더 잘하게 되는지 여부이다. 다른 하나는 학생들의 수용성, 즉 학생들이 짝 프로그래밍을 거부감 없이 잘 받아들이고 수업이 즐겁다고 느끼게 되는가 여부이다. 통제된 실험환경에서 짝 프로그래밍이 학습 성과 측면에서 효과가 있다고 하더라도 학생들이 이를 받아들이기를 거부하거나 수업 만족도를 저하시킨다면 실질적 효과를 얻기 어렵기 때문이다.

나는 2016년 2학기에 대학 자바프로그래밍 수업에 짝 프로그래밍을 적용하고 시험 성적을 기준으로 삼아 짝 프로그래밍이 학습 성과를 높이는 데 효과가 있는지 살펴보았다[3]. 두 개 분반이 같은 내용의 수업을 하고 중간고사를 치른 후, 한 반에는 짝 프로그래밍을 적용하고 다른 분반에는 적용하지 않았다. 짝 프로그래밍을 적용한 분반은 모든 학생에게 강제로 적용하였다. 그리고 기말고사 성적이 중간고사 성적에 비해 얼마나 높아졌는지 관찰하는 방법으로 짝 프로그래밍이 학습 성과에 미치는 영향을 측정하였다. 짝 프로그래밍을 적용한 분반의 평균 성적이 짝 프로그래밍을 적용하지 않은 반에 비해 꽤 많이 상승했지만 통계적으로 확신을 줄 만한 수준에까지 이르지는 못했다. 적지 않은 학생들이 프로그래밍을 어렵다고 느끼면서 중도에 포기하는데 이런 탓에 점수의 표준편차가 커지고 그에 따라 통계적 유의성 검정에서 p-value가 커진 탓이다. 외국의 다른 연구에서도 학습 성과가 높아진다는 것이 대체적인 결론이지만 학습 성과가 의미 있게 높아지는 것으로 일관되게 나타나지는 않는다[4].

2017년 2학기 자바프로그래밍 수업에서는 두 분반 모두에 짝 프로그래밍을 적용하되 학생이 짝 프로그래밍 여부를 임의로 선택할 수 있게 하고, 짝 프로그래밍에 대한 학생들의 수용성을 집중적으로 관찰하였다. 이를 위해 한 학기 수업을 진행하면서 2주 간격으로 모두 여섯 번에 걸쳐 설문 조사를 했고 매 설문 조사 결과에 따라 적절히 수업 방식을 조정하였다. 이 글은 설문 조사 결과에 나타난 학생들의 태도를 분석하고 이를 기초로 짝 프로그래밍의 효용성을 평가한다. 또, 짝 프로그래밍을 학교 실습수업에 도입하는 경우 적용할 바람직한 운영 방안을 제시한다. 짝 프로그래밍에 대한 과거 연구 리뷰는 이전 논문[3]에 들어 있으므로 이 글에서는 생략한다.

Ⅱ. 수업 운영과 설문 조사

수업 첫 주에 짝 프로그래밍의 목적과 효용, 실행 방법과 주의점을 학생들에게 말해 주고, 기본적인 프로그래밍 능력을 실기문제 풀이를 통해 테스트하였다. (이 수업은 1학기에 개설되는 프로그래밍기초 과목의 후속 과목이므로, 학생들은 이미 프로그래밍의 기초 능력을 갖추고 있다.) 두 번째 주부터 교실 실습에 짝 프로그래밍을 적용하되 원하는 학생만 짝 프로그래밍을 하고 그렇지 않은 학생은 혼자서 실습을 하도록 허용하였다. 짝 프로그래밍에 참여하는 학생들이 스스로 정한 짝은 그대로 인정하고, 짝을 스스로 정하지 못한 학생들은 프로그래밍 실기 능력이 비슷한 학생들을 짝으로 맺어 주었다. 학생들의 프로그래밍 실기 능력은 첫 수업 시간의 프로그래밍 실기 능력 테스트 결과로 평가하였다.

그 이후로는 2주마다 짝 프로그래밍을 원하는지, 원하는 경우 짝 변경을 원하는지, 짝 변경을 원하는 경우에는 어떤 학생을 새 짝으로 삼기를 원하는지 조사하였다. 짝 프로그래밍을 하고 싶지 않은 학생은 그 이후로 혼자서 실습을 하도록 허용했고, 짝을 바꾸고 싶은 학생들은 가능한 한 희망에 따라 짝을 맺어주었다.

표 1. 설문 조사 개괄.
Table 1. Questionnaire surveys summary.

Surveys	1st	2nd	3rd	4th	5th	6th
Date	9.8	9.22	10.13	10.27	11.17	12.15
Responses	84	82	81	82	71	78
Respondents who want pair-programming	78	71	71	67	58	
Paired students	82	74	74	72	70	
Students who told they practiced pair-programming for the last two weeks		75	71	71	60	
Students who want to keep current partner		52	53	63	49	
Students who find new partner by themselves	26	2	8	0	4	
Students who want to be pair-matched by the administrator	52	17	10	4	5	
Students total	88					

표 2. "짝 프로그래밍이 학습에 효과적인가?" 질문에 대한 응답.
Table 2. Responses to the question "Is pair-programming effective to learn programming?".

	3rd			4th			5th		
	Responses		Ratio(%)	Responses		Ratio(%)	Responses		Ratio(%)
Strong positive	45	63.4	87.3	38	53.5	83.1	34	56.7	70.0
Positive	17	23.9		21	29.6		8	13.3	
Neutral	7	9.9	12.7	10	14.1	16.9	14	23.3	30.0
Negative	2	2.8		2	2.8		4	6.7	
Strong Negative	0	0.0		0	0.0		0	0.0	
Sum	71	100.0	100.0	71	100.0	100.0	60	100.0	100.0

또, 2주마다 (중간고사, 기말고사 기간 제외) 짝 프로그래밍에 대한 학생들의 소감을 설문을 통해 조사하였다. 짝 프로그래밍이 학습에 효과적이라고 느끼는지, 공동 작업 연습에 효과적이라고 느끼는지를 중점적으로 물어보았지만 수업 진행에 따라, 새로운 궁금증이 생김에 따라, 설문 문항을 조금씩 다르게 조정하였다. 구체적인 설문 문항은 강좌 홈페이지[5]에서 볼 수 있다.

표 1은 설문 조사에 관한 전반적인 데이터를 보여준다. 다섯 번의 설문 조사는 짝 프로그래밍에 대한 학생들의 소감과 개선 희망, 짝 배정 희망 등을 조사했고, 강좌 마지막에 실시한 6차 조사는 강좌 전체의 내용과 짝 프로그래밍에 대한 전반적인 소감을 물어보았다. 1차 조사에서 수강생 88명 중 78명이 짝 프로그래밍을 희망했는데 설문에 응답하지 않은 4명도 짝 배정을 해 주었다. 이때 서로 상의해서 스스로 짝을 정한 학생들이 26명이고 52명은 교수에게 짝을 배정해 달라고 요청하였다. 그 이후로도 설문 조사에 응답하지 않은 학생들에게 짝을 배정해 주었는데 그 중 짝 프로그래밍을 한 학생도 있고 그렇지 않은 학생도 있었다.

Ⅲ. 학습 효과

강좌가 어느 정도 진행된 후인 3차, 4차, 5차 조사에서는 '짝 프로그래밍에 참여한' 학생들에게 짝 프로그래밍이 학습에 효과적이라고 느끼는지 물었다. 표 2에서 보듯이 긍정 응답이 3차에서는 87.3%, 4차에서는 83.1%, 5차에서는 70.0%였다. 강좌 초기에 짝 프로그래밍을 소개하고 참여를 유도했기 때문에 초기에는 학생들이 기대를 갖고 긍정적으로 평가하는 효과가 더해져 높은 평가가 나왔으나 시간이 갈수록 실익을 냉정하게 판단하여 답하게 됐다고 생각한다.

6차 조사에서는 '모든' 학생에게 짝 프로그래밍을 내년 수업에도 적용하는 데 찬성하는지 물었는데 긍정이 69.2%, 부정이 30.8%였다(표 3). 이는 5차 조사에서 '짝 프로그래밍에 참여한' 학생들에게만 짝 프로그래밍이 학습에 효과적이라고 느끼는지 물었을 때의 긍정, 부정 응답 비율과 비슷하다. 흥미로운 점은 6차 조사에서 짝 프로그래밍에 참여하지 않은 학생들(10명) 중 3명이 부정 응답을, 7명이 긍정 응답을 했다는 점이다. 즉, 자신은 나름대로 이유가 있어서 짝 프로그래밍에 참여하지 않았지만 짝 프로그래밍 자체에 대해서는 좋게 평가하고 다른 학생에게 권장한 학생이 70%였다. 전반적으로 보아 짝 프로그래밍을 긍정적으로 평가하는 학생이 70%를 넘지 않는다는 사실로부터 모든 학생에게 강제로 짝 프로그래밍을 강요하는 것은 바람직하지 않다고 말할 수 있다.

표 3. "내년에도 짝 프로그래밍을 하는 데 찬성하는가?" 질문에 대한 응답.
Table 3. Responses to the question "Do you recommend pair-programming for the next year class?".

	6th	
	Responses	Ratio(%)
Positive	54	69.2
Negative	24	30.8
Sum	78	100.0

Ⅳ. 역할 교대

3차, 4차 조사에서는 지난 2주간 짝 프로그래밍을 한 학생을 대상으로, 일정 시간 간격(약 15분)으로 짝과 운전사와 항해사로서의 역할을 서로 바꾸는 것이 학습에 어떤 영향을 미치는지 물었다. 이에 대한 응답은 표 4와 같다. 무관하다는 답은 역할 바꾸기가 별 효과 없다는 대답이므로 부정적 답변으로 분류할 수 있으며 결국 짝 프로그래밍을 한 학생들 중 약 40%만 역할 바꾸기가 학습에 도움이 된다고 답을 했다.

5차 조사에서는 지난 2주간 짝 프로그래밍을 한 학생을 대상으로, 짝이 자기보다 프로그래밍을 더 잘 하는지와 일정 시간 간격으로 짝과 역할을 바꾸는 데 대해 어떻게 생각하는지를 물었다. 구체적으로는, "적당한 시간 간격으로 운전사, 항해사 역할을 바꿔가면서 공동으로 프로그램을 작성하는 편이 좋다", "고정적으로 한 사람이 주도하여 프로그램을 작성하고 다른 사람은 보조적인 역할을 하는 편이 좋다", "각자 프로그램을 작성하면서 필요할 때만 서로 물어보거나 토론하는 편이 좋다"의 세 가지 중 하나를 선택하게 하였다. 그 결과 첫 번째를 선택한, 그러니까 역할 교대가 학습에 효과적이라고 느끼는 비율이 26.8%였다. 이는 3차와 4차에 비해 더욱 줄어든 수치이다. 두 번째를 선택한 학생은 17.9%이고, 세 번째를 택한 학생은 55.4%였다. 주기적 역할 교대에 대해 전반적으로 부정적인 평가를 하고 있으며 시간이 갈수록 더 부정적으로 변하였다.

자신의 프로그래밍 실력을 짝과 비교했을 때 자신의 수준을 어떻게 평가하는지에 따라 학생들을 분류한 후, 각 그룹이 역할 교대에 대해 어떻게 평가하는지를 살펴본 결과가 표 5이다. 우선 짝과의 실력 비교 평가에서 자신이 짝보다 더 잘한다고 답한 사람이 6명(10.7%)으로 매우 적은 반면, 짝이 더 잘한다고 답한 학생이 27명(48.2%)이나 된다. 나머지 23명(41.1%)은 비슷하다고 답했다. 학생들이 자신의 실력을 실제보다 낮게 스스로 평가했을 수도 있고, 실제 느끼는 것과 달리 겸손하게 답을 했 수도 있다. 이 결과는 미국 대학의 비슷한 조사 [6]에서 학생들이 균형이 맞는 응답을 한 예에 비추어 보아도 특이한 점으로 꼽힌다. 짝에 비해 자신의 프로그래밍 능력이 낫다고 생각하는 소수의 학생들은 역할을 바꾸지 않고 고정하는 쪽이 좋다고 답하였는데 이것은 자기가 고정적으로 운전사 역할을 맡고 싶다는 의미로 해석된다. 자기 실력이 짝에 비해 못하거나 비슷하다고 응답한 대부분의 학생들은 일관되게 각자 하는 편이 낫다고 대답한 비율이 높고, 역할을 고정적으로 나누고 싶어 하지는 않았다.

표 4. "주기적 역할 교대가 학습에 효과적인가?" 질문에 대한 응답.
Table 4. Responses to the question "Is the periodic role change effective to learn programming?".

	3rd			4th		
	Responses	Ratio(%)		Responses	Ratio(%)	
Positive	31	43.7	43.7	29	40.8	40.8
Neutral	29	40.8	56.3	30	42.3	59.2
Negative	11	15.5		12	16.9	
Sum	71	100.0	100.0	71	100.0	100.0

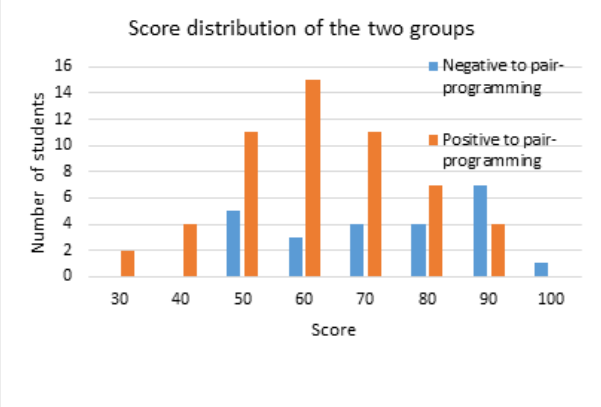

그림 1. 짝 프로그래밍에 대한 태도별 성적 분포.
Figure 1. Score distribution of the two groups of students (negative and positive to pair-programming).

짝 프로그래밍에서는 두 사람이 15분 정도의 시간 주기로 운전사와 항해사 역할을 바꾸는 것이 핵심이다. 그런데 실제로는 학생들이 자리를 바꿔 앉는 걸 매우 귀찮게 여긴다. 그래서 시간이 되면 자리를 바꿔 앉도록 독려해야만 했다. 그런데 학기가 지남에 따라 자리를 바꾸지 않는 학생들이 점점 많아졌다. 독려가 소홀해진 탓도 있을 것이고 학생들이 효용성을 느끼지 못하면서 역할 교대를 하지 않게 된 탓도 있을 것이다. 실습실 환경도 큰 영향을 미쳤다고 생각한다. 실습실은 한 줄에 여덟 자리씩 여섯 줄로 교탁을 향하여 평행으로 비좁게 자리가 배치되어 있다. 두 사람 각각 컴퓨터가 있으므로 두 대 중 한 대는 항해사가 웹 검색을 하는 데 사용하고 다른 한 대는 운전사가 코딩에 사용한다. 두 사람이 역할을 바꾸려면 일어서서 자리를 바꿔 앉아야 하는데 이렇게 하기에는 물리적으로 매우 불편한 환경이다. 짝 프로그래밍에 적합하도록 자리를 여유 있게 배치한다면 역할 교대의 부담이 훨씬 적을 것이다.

Ⅴ. 성적과 짝 프로그래밍 만족도 관계

표 5. 짝과의 실력 비교와 역할 분담 방식에 대한 선호.
Table 5. Subjective programming skill comparison with the partner and preferences to the role sharing method.

	Partner is far better		Partner is better		Partner is as good as me		Partner is worse		Partner is far worse		Sum	
	Responses	Ratio (%)	Responses	Ratio (%)	Responses	Ratio (%)	Responses	Ratio (%)	Responses	Ratio (%)	Responses	Ratio (%)
Role change	4	26.7	3	25.0	7	30.4	1	20.0	0	0.0	15	26.8
Fixed role	2	13.3	3	25.0	0	0.0	4	80.0	1	100.0	10	17.9
Parallel	9	60.0	6	50.0	16	69.6	0	0.0	0	0.0	31	55.4
Sum	15	100.0	12	100.0	23	100.0	5	100.0	1	100.0	56	100.0

표 6. 짝을 바꾸지 않는 이유.
Table 6. The reason not to change the partner.

What is the reason that you are not changing your partner?	Responses
We get along well.	46
He/she is of great help when I do the assignment.	42
We became friends.	38
A part of me wants to change, but I worry that it might make my partner feel bad.	5
A part of me wants to change, but I think my chances of meeting a better partner are low.	3
A part of me wants to change, but I don't feel comfortable about having to get to know a new person.	2
A part of me wants to change, but I am reluctant to request a change when most people in class are sticking to their current partner.	1
Other.	6

그림 1은 6차 조사에 참여한 78명의 학생들 중 짝 프로그래밍에 대해 긍정적으로 평가한 학생과 부정적으로 평가한 학생들의 최종 성적 분포를 보여준다. 대체로 부정적으로 평가한 학생들이 높은 성적, 혹은 낮은 성적을 보이며, 긍정적으로 평가한 학생들은 중간 성적을 보이는 경향이 있다. 이는 성적이 높은 학생의 경우 짝 프로그래밍에 대해 부정적인 경향이 있다는 이전 연구에서 발견한 사실과 일치한다[3]. 학생들은 대체로 자기와 실력이 비슷하거나 약간 더 잘하는 사람과 짝을 맺기를 희망한다. 잘 하는 학생들의 만족도가 낮은 것이 적당한 사람과 짝을 맺지 못한 탓인지는 알 수 없다. 그러나 잘 하는 학생 그룹이 학생 수도 작고 서로 간의 실력 편차도 크기 때문에 이상적인 짝을 찾기 어렵다는 점은 일반적인 제약이라 하겠다.

VI. 짝 배정

학생들은 언제든지 짝을 바꿀 수 있도록 했고, 짝을 바꾸고 싶고, 누구와 짝을 하고 싶다거나 어떠어떠한 학생을 짝으로 삼고 싶다고 설문 조사 때 응답하면, 교수가 가능한 범위에서 희망에 맞는 새 짝을 배정해 주었다. 강좌 초기, 짝 프로그래밍을 소개할 때는 짝을 바꿈으로써 성격이 다른 여러 사람과 협동작업 연습을 할 수 있고, 여러 친구를 사귈 기회가 되기도 하니까 적극적으로 짝을 바꾸어 보라고 권장하기도 했다. 그러나 학생들은 한 번 짝이 정해지고 나면 좀처럼 짝을 바꾸려고 하지 않았다(표 1). 4차 설문 조사에서는 짝을 유지하겠다고 답한 63명의 학생들에게 짝을 바꿈으로써 얻는 이득이 있음에도 불구하고 짝을 바꾸지 않는 이유가 무엇인지 물었다. 이 문항에는 중복 답변이 가능했다. 결과는 표 6과 같다. 대체로 한 번 짝이 정해지면 그 짝을 받아들이고 적응하는 경향이 있으며 여러 사람과의 다양한 짝 경험을 추구하지 않는다. 이런 상황에서는 짝을 바꾸고 싶은 일부 학생들도 짝을 바꾸려는 시도를 하기가 부담스러워진다. 짝 프로그래밍에 참여하는 학생들 모두를 강제로 주기적으로 짝을 바꿔주는 방안도 시도할 만하다고 생각한다. 마지막 6차 설문 조사의 서술식 답변 중에 전체 짝을 강제로 재배정하면 좋겠다는 의견이 적지 않게 (6명) 있기도 했다. 학생들의 수준과 성격에 대한 세밀한 관찰이 병행되면 좋은 효과가 날 가능성도 있다고 본다.

VII. 과제 결과물 제출

매주 두 사람이 짝을 이뤄 프로그램을 작성하고 그 결과물을 공동으로 제출하게 했다. 과제 결과를 채점한 점수는 성적에 반영된다. 그런데 매주 과제의 양이 많아 학생들이 과제를 실습 시간에 완료하지 못하고 수업 후에 추가로 작업을 해서 제출해야 했다. 수업 후에도 개방 실습실 등에서 공동으로 작업 하여 결과물을 완성하여 제출하도록 권유했으나 실제로는

수업 후에는 두 사람 중 한 사람이 남은 부분을 완성하여 공동 이름으로 제출하는 경우가 적지 않았다. 이럴 경우 둘 중 한 사람이 소외된다. 실습 시간에 풀 과제는 모두가 실습시간 내에 풀 수 있는 정도로 분량을 줄이고, 숙제 과제를 별도로 제시하여 숙제는 각자 제출하도록 하는 편이 짝 프로그래밍 참여 의욕을 떨어뜨리지 않는 방법이 될 것으로 생각한다.

Ⅷ. 자리 배치

짝 프로그래밍 방식으로 실습을 진행해 보면 그렇지 않은 때에 비해 교수진을 향한 학생들의 질문이 현저히 줄어들고 짝과의 토론이 많아진다. 그래서 교수와 조교는 질문 답변의 부담에서 벗어나 적극적, 능동적으로 실습을 지도할 여유를 갖게 된다. 학생들은 대개 자기보다 못하는 사람과는 짝을 하고 싶어 하지 않으므로 잘 못하는 학생들은 대체로 그들끼리 짝을 맺게 되는데 이들은 낙담하고 탈락하기 쉽다. 이런 학생들은 질문도 능동적으로 하지 않는 경향이 있다. 이들을 교수와 조교가 접근하기 쉬운 통로 근처에 앉게 한 후 적극적으로 도와주면 상당한 성과가 있는 것으로 보인다.

Ⅸ. 결론

학생들의 약 70% 정도가 짝 프로그래밍에 대해 학습에 도움이 된다거나 내년에도 적용하면 좋겠다는 등의 긍정적인 평가를 했다. 그러나 짝 프로그래밍의 핵심이라고 할 수 있는 운전사, 항해사 역할 교대가 효과적인가에 대해서는 약 27%의 학생만이 긍정적으로 평가했다. 더 많은 학생들은 짝을 맺어 학습하되 프로그램 작성은 각자 하고 필요할 때마다 서로 도움을 주는 방식을 선호했다.

그러나 운전사, 항해사 역할 분담과 주기적 역할 교대가 없이는 짝 프로그래밍이라고 할 수 없다. 두 사람이 짝을 맺기는 하지만 단순히 각자 학습하면서 서로 돕는 정도에 그치기 때문이다. 이럴 경우에도 혼자 학습할 때에 비해 학습효과가 높을 수 있고 친구를 사귀는 이득도 있겠지만 짝 프로그래밍은 정도에서 그치지 않고 더 많은 효과를 얻고자 하는 것이다. 짝 프로그래밍을 제대로 할 때 학생들은 단축키 등 도구 사용 요령, 코딩 스타일 등 단순한 것부터 알고리즘이나 테스트 방법과 같이 복잡한 것에 이르기까지, 짝과 서로 다른 점을 통해 효과적으로 학습을 하게 된다. 협업의 정도가 한층 높고 이를 통해 더욱 높은 효과를 얻고자 하는 것이다.

강제로 짝 프로그래밍을 시행한 이전 연구를 통해 많은 학생들에게서 학습 성과가 뚜렷이 높아짐을 확인할 수 있었다. 그러나 그 당시에도 학생들이 역할 교대를 잘 하는 경우도 있었고 그렇지 않은 경우도 꽤 있었으므로 그 효과는 단순 협조와 짝 프로그래밍 등 두 가지 효과가 합쳐진 결과라고 할 것이다. 학생들이 짝 프로그래밍을 더 잘 받아들이고 충실히 실행하면 협업의 효과가 더욱 클 가능성이 있다고 생각한다.

본문에서 분석한 내용을 기초로 하여 학생들이 짝 프로그래밍을 잘 받아들이게 하려면 무엇에 신경을 써야 하는지 정리하면 다음과 같다. 첫째로, 약 70%의 학생들이 짝 프로그래밍을 긍정적으로 평가했지만 이들은 역할 교대를 하지 않고 단순히 두 사람이 협조를 하는 것도 짝 프로그래밍이라고 여기고 답을 했다고 보인다. 30% 정도의 학생들은 아예 홀로 하는 학습을 선호한다고 생각할 수 있고 따라서 모든 학생에게 짝 프로그래밍을 강제로 적용하는 것은 옳지 않다고 여겨진다. 특히 상위권 학생들은 혼자 학습하기를 원하는 경향이 뚜렷하다. 학생 희망에 따라 짝 프로그래밍 여부를 선택할 수 있게 하고 중도에 포기할 수도 있게 해야 할 것이다. 둘째로, 짝 프로그래밍에 참여하는 학생들이 짝과의 역할 교대를 꺼리게 되는 요인들을 제거해 주어야 할 것이다. 실습과제의 양을 적절히 제한하여 학생들이 여유 있게 실습 시간을 즐길 수 있게 배려하고, 물리적으로 자리를 바꿔 앉기 쉽도록 실습실의 여유 공간을 확보하고 자리 배치에 신경을 써야 한다. 물론 감독자의 지속적인 격려와 분위기 조성도 필요하다. 셋째로 짝 배정에 세심한 주의를 기울여야 한다. 짝 프로그래밍에 참여하는 모든 학생들의 짝을 주기적으로 재배정해 주는 편이 좋다고 생각한다. 그렇지 않으면 학생들이 현재에 안주하는 경향이 있고 일부 재배정을 원하는 학생들도 짝을 변경하기 어려워지기 때문이다. 학생들은 수준이 비슷하거나 자기보다 조금 잘 하는 짝을 원하기 때문에 감독자는 테스트나 관찰을 통해 개개인의 수준과 성향을 파악하고 적절한 짝을 맺어주기 위한 노력을 해야 한다. 또, 잘 못하는 학생들끼리 짝을 맺은 학생들을 적극적으로 도와주어야 한다. 짝 프로그래밍을 통해 학생들이 짝과 서로 돕게 되면서 교수진에게 향하는 질문이 현저히 줄어듦으로 그렇게 할 여유가 생긴다.

과거 규율에 의존하는 교육에서 흥미와 재미 유발을 통한 자발적 학습으로 교육의 큰 흐름이 변하고 있다. 게임을 하는 듯한 느낌으로 학습을 하게 하는 방법[7] 등 다양한 시도가 있다. 이 글에서 제시한 짝 프로그래밍 경험이 교실에서의 프로그래밍 학습에서 학생들의 학습 동기를 높이는 데 도움이 되기를 바란다.

감사의 글

2017년도 강원대학교 대학회계 학술연구조성비로 연구하였음(관리번호-520170083).

참고문헌

[1] Kent Beck, *Extreme Programming Explained: Embrace Change,* Addison-Wesley Longman Publishing, 2000.

[2] Alistair Cockburn and Laurie Williams, "The Costs and Benefits of Pair Programming," *in Proceedings of the First International Conference on Extreme Programming and*

Flexible Processes in Software Engineering, Cagliari, Italy pp. 223-247, June 2000.

[3] Choong-Kyo Jeong, “Effects of Pair Programming in an Introductory Programming Course for College Students: Academic Performance and Student Satisfaction,” *Journal of The Korean Association of Information Education,* Vol. 21, No. 5, pp. 537-545, October 2017.

[4] Norsaremah Salleh, Emilia Mendes, and John C. Grundy. “Empirical Studies of Pair Programming for CS/SE Teaching in Higher Education: A Systematic Literature Review,” *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING,* Vol. 37, No. 4, pp. 509-525, July 2011.

[5] Class homepage [online] Available: http://dmrl.kangwon.ac.kr/lecture/1702/java/1702_java.html

[6] Tendai Dongo, April H. Reed, and Margaret O’Hara, "Exploring Pair Programming Benefits for MIS Majors," *Journal of Information Technology Education*, Vol. 15, pp. 223-239, 2016.

[7] Seungheon Kang, Ji-Yong Jeong, Sung-Jin Park, Sang-Kyun Kim, "Communication Effects of Gamification App: Focused on <Everybody’s Neighbor>," *Journal of Digital Contents Society*, Vol. 19, No. 7, July 2018.

정충교 (Choong-Kyo Jeong)

1995~현재 강원대학교 컴퓨터학부 교수

1989~1995 LG정보통신(주)

관심분야: 통신망성능분석, 시스템프로그래밍, 소프트웨어품질, 소프트웨어 교육

e-mail: ckjeong@kangwon.ac.kr