



AI 기반 짝 프로그래밍 기법을 활용한 프로그래밍 수업모델 설계*

Designing a Programming Course Model Using AI-Based Pair Programming Techniques

황혜민[†] · 이장원^{††}

Hyemin Hwang[†] · Jangwon Lee^{††}

요약

본 연구는 생성형 AI를 접목한 대학 프로그래밍 수업모델을 개발하고, 실제 이 모델에 기반한 수업을 평가하여 생성형 AI의 교육적 효과와 향후 활용 방안에 대하여 고찰하고자 하였다. 실험은 S대학교 사범대학 컴퓨터교육과 2024년 2학기 인공지능 교과목 강의 수강생 19명을 대상으로 진행되었으며, ChatGPT를 파트너로서 활용한 짝 프로그래밍 실습 강의를 진행하였다. 사전-사후 설문조사와 심층 인터뷰를 통해 설계한 짝 프로그래밍 기반 수업모델의 효과를 측정하였으며 생성형 AI를 활용한 짝 프로그래밍 강의를 학생들의 컴퓨팅 사고력, 학습 동기, 프로그래밍 자기효능감에 미치는 영향을 분석하였다. 실험 결과, 컴퓨팅 사고력의 알고리즘적 사고와 문제해결능력, 프로그래밍 자기효능감의 응용과 코드 리터러시 능력 영역에서 유의미한 향상이 확인되었다. 심층인터뷰에서는 AI와의 상호작용을 통한 능동적 학습이 가능했다는 긍정적 평가가 있었다. 이를 통해 생성형 AI를 프로그래밍 교육에도 효과적으로 활용할 수 있다는 가능성을 확인하였다.

주제어 생성형 AI, 짝 프로그래밍, 컴퓨팅 사고력, 프로그래밍 자기효능감, ChatGPT

ABSTRACT

This study designs a university programming course model that integrates Generative Artificial Intelligence (AI) with pair programming techniques. The model's educational effectiveness and future applications are evaluated through practical sessions conducted with 19 students from an artificial intelligence course at the College of Education, S University, using ChatGPT as a programming partner. Effectiveness is measured through pre- and post-surveys and in-depth interviews, assessing the impact on students' computational thinking, learning motivation, and programming self-efficacy. The results indicate significant improvements in algorithmic thinking and problem-solving within computational thinking, as well as in the application and code literacy aspects of programming self-efficacy. In-depth interviews highlight the potential for active learning through AI interaction, demonstrating generative AI's effective use in programming education.

Keywords Generative AI, Pair Programming, Computational Thinking, Programming Self-Efficacy, ChatGPT

†정회원 성균관대학교 대학원 실감미디어공학과 석사과정

††정회원 성균관대학교 사범대학 컴퓨터교육과 교수 (교신저자)

논문투고 2024년 11월 12일

심사완료 2024년 12월 30일

게재확정 2025년 01월 02일

발행일자 2025년 02월 12일

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 메타버스 융합대학원(IITP-2024-RS-2023-00254129)과 디지털 분야 해외 석학 유치 지원(RS-2024-00459638)의 연구 결과로 수행되었음.

1. 서론

코드 생성형 AI는 프롬프트를 입력하면 원하는 코드를 생성하고, 코드에 대한 해석도 제공해줄 수 있는 자연어 처리 기반 도구이다. 대표적인 코드 생성형 AI 도구로는 (GitHub Copilot, 2023)과 OpenAI의 (ChatGPT, 2023)가 있다[1-2]. 이러한 도구들의 사용은 이미 실무 개발 환경에서도 보편적으로 사용되고 있으며, 교육 현장에서도 사용이 증가하고 있어 효과적인 활용 방안에 대한 연구가 필요하다.

개발자들에게 요구되는 핵심 역량 또한 단순히 코드를 작성하는 능력에서 AI가 제시한 코드를 비판적으로 분석하고 이해하며, 주어진 문제 상황에 맞게 수정하여 활용할 수 있는 코드 리터러시 능력으로 바뀔 것으로 예상된다[3]. 따라서 프로그래밍 교육도 기존의 코드 작성 중심 교육에서 벗어나, 코드 해석 능력과 응용력을 향상시키는 방향으로 변화할 필요가 있다.

짝 프로그래밍(Pair Programming)은 이러한 맥락에서 주목할 만한 교육 방법론이다. 짝 프로그래밍은 두 명의 개발자가 협업하여 코드를 작성하는 방식으로, 한 명은 운전자(driver) 역할을 하여 실제 코드를 작성하고, 다른 한 명은 항해사(navigator) 역할을 맡아 코드를 검토하고 피드백을 제공한다[4]. 두 사람은 일정 시간 간격으로 역할을 교대하며 협업을 지속하는데, 이는 코드 품질 향상과 문제 해결 능력 증진에 효과적인 것으로 알려져 있다[5-6].

정중교(2017, 2018)의 연구에서 대학 프로그래밍 강의에 짝 프로그래밍을 도입한 결과, 짝과의 성격이나 실력 차이로 인해 참여도와 만족도가 떨어질 수 있다는 문제점이 발견되었다[7-8]. 이러한 문제점들이 짝 프로그래밍의 본래 취지인 상호 학습과 협력적 문제 해결을 저해하는 요소로 작용한 것으로 보인다.

이를 해결하기 위해 본 연구에서는 생성형 AI를 프로그래밍 학습 보조 도구로 활용하는 최근의 연구 동향(Qureshi, 2023; Lau&Guo, 2023; Yilmaz&Yilmaz, 2023)을 고려하여, ChatGPT를 짝 프로그래밍의 파트너로 활용하는 새로운 학습 방법을 제안하고자 한다[9-11]. 본 연구에서 제안하는 방식은 학생들의 성격이나 실력에 상관없이 평등한 조건에서 실습을 진행할 수 있다는 장점이 있다. 또한, 교수나 조교에게 하기 어려웠던 질문도 ChatGPT에게 자유롭게 할 수 있어 학생들의 능동적인 참여를 유도할 수 있을 것으로 기대된다.

짝 프로그래밍 수업은 학생들이 ChatGPT와 항해사와 운전자 역할을 일정 시간 간격으로 번갈아가며 수행하고, 서로가 작성한 코드를 검토하고 피드백을 제공하거나 새로운 아이디어를 제안하는 방식으로 설계하였다. S대학교 사범대학 인공지능 과목에서 이를 적용한 후, 학생들의 프로그래밍 능력을 컴퓨팅 사고력, 학습 동기, 프로그래밍 자기효능감 세 가지 항목으로 나누어 AI 짝 프로그래밍 수업 전후의 변화를 설문조사를 통해 분석하였다.

설문 문항은 컴퓨팅 사고력과 학습 동기 측정을 위해 Yilmaz&Yilmaz(2023)의 연구와 Law, Lee, & Yu(2010)의

연구에서 사용된 문항을 채택하였으며, 프로그래밍 자기효능감 측정을 위해서는 김갑수(2014)의 설문 문항을 보완하여 활용하였다[12-14]. 특히 프로그래밍 자기효능감 문항의 경우, 코드 리터러시 능력을 평가하기 위해 새로운 문항들을 추가하였다. 또한, 수강생 2명을 대상으로 심층 인터뷰를 진행하여 ChatGPT와의 짝 프로그래밍 경험이 학습자들의 프로그래밍 학습에 미치는 영향을 보다 구체적으로 파악하고자 하였다.

2. 관련 연구

2.1 생성형 AI를 활용한 프로그래밍 교육

생성형 AI는 인공지능 기술의 한 분야로서 주어진 데이터를 기반으로 새로운 콘텐츠를 생성할 수 있다. 자연어 처리(NLP)와 딥러닝 기술의 급격한 발전으로 인해, ChatGPT와 같은 생성형 AI 모델이 등장하며 교육, 의료, 금융 등 다양한 산업에서 활용되고 있다. 특히 교육 분야에서는 ChatGPT와 GitHub Copilot과 같은 AI 도구들이 학습을 위한 새로운 가능성을 제시하고 있다.

이러한 도구들의 교육적 활용에 대한 연구도 활발히 진행되고 있다. Lau&Guo(2023)는 프로그래밍 교육에서 AI 도구의 활용이 불가피한 추세임을 지적하며, 이러한 도구들이 학생들의 프로그래밍 과제 수행에 실질적인 도움을 줄 수 있다고 보고하였다. 특히 코드 생성과 설명 기능이 학습자의 이해도를 높이는 데 긍정적인 영향을 미치는 것으로 나타났다[10].

Denny et al.(2023)은 ChatGPT와 GitHub Copilot과 같은 생성형 AI 도구들이 초급 프로그래밍 교육에서 코드 생성 및 설명 기능을 통해 학습자들의 이해도를 높이는 데 기여할 수 있음을 강조하였다. 그러나 학습자가 도구에 지나치게 의존하거나, 생성된 코드의 정확성 및 보안성을 비판적으로 평가하는 능력이 저하될 위험성도 함께 언급하였다[15].

한편, Qureshi(2023)는 대학 컴퓨터 과학 교육과정에서 ChatGPT를 학습 및 평가 도구로 활용하는 실험을 진행하였다. ChatGPT 사용 그룹과 비사용 그룹을 비교한 결과, ChatGPT를 활용한 그룹이 프로그래밍 문제 해결에서 더 높은 점수를 얻었다. 그러나 제출된 코드의 불일치와 부정확성으로 인해 전반적인 학습 성과에는 부정적인 영향이 있음도 확인되었다[9]. 이는 생성형 AI를 교육에 활용할 때 적절한 가이드라인과 평가 기준이 필요함을 시사한다.

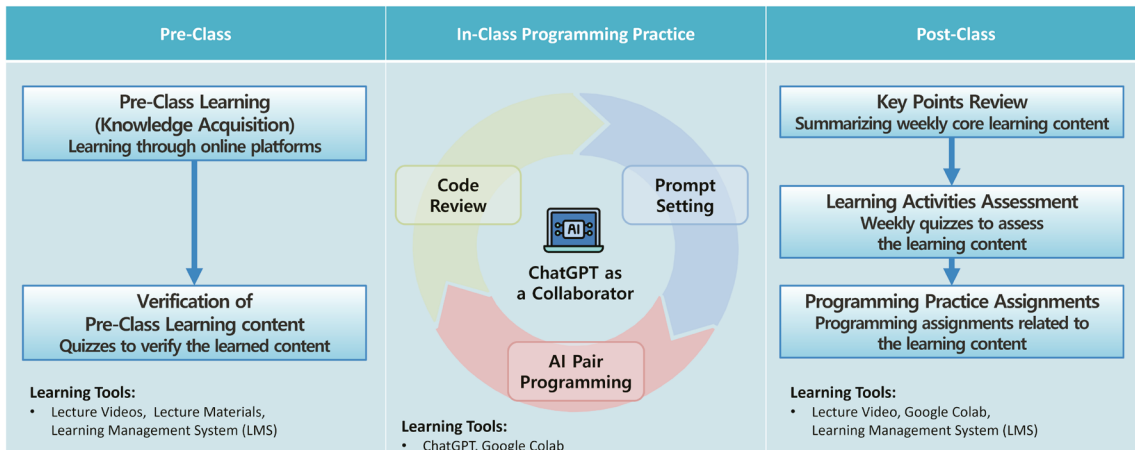


Figure 1. An Instructional Model for Pair Programming Utilizing Generative AI

2.2 짝 프로그래밍(Pair Programming)

짝 프로그래밍은 애자일 방법론의 대표적인 사례인 XP(eXtreme Programming)의 실천 방법 중 하나로, 두 명의 개발자가 한 컴퓨터 앞에 앉아 함께 작업하는 방식이다[4]. 이 때 한 명이 코드를 작성하는 운전자(driver) 역할을 하고, 다른 한 명이 코드를 검토하며 피드백을 주는 항해사(navigator) 역할을 한다. 두 사람은 약 15분 간격으로 역할을 바뀌가며 짝 프로그래밍을 진행한다.

Cockburn&Williams(2000)의 연구는 짝 프로그래밍이 소프트웨어 개발에서 설계 품질을 향상시키고, 오류를 줄이며 기술 향상과 팀 간 소통을 촉진시킨다고 밝혔다[5]. 또한, McDowell 외(2006)은 짝 프로그래밍이 개발자들의 자신감을 향상시키고 프로그래밍에 대한 긍정적인 태도를 형성하는 데 도움이 된다고 보고하였다[6]. Williams(2010)는 실제 기업 환경에서도 짝 프로그래밍을 도입한 팀이 더 높은 생산성과 코드 품질을 보였다고 보고하였다[16].

이러한 장점들로 인해 대학 수준의 프로그래밍 수업에서도 짝 프로그래밍이 널리 활용되고 있다. 그러나 정중교(2017, 2018)의 연구에 따르면, 짝 프로그래밍은 학습 성과 향상에 효과적이거나, 짝의 구성이 성과에 중요한 영향을 미치는 것으로 나타났다. 특히 15분 간격의 역할 교체에 대해서는 약 27%의 학생만이 긍정적으로 평가했으며, 대부분의 학생들은 필요할 때마다 서로 도움을 주는 방식을 선호했다. 이는 짝과의 실력 차이나 성격 차이로 인해 짝 프로그래밍의 본래 취지인 상호 학습과 협력적 문제 해결이 효과적으로 이루어지지 않을 수 있음을 시사한다. 따라서 학생들의 수준과 성격을 고려한 세밀한 짝 배정이 필요하다는 의견도 제시되었다[7-8].

이러한 한계를 극복하기 위해 본 연구에서는 짝을 생성형 AI로 대체하여, 학생들의 성격이나 실력에 상관없이 공평한 조건에서 짝 프로그래밍의 장점을 최대한 활용할 수 있도록 하였다. 특히 생성형 AI는 각 학생의 수준과 특성

에 맞춰 응답과 피드백을 유연하게 조정할 수 있어, 더욱 효과적인 학습이 가능하다는 장점이 있다.

2.3 프로그래밍 자기효능감

프로그래밍 자기효능감은 학습자가 프로그래밍 과제를 수행할 때 느끼는 자신감과 능력에 대한 개념을 의미한다. 이는 학습자가 주어진 문제를 해결할 수 있다는 믿음과 직접적인 연관이 있으며, 프로그래밍 능력 습득의 중요한 요소로 작용한다. Artino(2012)는 자기효능감이 학습 동기와 성취도에 직접적인 영향을 미친다고 보고하였으며[17], 이는 프로그래밍 교육에서도 동일하게 적용될 수 있다.

김갑수(2014)의 연구에서는 프로그래밍 자기효능감을 측정하기 위한 구체적인 평가 기준을 제시하였다[14]. 본 연구에서는 이 평가 기준을 바탕으로 실제 프로그래밍에 필수적인 다섯 가지 항목으로 재구성하였다. 여기에 더해, 생성형 AI 시대의 새로운 요구사항을 반영하여 코드 리터러시 능력을 평가하는 항목을 추가하였다.

코드 리터러시는 단순히 코드를 읽고 이해하는 능력을 넘어서는 개념이다. 특히 생성형 AI가 보편화된 현재의 프로그래밍 환경에서는, AI가 생성한 코드를 비판적으로 분석하고, 이를 자신의 목적에 맞게 수정하며, 더 나아가 이를 기존 코드와 효과적으로 통합할 수 있는 능력이 필수적이다. Lau&Guo(2023)의 연구에서도 강조했듯이, 이러한 코드 리터러시 능력은 AI 도구를 활용하는 프로그래밍 교육에서 핵심적인 평가 요소가 되어야 한다[10].

따라서 본 연구에서는 프로그래밍 자기효능감을 측정함에 있어, 기존의 프로그래밍 능력에 대한 자신감뿐만 아니라, AI가 생성한 코드를 이해하고 활용하는 능력에 대한 자신감까지 포함하여 종합적으로 평가하고자 하였다.

3. 연구 방법

생성형 AI를 활용한 짝 프로그래밍 수업은 Fig. 1에 제시된 수업 모형을 기반으로 진행되었다. 이 수업 모형은 정충교(2017, 2018)의 짝 프로그래밍 수업 모델을 참고하였으며, 기존의 학생 간 짝 프로그래밍 방식을 생성형 AI와의 짝 프로그래밍으로 수정하여 설계되었다. 설계된 모형은 컴퓨터교육 전문가 3인의 타당화 과정을 거쳐 구성의 적절성과 효과성을 검토받았다[7-8].

Table 1. Reliability

Area	Sub-area	Number of Items	Reliability
Computational Thinking	Creativity	5	0.62
	Algorithmic thinking	6	0.88
	Cooperativity	5	0.88
	Critical thinking	5	0.83
	Problem solving	4	0.73
Learning Motivation	Individual attitude and expectation	3	0.94
	Challenging goals	3	0.88
	Clear direction	2	0.89
	Reward and recognition	3	0.93
	Social pressure and competition	4	0.89
Programming Self-Efficacy	Understanding of Functionality	3	0.92
	Application	3	0.91
	Debugging	3	0.84
	Code Literacy	3	0.87
	Interest	3	0.94

본 수업은 컴퓨터 교육에서 자주 활용되는 플립드 러닝(Flipped Learning) 방식을 적용하였다. 프로그래밍 개념 및 이론 수업의 경우 온라인 사전학습(Pre-Class)으로 진행하고, 생성형 AI를 활용한 짝 프로그래밍 실습은 오프라인 실습 수업(In-Class) 시간에 실시하였다. 사후 학습(Post-Class) 시간에는 핵심 내용 요약 강의, 학습 평가를 위한 퀴즈, 그리고 실습 시간에 학습한 내용을 바탕으로 한 프로그래밍 과제 평가가 이루어졌다.

3.1 연구 대상

본 연구는 인공지능 강의를 수강하는 S대학교 사범대학 컴퓨터교육과 인공지능 교과목의 2024학년도 1학기 수강생 19명을 대상으로 생성형 AI를 활용한 짝 프로그래밍 실습의 효과를 분석하였다.

3.2 연구 도구

짝 프로그래밍 실습에 활용할 AI 도구로는 (Yetiştiren 외, 2023)의 연구를 참고하여 코드 생성 정확성이 가

장 높은 ChatGPT를 선정하였다[18]. 또한, 컴퓨팅 사고력, 학습 동기, 프로그래밍 자기효능감을 측정하기 위해 설문 도구를 사용하였다. 설문지의 구성은 Yilmaz&Yilmaz(2023)의 연구와 Law, Lee, & Yu(2010)의 연구에서 사용한 컴퓨팅 사고력과 학습 동기 문항을 일부 수정하여 활용하였다[12-13]. 프로그래밍 자기효능감 영역의 경우, 김갑수(2014)의 연구에서 제시한 척도를 본 연구의 목적에 맞게 재구성하였다[14].

초기 설문지는 컴퓨터교육 전문가 3인의 타당화 과정을 거쳤으며, 설문 문항의 적절성, 명확성, 그리고 연구 목적과의 부합성을 검토한 후 피드백을 반영하여 수정 및 보완하여 최종 설문지를 작성하였다. 설문지는 컴퓨팅 사고력(5점 척도), 학습 동기(6점 척도), 프로그래밍 자기효능감(5점 척도)으로 구성되었으며, 모든 문항은 리커트 척도를 사용하였다.

본 연구에서 사용한 설문 도구의 신뢰도 검증 결과는 Table 1에서 확인할 수 있으며, 대부분의 영역에서 Cronbach's α 값이 0.7 이상으로 나타나 타당성을 확보하였다. 다만, 창의력 설문지의 신뢰도가 0.62로 상대적으로 낮았으나, Taber(2018)의 연구에 따르면 탐색적 연구나 복합적 구인 측정에서는 0.60 이상의 신뢰도도 의미 있게 해석될 수 있다[19]. 이를 바탕으로 본 연구의 창의력 설문 문항 역시 적절한 신뢰도를 확보한 것으로 판단하여 그대로 사용하였다.

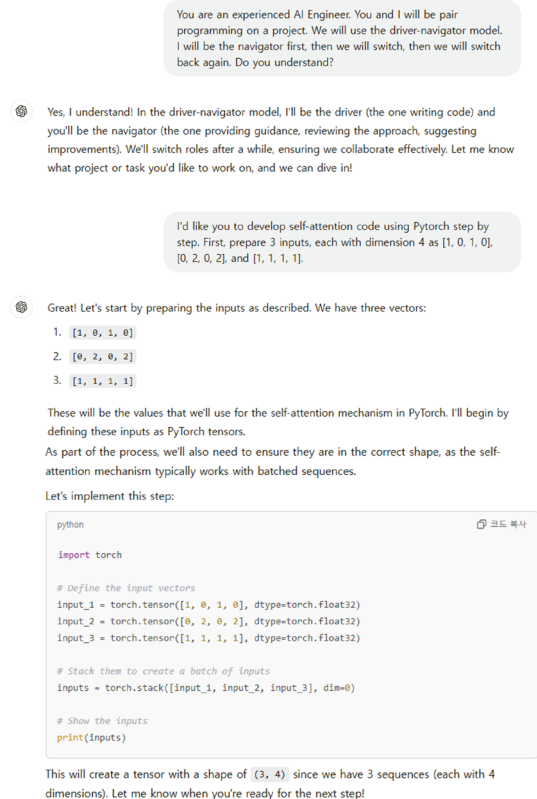


Figure 2. Example conversation of pair programming with ChatGPT: Setting up self-attention code using PyTorch

3.3 연구 설계 및 절차

본 연구는 16주 과정의 인공지능 교과목 중 생성형 AI를 활용한 짝 프로그래밍 실습을 3차시로 계획하였다. 실제 운영에서는 학사 일정으로 인해 2차시가 진행되었다. 연구 참여자들은 컴퓨터교육과 학생들로 대부분 구성되어 있어 프로그래밍 경험이 풍부하고 AI 활용에도 친숙하여, ChatGPT와의 상호작용에 필요한 프롬프트 작성에 어려움이 크지는 않았다. ChatGPT는 무료로 제공되는 GPT-3.5 버전을 사용하였다.

각 차시별로 학생들은 ChatGPT와 운전자와 항해사 역할을 번갈아가며 수행하였으며, 실습 과정에서 발생하는 문제점과 해결 방안을 공유하며 수업을 진행하였다. Fig. 2와 같이 ChatGPT에게 경험이 풍부한 AI 엔지니어 역할을 부여하고, 운전자-항해사 모델을 사용하여 프로그래밍을 수행할 것임을 명시하였다. 이후 일정 시간이 지나면 번갈아가며 역할을 전환하는 방식으로 짝 프로그래밍을 진행하였다.

본 연구에서 진행된 수업은 Python 언어와 딥러닝 프레임워크인 PyTorch를 활용하였으며, 학습 내용은 Attention 메커니즘과 Object Detection 기법을 중심으로 구성되었다. 학생들의 학습 효과를 분석하기 위해서는 수업 전후 설문조사를 통해 컴퓨팅 사고력, 학습 동기, 프로그래밍 자기효능감의 변화를 측정하였다. 또한 생성형 AI의 활용이 학생들의 학습 경험에 미친 영향을 심층적으로 이해하기 위해 수업에 참여한 학생 중 자발적으로 지원한 2명을 대상으로 심층 인터뷰를 실시하였다. 이 인터뷰는 학생 개개인의 학습 방식 변화, AI와의 상호작용에서 경험한 장단점, 그리고 학습 동기 및 자신감에 대한 구체적인 의견을 수집하는 데 중점을 두었다.

본 연구에서는 다음과 같은 연구가설을 통계적으로 검증하려고 한다.

연구가설 1: 생성형 AI를 활용한 짝 프로그래밍을 진행했을 때 실습 전후의 컴퓨팅 사고력에 통계적으로 유의미한 차이가 있을 것이다.

연구가설 2: 생성형 AI를 활용한 짝 프로그래밍을 진행했을 때 실습 전후의 학습 동기에 통계적으로 유의미한 차이가 있을 것이다.

연구가설 3: 생성형 AI를 활용한 짝 프로그래밍을 진행했을 때 실습 전후의 프로그래밍 자기효능감에 통계적으로 유의미한 차이가 있을 것이다.

4. 연구 결과

4.1 설문조사 결과

본 연구의 가설 검증을 위해 대응표본 t-검정을 실시하였다. 사전-사후 설문조사를 통해 컴퓨팅 사고력, 학습 동기, 그리고 프로그래밍 자기효능감에 미치는 영향을 종합적으로 분석하였다.

Table 2에 제시된 컴퓨팅 사고력에 대한 설문조사 결과, 생성형 AI를 활용한 짝 프로그래밍은 학습자의 컴퓨팅 사고력 향상에 전반적으로 긍정적인 영향을 미친 것으로 나타났다.

특히 알고리즘적 사고($t=-3.32, p<.01$)와 문제해결력($t=-3.58, p<.01$)에서 통계적으로 유의미한 향상이 관찰되었다. 창의력 영역에서도 유의미한 향상($t=-2.50, p<.05$)이 나타났는데, 흥미로운 점은 창의력 영역의 신뢰도 계수(Cronbach's $\alpha=0.62$)가 다른 영역에 비해 다소 낮게 측정되었다는 것이다. 이는 학생들이 AI와의 상호작용 과정에서 창의력을 각기 다른 방식으로 경험했음을 시사한다.

반면 협동성과 비판적 사고 영역에서는 평균값이 소폭 상승하였으나, 통계적으로 유의미한 차이는 나타나지 않았다. 이러한 결과는 본 연구의 2차시라는 비교적 짧은 실험 기간의 한계를 반영하는 것으로 보이며, 향후 더 장기적인 관찰을 통해 이들 영역에서의 변화를 면밀히 살펴볼 필요가 있다.

Table 2. Survey Results on the Effects of Pair Programming with Generative AI on Computational Thinking

	Source	M	SD	t	P
Creativity	pre	4.13	0.50	-2.50*	.022
	post	4.31	0.61		
Algorithmic thinking	pre	3.49	0.85	-3.32**	.004
	post	3.94	0.64		
Cooperativity	pre	3.82	0.95	-1.00	.331
	post	3.96	0.93		
Critical thinking	pre	4.00	0.66	-0.48	.634
	post	4.05	0.70		
Problem solving	pre	3.95	0.64	-3.58**	.002
	post	4.25	0.58		

(** $p<0.01$, * $p<0.05$)

Table 3. Survey Results on the Effects of Pair Programming with Generative AI on Learning Motivation

	Source	M	SD	t	P
Individual attitude and expectation	pre	5.21	0.96	-0.11	.92
	post	5.23	0.90		
Challenging goals	pre	4.32	1.21	-0.77	.45
	post	4.47	1.38		
Clear direction	pre	5.16	0.90	-0.48	.64
	post	5.24	0.73		
Reward and recognition	pre	5.35	0.81	0.19	.85
	post	5.32	0.81		
Social pressure and competition	pre	3.93	1.40	-0.99	.34
	post	4.08	1.33		

Table 3에 제시된 학습 동기에 대한 설문조사의 경우, 모든 하위 요인에서 사전-사후 검사 간 통계적으로 유의미한 차이는 발견되지 않았다. 도전적 목표 설정(사

전 $M=4.32$, 사후 $M=4.47$)과 사회적 압박과 경쟁(사전 $M=3.93$, 사후 $M=4.08$) 영역에서는 소폭의 평균 증가가 관찰되었다. 개인의 태도 및 기대, 명확한 방향성, 보상과 인정 항목은 이미 사전 검사에서 5점 이상의 높은 점수를 보였으며, 사후 검사에서도 유지되었다. 이는 학생들이 실험 전부터 이미 높은 수준의 학습 동기를 가지고 있었을 가능성을 시사한다. 짧은 실험 기간을 고려했을 때, 학습 동기에 미치는 영향을 정확하게 파악하기 위해서는 보다 장기적인 연구가 필요할 것으로 보인다.

Table 4에서 제시된 프로그래밍 자기효능감 측면에서는 특히 응용 능력에서 가장 큰 향상을 보였으며($t=-3.76$, $p<.01$), 코드 리터러시 영역에서도 유의미한 증가가 관찰되었다($t=-2.75$, $p<.05$). 이는 생성형 AI를 활용한 짝 프로그래밍이 학습자들의 프로그래밍 적용 능력과 코드 이해 능력 향상에 효과적이었음을 시사한다. 응용 영역의 경우 사전 평균 3.21에서 사후 평균 3.81로 가장 큰 폭의 향상을 보였으며, 이는 AI와의 상호작용을 통한 실제적인 프로그래밍 경험이 학습자들의 응용 능력 향상에 긍정적인 영향을 미쳤음을 나타낸다.

Table 4. Survey Results on the Effects of Pair Programming with Generative AI on Programming Self-Efficacy

	Source	M	SD	t	P
Understanding of Functionality	pre	3.63	1.23	-1.57	.134
	post	3.91	0.88		
Application	pre	3.21	1.20	-3.76**	.001
	post	3.81	1.03		
Debugging	pre	3.49	1.03	-1.48	.157
	post	3.70	0.99		
Code Literacy	pre	3.42	1.05	-2.75*	.013
	post	3.86	0.95		
Interest	pre	3.23	1.25	-1.73	.101
	post	3.58	1.06		

(** $p<.01$, * $p<.05$)

이러한 결과들을 통해 본 연구의 가설 1(컴퓨팅 사고력 향상)과 가설 3(프로그래밍 자기효능감 향상)은 지지되었으나, 가설 2(학습동기 향상)는 기각되었음을 알 수 있다. 특히 알고리즘적 사고, 문제해결력, 응용 능력, 코드 리터러시 영역에서의 유의미한 향상은 생성형 AI를 활용한 짝 프로그래밍의 교육적 효과를 실증적으로 보여준다.

비록 모든 영역에서 통계적으로 유의미한 향상이 나타나지는 않았으나, 전반적인 평균값의 상승은 본 연구에서 제시한 교육 방식이 프로그래밍 역량 향상에 긍정적인 영향을 미칠 가능성을 시사한다. 다만, 학습 동기 영역에서는 유의미한 차이가 나타나지 않았으며, 이는 짧은 실험 기간과 참여 학생들의 이미 높은 초기 수준이 주요 요인으로 작용했을 가능성이 있다. 또한, 생성형 AI와의 상호작용이 인간 동료와의 협업과는 다른 특성을 가지므로, 협동성 부

분에서 이를 보완할 수 있는 추가적인 교육적 방안을 모색할 필요가 있다.

4.2 심층 인터뷰

수업이 끝난 후, 생성형 AI를 활용한 짝 프로그래밍의 전반적인 경험에 대해 분석하기 위해 2명의 학생들(이하 S1, S2)과 심층 인터뷰를 진행하였다.

먼저, 학생들에게 생성형 AI의 사용으로 인해 생긴 학습 방법의 변화와 장점에 대해 물어보았을 때, 다음과 같은 답변을 얻을 수 있었다.

S1: “원래는 스택오버플로우나 구글로 찾아보는 방법밖에 없었는데, 이제 생성형 AI에게 직접 물어보고 오류를 수정하는 등 코드에 대한 피드백을 받는다.”

S2: “이론 수업과 실습 사이에 있던 간극을 채우는 과정에 시간이 많이 소요되었는데, AI를 사용했을 때 그러한 시간을 절약할 수 있었다.”

수업 참여도 측면에서도 긍정적인 변화가 관찰되었는데, 학생들은 다음과 같이 답변하였다.

S1: “예전에는 교수님이 보여주시는 코드를 그냥 보기만 하거나, 직접 코딩할 때 막막함을 많이 느꼈는데, AI와 함께하니 궁금한 점을 자유롭게 물어볼 수 있어서 좋았다.”

S2: “AI가 개념에 대한 코드 예시를 바로 보여주고 직관적으로 설명해줘서, 새로운 내용을 배우는 데 있어서 좀 더 재밌어졌다.”

자신감 측면에서는 다음과 같은 답변이 있었다.

S1: “실제로도 자신감이 향상되었다는 게 느껴졌지만, 코드 작성을 AI가 전부 해준다는 것이 아니라 보조 도구로써 사용한다고 생각했을 때 좀 더 자신감이 생겼다.”

S2: “처음 읽었을 때 모르는 문제가 좀 많았는데, 생성형 AI를 사용하면서 한 번 풀었던 문제에 대한 이해도가 올라 실습 과정에서 자신감이 생겼다.”

한편, 생성형 AI 활용 과정에서 나타나는 한계점에 대해서는 다음과 같은 답변이 있었다.

S1: “AI가 때로는 전체 코드의 맥락을 모르다 보니 완벽한 도움을 주기 어려웠고, 무료 버전을 사용하다 보니 횟수 제한으로 인한 어려움이 있었다.”

S2: “프롬프트 엔지니어링이 중요하다고 생각되어서 수업에서 생성형 AI를 사용한다면 구체적인 작성 방법이나 설정에 대해 더 자세히 알려줘야 모든 학생들이 동일하게 결과물을 얻을 수 있을 것 같다.”

전반적으로 학생들은 AI와의 짝 프로그래밍을 긍정적으로 평가했다. 사람과 하기 어려운 활동을 AI와 함께 할 수 있어 좋았다는 장점이 있지만, 학습 도구로 활용하려면 프롬프트 작성 능력이 중요할 것 같다는 의견도 공통적으로 제시되었다.

다만, 학생 2명의 인터뷰 결과는 연구 대상 전체를 대표한다고 보기에는 한계가 있다. 이는 소규모 표본의 특성에 기인한 것이며, 본 연구의 제한점 중 하나로 고려된다. 이를 보완하기 위해 후속 연구에서는 더 다양한 배경을 가진

학생들을 포함한 심층 인터뷰와 정량적 설문 데이터를 병행하여 분석할 필요가 있다.

추가적으로, 인터뷰에서 언급된 프롬프트 작성 능력은 참여도와 만족도에 직접적인 영향을 미칠 가능성이 높다. 따라서, 후속 연구에서는 프롬프트 작성 능력을 향상시키기 위한 교육을 실시하고, 이를 통해 생성형 AI 활용이 학습자 경험에 미치는 영향을 보다 체계적으로 분석할 필요가 있다.

5. 결론

본 연구는 ChatGPT를 짝 프로그래밍의 파트너로 활용하여 학생들의 컴퓨팅 사고력, 학습 동기, 프로그래밍 자기 효능감의 변화를 분석하였다. 연구 결과, 생성형 AI를 활용한 짝 프로그래밍이 특히 알고리즘 사고력과 문제해결 능력, 그리고 프로그래밍 응용 능력과 코드 리터러시 향상에 효과적인 것으로 나타났다. 심층 인터뷰에서도 AI와의 상호작용을 통한 능동적 학습이 가능했다는 긍정적 평가가 있었다.

그러나 본 연구는 몇 가지 한계점을 가진다. 19명이라는 비교적 적은 표본으로 인해 통계적 분석 결과를 일반화하기 어렵다는 제한이 있다. 또한 연구 참여자들 다수가 컴퓨터교육과 학생들로서 인공지능에 대해 호의적이고 관심이 높은 집단이었으며, 이미 프로그래밍과 인공지능 활용 경험이 있어 ChatGPT와의 상호작용에 어려움이 적었다는 특수성이 있다. 2차시라는 비교적 짧은 기간 동안 진행되어 변화를 충분히 관찰하기 어려웠으며, 참여자들이 사전에 이미 일정 수준 이상의 프로그래밍 역량을 보유하고 있어 컴퓨팅 사고력의 특정 영역에서의 향상이 제한되었을 가능성도 있다.

이러한 한계점을 보완하기 위해 향후 연구에서는 더 많은 표본을 대상으로 한 연구가 필요하며, 프로그래밍과 인공지능 활용 경험이 상대적으로 적은 학습자들을 포함하여 연구를 진행할 필요가 있다. 이를 통해 생성형 AI를 활용한 짝 프로그래밍의 효과를 더욱 일반화하여 검증할 수 있을 것이다. 또한 보다 장기적인 관찰을 통해 협동성, 비판적 사고, 학습 동기 영역에서의 변화를 면밀히 분석할 필요가 있다. 아울러 생성형 AI 활용에 대한 구체적인 가이드라인 개발이 필요하다. 본 연구의 참여자들은 AI 활용에 능숙했으나, 일반적인 교육 현장에서는 효과적인 프롬프트 작성법과 AI 활용 전략에 대한 체계적인 지원이 필요할 것으로 보인다.

본 연구는 생성형 AI를 프로그래밍 교육에 활용하는 구체적인 방안을 제시하고 그 효과성을 실증적으로 분석했다는 점에서 의의가 있다. 이러한 연구 결과를 바탕으로 다양한 학습자 특성과 교육 맥락을 고려한 생성형 AI 활용 교육 방법이 개발되기를 기대한다.

참고문헌

- [1] Dohmke, T. (2023, February 14). *GitHub Copilot for business is now available*. <https://github.blog/2023-02-14-github-copilot-for-business-is-now-available/>
- [2] OpenAI. (2023, March 14). *GPT-4 research*. <https://openai.com/research/gpt-4>
- [3] Brennan, R. W., & McDermott, B. (2023). Coding Literacy in the Age of Generative AI. *Service oriented, holonic and multi-agent manufacturing systems for industry of the future*, Springer, 1136, 445-456. https://doi.org/10.1007/978-3-031-53445-4_37
- [4] Cockburn, A., & Highsmith, J. (2001). Agile software development, the people factor. *Computer*, 34(11), 131-133. <https://doi.org/10.1109/2.963450>
- [5] Cockburn, A., & Williams, L. (2000). The costs and benefits of pair programming. In *Extreme Programming Examined*, 223-243.
- [6] McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM*, 49(8), 90-95. <https://doi.org/10.1145/1145287.1145293>
- [7] Jeong, C.-K. (2017, December 1). Management of Pair Programming in Programming Classes: PairMatching. *Proceedings of KIIT Conference*, Daejeon. 125-126.
- [8] Jeong, C.-K. (2018). Pair programming in programming lab: The effects, limits, and guidelines based on the student receptivity. *Journal of Digital Contents Society*, 19(9), 1663-1669. <https://doi.org/10.9728/dcs.2018.19.9.1663>
- [9] Qureshi, B. (2023). Exploring the use of ChatGPT as a tool for learning and assessment in undergraduate computer science curriculum: Opportunities and challenges. *arXiv*. <https://doi.org/10.48550/arXiv.2304.11214>
- [10] Lau, S., & Guo, P. (2023). From “Ban It Till We Understand It” to “Resistance is Futile”: How University Programming Instructors Plan to Adapt as More Students Use AI Code Generation and Explanation Tools such as ChatGPT and GitHub Copilot. *Proceedings of the 2023 ACM Conference on International Computing Education Research*, New York, NY, USA, 1(1), 106-121. <https://doi.org/10.1145/3568813.3600138>
- [11] Yilmaz, R., & Yilmaz, F. G. K. (2023). Augmented intelligence in programming learning: Examining student views on the use of ChatGPT for programming learning. *Computers in Human Behavior: Artificial Humans*, 1(2), 100005. <https://doi.org/10.1016/j.chbah.2023.100005>
- [12] Yilmaz, R., & Yilmaz, F. G. K. (2023). The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation. *Computers and Education: Artificial Intelligence*, 4, 100147. <https://doi.org/10.1016/j.caeai.2023.100147>

- [13] Law, K. M. Y., Lee, V. C. S., & Yu, Y. T. (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education*, 55(1), 218-228. <https://doi.org/10.1016/j.compedu.2010.01.007>
- [14] Kim, K. S. (2014). Measuring and applying the self-efficacy in computer programming education. *Journal of the Korean Association of Information Education*, 18(1), 111-120. <https://doi.org/10.14352/jkaie.2014.18.1.111>
- [15] Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J., ... & Sarsa, S. (2024). Computing education in the era of generative AI. *Communications of the ACM*, 67(2), 56-67. <https://doi.org/10.1145/3624720>
- [16] Williams, L. A. (2010). Pair Programming. *Encyclopedia of software engineering*, 2.
- [17] Artino, A. R. (2012). Academic self-efficacy: from educational theory to instructional practice. *Perspectives on medical education*, 1, 76-85. <https://doi.org/10.1007/s40037-012-0012-5>
- [18] Yetiştiren, B., Özsoy, I., Ayerdem, M., & Tüzün, E. (2023). Evaluating the code quality of ai-assisted code generation tools: An empirical study on github copilot, amazon codewhisperer, and chatgpt. *arXiv preprint arXiv:2304.10778*. <https://doi.org/10.48550/arXiv.2304.10778>
- [19] Taber, K. S. (2018). The use of Cronbach's alpha when developing and reporting research instruments in science education. *Research in science education*, 48, 1273-1296. <https://doi.org/10.1007/s11165-016-9602-2>



황혜민

- 2023년 한국항공대학교 전자 및 항공전자공학과(공학사)
- 2023 ~ 현재 성균관대학교 실감미디어공학과 석사과정
- 2023년 ~ 현재 성균관대학교 실감미디어공학과 석사과정

✚ 관심분야 : Computer Vision, Deep Learning
 ✉ hyemin00@skku.edu



이장원

- 2006년 성균관대학교 정보통신공학부 (공학사)
- 2008년 성균관대학교 전자전기컴퓨터공학과 (공학석사)
- 2018년 Indiana University Intelligent and Interactive Systems (이학박사)
- 2018년 ~ 2020년 : ObjectVideo Labs at Alarm.com Research Scientist
- 2020년 ~ 2023년 : 한국항공대학교 항공전자정보공학부 교수
- 2023년 ~ 현재 : 성균관대학교 컴퓨터교육과 및 실감미디어공학과 교수

✚ 관심분야 : Computer Vision, Human-Robot-Interaction (HRI)
 ✉ leejang@skku.edu

부 록

1. 컴퓨팅 사고력 측정 설문 문항

영역	문항 내용
개인의 태도 및 기대치	프로그램 작성을 잘 수행하면 좋은 성적을 받을 수 있다는 것을 알면 더 잘 수행할 것이다.
	높은 성적을 기대하면 더 잘해야겠다는 동기가 생긴다.
	좋은 프로그램에 대한 기대가 커질수록 더 열심히 노력할 것이다.
도전적인 목표	한 번도 접해보지 못한 문제를 해결하기 위해 프로그램을 작성해야 할 때, 그 과정을 통해 새로운 학습 경험을 쌓고 완성하고자 하는 동기를 부여 받게 될 것이다.
	프로그래밍 연습이 어려워 보이면 더 잘하고 싶은 동기가 생긴다.
	도전적인 프로그래밍 연습은 나에게 더 열심히 하도록 동기를 부여한다.
명확한 방향성	프로그래밍 실습 문제의 목표가 명확하면 더 나은 성과를 내기 위한 동기가 부여된다.
	프로그래밍 실습 문제에서 무엇을 달성할 것인지 구체적으로 알면 더 잘 수행할 수 있다.
보상 및 인정	나의 좋은 성과를 다른 사람(같은 반 친구나 교수자)이 긍정적으로 평가할 때 나의 성과는 더욱 향상될 것이다.
	적절한 보상(예: 보너스 점수 및 높은 점수)이 주어지면 프로그래밍 연습에서 더 잘할 수 있는 동기가 부여될 것이다.
	교수자의 격려와 좋은 코멘트가 학습에 대한 동기를 부여한다.
사회적 압박과 경쟁	반 친구들과의 경쟁으로 인해 더 나은 성과를 내기 위해 노력한다.
	교수자의 압박은 내가 더 잘 배우고 더 열심히 일하도록 한다.
	반 친구들이 더 잘하면 따라잡기 위해 더 잘 배우려는 동기가 생긴다.
	반 친구들의 압박이 나를 더 잘 배우도록 만든다.

2. 학습 동기 측정 설문 문항

영역	문항 내용
창의력	나는 대부분 자신의 의사 결정에 확신이 있는 사람들을 좋아한다.
	나는 새로운 상황에 직면했을 때 발생할 수 있는 문제들을 해결할 수 있다는 믿음이 있다.
	나는 계획을 세울 때 그 계획을 적용하면 문제를 해결할 수 있다고 믿는다.
	꿈을 꾸는 것은 나의 가장 중요한 프로젝트를 실현 시키는 원동력이다.
알고리즘 사고력	나는 어떤 문제에 부딪히면 다른 주제로 넘어가기 전에 잠시 멈추고 그 문제에 대해 생각한다.
	나는 문제의 해결을 위한 공평한 기준을 즉시 수립할 수 있다.
	나는 수학적 과정에 특별한 관심이 있다고 생각한다.
	나는 수학적 기호와 개념의 도움으로 만들어진 지침을 더 잘 배운다고 생각한다.
흥미	나는 다른 사람이 작성한 코드를 보고 어떤 프로그램인지 잘 이해할 수 있다
	나는 다른 사람의 코드를 수정하여 나에게 필요한 용도로 사용할 수 있다.
	나는 다른 사람이 만든 코드 조각을 나의 프로그램에 통합 시킬 수 있다.
	나는 컴퓨터 프로그래밍 수업을 들을 때 수업 시간에 배우지 않은 내용도 스스로 공부하고 싶어한다.
활용	나는 프로그래밍 언어를 학습할 때 다른 학생들보다 더 좋은 소프트웨어를 만들고 싶어한다.
	나는 컴퓨터 프로그래밍 언어를 학습할 때 시간 가는 줄 모르고 학습한다.

협동성	나는 그룹 친구들과 함께 협동 학습을 하는 것을 좋아한다.
	협동 학습에서는 그룹으로 활동하기 때문에 더 성공적인 결과를 얻을 수 있다고 생각한다.
	나는 협동 학습에서 친구들과 함께 조별 프로젝트와 관련된 문제를 해결하는 것을 좋아한다.
	협동 학습에서는 더 많은 아이디어가 떠오른다.
비판적 사고	나는 프로그래밍 작업을 여러 사람들을 위한 하위 작업으로 나눌 수 있다는 것을 알고 있다.
	나는 복잡한 문제를 해결하기 위해 정기적으로 계획을 세우는데 능숙하다.
	나는 복잡한 문제를 해결하려고 노력하는 것이 즐겁다.
	나는 처음 보는(도전적인) 내용을 기꺼이 배운다.
문제 해결능력	나는 선택지를 비교하고 결정에 도달할 때 체계적인 방법을 활용한다.
	머릿속에 있는 문제의 해결책을 증명할 수 있다.
	문제 풀이에서 X, Y와 같은 변수를 어디에 어떻게 사용해야 하는지 알 수 있다.
	내가 계획한 해결 방법을 하나씩, 점진적으로 적용할 수 있다.
문제와 관련하여 가능한 해결 방법을 생각하면서 많은 선택지를 만들어낼 수 있다.	

3. 프로그래밍 자기효능감 측정 설문 문항

영역	문항 내용
기능 이해	나는 파이썬 프로그램 언어의 기본적인 데이터 유형, 변수와 상수 사용, 제어문, 입출력문을 이용하여 간단한 프로그램을 만들 수 있다.
	나는 복잡한 제어문, 배열, 함수, 구조체, 다양한 파일 처리를 이해할 수 있다.
	파이썬 라이브러리에서 제공되는 다양한 내장 함수를 사용할 수 있다.
활용	나는 컴퓨터 프로그래밍 언어를 학습할 때에 기능을 확장하고 싶은 마음이 든다.
	나는 프로그래밍 언어의 기본적인 데이터 유형, 변수와 상수 사용, 간단한 제어문, 간단한 입출력문을 이용하여 수학, 과학, 사회에서의 각 문제들을 해결하는 프로그램을 만들 수 있다.
	나는 컴퓨터 프로그래밍 언어를 학습할 때에 여러 가지 데이터로 이용하여 시험하는 것을 즐긴다.
디버깅	프로그램을 테스트하는 동안 오류의 원인을 찾을 수 있다.
	프로그램을 테스트하는 동안 오류를 수정할 수 있다.
	디버깅 프로세스를 통해 프로그래밍에 대해 더 많이 배울 수 있다.
코드 리터러시 능력	나는 다른 사람이 작성한 코드를 보고 어떤 프로그램인지 잘 이해할 수 있다
	나는 다른 사람의 코드를 수정하여 나에게 필요한 용도로 사용할 수 있다.
	나는 다른 사람이 만든 코드 조각을 나의 프로그램에 통합 시킬 수 있다.
흥미	나는 컴퓨터 프로그래밍 수업을 들을 때 수업 시간에 배우지 않은 내용도 스스로 공부하고 싶어한다.
	나는 프로그래밍 언어를 학습할 때 다른 학생들보다 더 좋은 소프트웨어를 만들고 싶어한다.
	나는 컴퓨터 프로그래밍 언어를 학습할 때 시간 가는 줄 모르고 학습한다.

〈표 1〉 신뢰도

영역	하위영역	문항 수	신뢰도
컴퓨팅 사고력	창의력	5	0.62
	알고리즘 사고력	6	0.88
	협동성	5	0.88
	비판적 사고	5	0.83
	문제해결능력	4	0.73
학습 동기	개인의 태도 및 기대치	3	0.94
	도전적인 목표	3	0.88
	명확한 방향성	2	0.89
	보상 및 인정	3	0.93
	사회적 압박과 경쟁	4	0.89
프로그래밍 자기효능감	기능 이해	3	0.92
	활용	3	0.91
	디버깅	3	0.84
	코드 리터러시 능력	3	0.87
	흥미	3	0.94

〈표 2〉 생성형 AI를 활용한 짝 프로그래밍이 컴퓨팅 사고력에 미치는 영향에 대한 설문조사 결과

	Source	M	SD	t	P
창의력	pre	4.13	0.50	-2.50*	.022
	post	4.31	0.61		
알고리즘 사고력	pre	3.49	0.85	-3.32**	.004
	post	3.94	0.64		
협동성	pre	3.82	0.95	-1.00	.331
	post	3.96	0.93		
비판적 사고	pre	4.00	0.66	-0.48	.634
	post	4.05	0.70		
문제해결능력	pre	3.95	0.64	-3.58**	.002
	post	4.25	0.58		

(**p<0.01, *p<0.05)

〈표 3〉 생성형 AI를 활용한 짝 프로그래밍이 학습 동기에 미치는 영향에 대한 설문조사 결과

	Source	M	SD	t	P
개인의 태도 및 기대치	pre	5.21	0.96	-0.11	.92
	post	5.23	0.90		
도전적인 목표	pre	4.32	1.21	-0.77	.45
	post	4.47	1.38		
명확한 방향성	pre	5.16	0.90	-0.48	.64
	post	5.24	0.73		
보상 및 인정	pre	5.35	0.81	0.19	.85
	post	5.32	0.81		
사회적 압박과 경쟁	pre	3.93	1.40	-0.99	.34
	post	4.08	1.33		

〈표 4〉 생성형 AI를 활용한 짝 프로그래밍이 프로그래밍 자기효능감에 미치는 영향에 대한 설문조사 결과

	Source	M	SD	t	P
기능 이해	pre	3.63	1.23	-1.57	.134
	post	3.91	0.88		
활용	pre	3.21	1.20	-3.76**	.001
	post	3.81	1.03		
디버깅	pre	3.49	1.03	-1.48	.157
	post	3.70	0.99		
코드 리터러시 능력	pre	3.42	1.05	-2.75*	.013
	post	3.86	0.95		
흥미	pre	3.23	1.25	-1.73	.101
	post	3.58	1.06		

(**p<0.01, *p<0.05)