컴퓨터교육학회 논문지 2026년 제29권 제8호
https://doi.org/10.32431/kace.2026.29.8.012

C 프로그래밍 학습 지원을 위한 AI 에이전트 프로토타입 개발

Development of an AI Agent Prototype for Supporting Learning C Programming

김민지† · 이동국††
Minji Kim† · Dongkuk Lee††

요약

프로그래밍 학습에서 적절한 학습 지원이 이루어지지 않으면 학습 지속이 어려워지고, 반복적인 오류 경험으로 인해 학습 동기가 저하될 수 있다. 이에 본 연구는 C 프로그래밍 학습을 지원하기 위한 AI 에이전트 프로토타입을 개발하였다. 이를 위해 설계·개발 연구 방법을 적용하였다. 문헌 및 사례 분석을 통해 6가지 설계원리를 도출하고 이를 반영한 AI 에이전트 프로토타입을 개발하였다. 이후 전문가 타당화와 사용성 평가를 실시하여 프로토타입의 타당성과 활용 가능성을 확인하고, 그 결과를 바탕으로 최종 프로토타입을 개발하였다. 본 연구는 AI 에이전트가 프로그래밍 학습에서 단순한 코드 생성 도구를 넘어 학습자의 문제해결 과정과 학습 지속을 지원하는 교수적 도구로 활용될 가능성을 제시하였다.

주제어 프로그래밍 학습, AI 에이전트, AI활용교육, 스캐폴딩, 설계개발연구방법

ABSTRACT

In learning programming, insufficient support can make it difficult for learners to persist and may reduce learning motivation due to repeated errors. Therefore, this study developed an AI agent prototype to support to learn C programming. To this end, a design-and-development research method was applied. Six design principles were derived through a review of prior studies and case analyses, and an AI agent prototype reflecting these principles was developed. Expert validation and usability evaluation were conducted to assess the prototype's validity and applicability. Based on the results, the final prototype was developed. This study demonstrates the potential of AI agents to function not only as code-generation tools but as valuable pedagogical partners that actively foster problem-solving skills and sustained learner engagement in programming education, ultimately contributing to improved learning experiences.

Keywords Learning Programming, AI Agent, AI in Education, Scaffolding, Design and Development Research

†정회원 을지대학교 교양학부 조교수
††종신회원 경북대학교 정보·컴퓨터교육과 조교수 (교신저자)

논문투고 2026년 06월 05일
심사완료 2026년 06월 29일
게재확정 2026년 07월 16일
발행일자 2026년 08월 27일

1. 서론

프로그래밍(programming) 학습은 컴퓨터 프로그램을 작성하고 실행하는 과정에서 문제를 분석하고, 알고리즘을 설계하며, 프로그래밍 언어를 활용해 해결책을 구현하는 지식과 기능을 습득하는 과정을 의미한다. 프로그래밍 학습을 통해 논리적 사고력, 문제해결력, 창의력을 기를 수 있기 때문에[1], 우리나라 2022 개정 교육과정에서도 정보 교육 시수를 확대하고 프로그래밍 학습을 강조하고 있다[2]. 특히 C 언어는 컴퓨터 구조와 메모리 개념에 대한 이해를 요구하는 텍스트 기반 프로그래밍 언어로, 초보 학습자에게 높은 인지 부하를 유발할 수 있다. 초보 학습자들은 프로그래밍 학습 과정에서 알고리즘 설계, 문법 오류 수정, 디버깅 등 다양한 어려움을 겪으며, 이 과정에서 적절한 피드백이 신속하게 제공되지 않으면 학습 동기가 저하되고 학습을 포기하는 경향이 나타날 수 있다[3].

이러한 문제를 해결하기 위하여 스캐폴딩(Scaffolding) 기반 학습 전략을 활용할 수 있다. 스캐폴딩은 학습자가 스스로 문제를 해결할 수 있도록 단계적 도움을 제공하는 교수 전략으로, 학습자의 인지 부하를 줄이고 학습 태도에 긍정적인 영향을 준다[1]. 프로그래밍 학습에서 즉각적이고 개별화된 스캐폴딩을 제공함으로써, 학습자가 스스로 오류를 분석하고 해결 전략을 탐색하도록 지원할 수 있다. 따라서 적시적 스캐폴딩 전략은 컴퓨팅 사고력과 알고리즘적 사고력을 기르는 데 도움을 주며, 오류 상황에서도 문제 해결을 지속하도록 유도하는 핵심 기제로 작용할 수 있다[3]. 그러나 전통적인 교실 환경에서 한 명의 교수자가 다수의 개별 학습자 각각의 오류를 즉각적으로 진단하고 맞춤형 스캐폴딩을 제공하기에는 한계가 있다[4]. 이러한 한계를 보완하기 위하여 최근 생성형 AI를 프로그래밍 학습 지원에 활용하려는 시도가 이루어지고 있다[5, 6].

최근 대규모 언어 모델(Large Language Model, LLM)을 기반으로 하는 인공지능(Artificial Intelligence, AI) 에이전트(Agent)의 발전은 프로그래밍 학습 지원의 새로운 가능성을 제시한다. AI 에이전트는 스스로 목표를 이해하고, 계획을 세우며, 적절한 도구를 사용하여 필요한 작업을 수행하는 자동화된 AI 시스템이다. 프로그래밍 학습 지원도구로서 AI 에이전트는 학습자의 코드 작성 상태, 오류 유형, 학습 진행 상황을 실시간으로 확인하고, 학습 목표 달성을 위한 피드백을 제공할 수 있다[4]. 이에 프로그래밍 학습을 효과적으로 지원하기 위하여 AI 에이전트를 개발하는 시도가 이루어지고 있다[2, 4]. 선행 연구들은 LLM 기반 AI 에이전트가 프로그래밍 학습 과정에서 최적화된 개인 맞춤형 피드백을 제공할 수 있으며, 코드 설명, 오류 수정, 디버깅 등 복잡한 프로그래밍 과제 수행을 지원하는 데 효과적임을 보여주었다[4]. 이는 AI 에이전트가 단순한 보조 도구를 넘어 프로그래밍 학습 과정을 개인화하여 지원하는 인지적 학습 파트너로 기능할 수 있음을 시사한다.

그러나 프로그래밍 학습을 위한 AI 에이전트 관련 선행 연구들은 주로 맞춤형 피드백의 정확성이나 코드 생성 및 오류 교정 기능에 주목해 왔다. 이에 비해 학습자가 스스로 문제를 분석하고 해결하도록 유도하는 학습 지원 전략을 반영하여 AI 에이전트를 체계적으로 설계·개발한 연구는 상대적으로 부족하다. 특히 C 언어는 문법 구조와 메모리 개념이 복잡하여 초보 학습자가 학습 과정 전반에서 지속적인 지원을 필요로 하지만, C 프로그래밍 학습에 특화된 AI 에이전트 개발 연구는 충분히 이루어지지 않았다.

이에 본 연구에서는 C 프로그래밍 학습을 효과적으로 지원하기 위한 AI 에이전트 프로토타입을 개발하고, 전문가 타당화와 사용성 평가를 통해 그 타당성과 활용 가능성을 확인하고자 하였다. 이를 통해 C 프로그래밍 입문 학습자가 AI에 과도하게 의존하지 않고, 자신의 문제해결 과정을 점검하며 학습을 지속할 수 있는 AI 기반 학습 지원 방안을 제시하고자 한다. 구체적인 연구 문제는 다음과 같다.

연구문제 1. C 프로그래밍 학습 지원을 위한 AI 에이전트 프로토타입은 어떠한 형태로 개발되는가?

연구문제 2. C 프로그래밍 학습 지원을 위한 AI 에이전트 프로토타입의 타당성은 어떠한가?

연구문제 3. C 프로그래밍 학습 지원을 위한 AI 에이전트 프로토타입의 사용성은 어떠한가?

2. 이론적 배경

2.1 프로그래밍 학습을 위한 AI 활용

생성형 AI의 발전은 프로그래밍 학습 환경에 새로운 변화를 가져오고 있으며, AI는 학습의 전 과정을 다양한 방식으로 지원할 수 있다. 첫째, AI는 질의응답 도구로 활용될 수 있다. 학습자는 프로그래밍 과정에서 오류가 발생하거나 코드 이해에 어려움을 겪을 때 AI에게 질문하고 즉각적인 피드백을 받을 수 있다[4]. 둘째, AI는 코드 생성 및 설명 도구로 활용될 수 있다. 학습자가 프롬프트를 통해 원하는 기능이나 알고리즘을 설명하면, AI는 이에 적합한 코드를 생성하고 코드의 의미와 작동 방식을 함께 설명할 수 있다[5]. 셋째, AI는 평가 및 피드백 도구로 활용될 수 있다. 학습자의 코드를 루브릭에 기반하여 자동으로 평가하고, 코드의 오류, 개선점, 보완 방향에 대한 맞춤형 피드백을 제공할 수 있다[6, 7]. 또한 수준별로 실습 문제를 직접 생성할 수도 있다. 이러한 AI 활용 방식은 공통적으로 학습자의 현재 수준과 학습 맥락에 맞는 개별화된 피드백 제공을 기반으로 한다.

프로그래밍 학습에 AI를 도입하면 학습자의 문법적 인지 부하와 심리적 장벽을 완화할 수 있다[2]. AI가 오류 메시지를 실시간으로 해석하고 수정 방향을 제안하여 즉시 수정하는 과정을 통해 디버깅(Debugging) 과정에서의 좌절감을 줄여주며, 이는 컴퓨팅 사고력과 특히 알고리즘적 사고 및 문제해결력 향상으로 이어진다[2, 5]. 학습자 수준에 적절한 개별화된 피드백은 프로그래밍 자기효능감과 학습

동기를 고취하며, AI가 생성한 코드를 비판적으로 분석하는 과정을 통해 코드를 이해하는 리터러시 역량을 강화할 수 있다[2, 8]. 특히 초보 학습자에게 AI의 즉각적인 학습 지원은 문제 해결에 대한 몰입도를 높여주며, 자기주도적 학습 환경을 조성해준다[9]. 그러나 AI에 지나치게 의존할 경우 학습자가 수동적 학습자로 변할 수 있다는 우려도 제기된다[4]. AI가 정답 코드나 해결 절차를 쉽게 제공하면, 학습자는 문제를 분석하고 오류 원인을 추론하며 해결 전략을 수립하는 과정을 충분히 경험하지 못할 수 있다. 따라서 프로그래밍 학습에서 AI는 단순히 정답을 제시하는 도구가 아니라, 학습자의 프로그래밍 사고 과정을 촉진하는 튜터로서 체계적으로 설계될 필요가 있다[4].

2.2 학습 지원 도구로서 AI 에이전트의 가능성

AI 에이전트는 프로그래밍 학습 과정에서 학습자의 문제해결을 지원하는 학습 지원 도구로 활용될 수 있다. AI 에이전트는 학습자의 질문, 코드 작성 상태, 오류 메시지, 시도 이력 등을 바탕으로 즉각적인 피드백을 제공할 수 있으므로, 프로그래밍 학습에서 교수자의 개별 피드백을 보완하는 도구로 기능할 수 있다.

이러한 AI 에이전트의 학습 지원 기능은 스캐폴딩(scaffolding) 전략과 밀접하게 관련된다. 스캐폴딩은 학습자가 혼자서는 해결하기 어려운 문제를 점차 스스로 해결할 수 있도록 교수자나 동료가 학습자의 요구에 맞게 체계적인 지원을 제공하는 교수 전략이다[1]. 스캐폴딩은 시스템 사용과 관련된 절차적 스캐폴딩, 과제 접근 전략 안내를 위한 전략적 스캐폴딩, 핵심 개념 이해를 돕는 개념적 스캐폴딩, 그리고 학습 과정의 점검과 조절을 지원하는 메타인지적 스캐폴딩으로 구분할 수 있다[10]. 이러한 스캐폴딩은 프로그래밍 학습에서도 효과적으로 활용될 수 있다. 예를 들어 김승연과 정인기[1]는 교수자가 스캐폴딩 전략을 사용하여 블록 코딩 프로그래밍 수업을 운영하고 학습 태도와 수업 만족도에 긍정적인 영향을 미쳤다고 보고하였다.

LLM 기술의 발달로 AI 에이전트는 교수자를 보완하여 프로그래밍 학습 과정에서 적시적이고 개별화된 스캐폴딩을 제공할 수 있게 되었다. 직접적인 코드 정답을 제시하기보다 스스로 사고할 수 있도록 학습자가 도움을 요청하면, 질문 중심 힌트, 방향 제시, 정답 코드 안내의 3단계의 피드백을 순차적으로 제공함으로써, 구성주의적 관점에서 학습자 스스로 프로그래밍 지식을 습득할 수 있도록 유도할 수 있다[4]. 학습자의 질문 횟수나 코드 상태에 따라 힌트의 수준을 조절하는 단계별 힌트를 제공하고, 학습 수준이 높아지면 점진적으로 지원을 줄여가는 페이딩(Fading) 전략을 활용할 수 있다[5].

최근 프로그래밍 학습을 지원하기 위하여 AI 에이전트를 개발 연구가 진행되고 있다. 좌하은과 구덕회[4]는 블록 기반 프로그래밍 학습을 지원하기 위해 단계적 비계 제공, 실시간 코드 검증, RAG 기반 응답 생성, 정의적 지원의 네 가지 설계원리를 적용한 AI 에이전트를 개발하였다. 유인환과 박대륜[2]은 AI 에이전트에 교육용 로봇을 결합하여 텍스트 기반 프로그래밍 학습을 지원하였으며, 컴퓨팅 사고력이 향상되었음을 확인하였다. Yan과 그의 동료들[11]은 스캐폴딩을 제공하는 생성형 AI 에이전트를 활용한 프로그래밍 학습에서 학습자의 이해도가 향상되었다고 보고한 바 있다. 이상의 선행연구들은 통해 프로그래밍 학습에서 스캐폴딩 전략을 구현한 AI 에이전트가 컴퓨팅 사고력과 문제해결력 촉진하는 교육적 매개체로서 효과적으로 기능할 수 있음을 보여준다[2]. 특히 문법 복잡성이 높아서 학습자가 빈번한 오류를 경험하는 C 프로그래밍 학습 상황에서 지능형 스캐폴딩 에이전트의 교육적 필요성이 더욱 크다고 할 수 있다.

이상의 논의를 종합하면, C 프로그래밍 학습을 위한 AI 에이전트는 단순히 코드 생성이나 오류 수정 기능을 제공하는 도구가 아니라, 학습자가 문제를 이해하고 해결 과정을 스스로 구성하도록 돕는 학습 지원 도구로 설계될 필요가 있다.

3. 연구 방법

3.1 연구 절차

본 연구는 Richey와 Klein[12]의 설계·개발 연구 방법의 유형 1의 절차를 본 연구 맥락에 맞게 수정하여 적용하였다. 설계·개발 연구는 교육 현장의 실제적 문제를 해결하기 위한 산출물, 도구, 모형 등을 체계적으로 설계·개발하고, 개발 과정과 산출물의 타당성을 검토하는 데 적합한 연구 방법이다. 연구의 목적을 달성하기 위하여 문헌 분석 및 사례 분석, 설계원리 도출, 초기 프로토타입 개발, 전문가 타당화, 사용성 평가의 과정을 거쳐 최종 프로토타입을 개발하였다.

3.2 문헌 분석 및 사례 분석을 통한 설계원리 도출

본 연구에서는 프로토타입의 설계원리와 주요 기능을 도출하기 위하여 문헌 분석 및 사례 분석을 실시하였다. 문헌 분석은 국내외 학술 데이터베이스인 Web of Science, Google Scholar, RISS, KCI 등을 활용하여 수행하였다. 검색어는 '프로그래밍 학습', 'C 프로그래밍', '스캐폴딩 전략', 'AI 튜터', 'AI 챗봇', 'LLM 학습' 등을 조합하여 사용하였다. 이어서 사례 분석에서는 국내외 AI 기반 코딩학습 플랫폼 및 AI 튜터 사례(Elice, goorm, Code.org 등), 상용 LLM/IDE 연동 AI 코딩 보조도구(GitHub Copilot, Cursor, Replit AI 등)를 검토하였다. 문헌 분석 및 사례 분석 결과를 종합하여 C 프로그래밍 학습을 위한 AI 에이전트의 설계원리를 도출하였다. 도출된 설계원리는 학습자 주도 문제해결 원리, 단계적 스캐폴딩 원리, 오류 기반 피드백 원리, 자기설명 기반 AI 수용 원리, 자기조절학습 지원 원리, 정서적 안정 지원 원리이다. 설계원리는 초기 프로토타입의 기능 구성과 상호작용 설계의 기준으로 활용하

였으며, 구체적인 내용은 Table 1과 같다.

Table 1. Design Principle to Develop AI Agent for Supporting Programming Learning

Design Principle	Guidelines
1. Learner Centered Problem Solving Principle [13–16]	1.1 Provide sufficient information to enable students to identify and analyze problems independently.
	1.2 Guide students to first comprehend the task requirements, conditions, inputs, processing procedures, and output formats.
	1.3 Deliver AI feedback based on the student's written code, execution results, and history of previous attempts.
	1.4 Prioritize providing questions, checkpoints, and directional hints rather than presenting the correct solution directly.
2. Scaffolding Principle [17–19]	2.1 Calibrate the level of feedback specificity by considering the student's current learning level, number of repetitive attempts, identical errors, prolonged stagnation, and frequency of hint requests.
	2.2 Expand learning support in a structured sequence: hints, conceptual explanations, procedural guidance, and example code.
	2.3 Provide example code restrictively, only after the student has engaged in sufficient attempts and self-explanation.
	2.4 Fragment feedback into single units of core errors or concepts, rather than presenting multiple pieces of information simultaneously.
3. Error-Based Feedback Principle [16, 20, 21]	3.1 Paraphrase and explain compiler messages in a learner-appropriate manner.
	3.2 Provide concurrent guidance on the underlying causes of the error and the specific code elements that require verification.
	3.3 Offer check questions that allow students to self-correct, rather than immediately providing the correct code.
	3.4 Guide students to re-verify relevant concepts when the identical error occurs repetitively.
4. Self Explanation Based AI Acceptance Principle [22–24]	4.1 Prompt students to explain the rationale before applying AI-suggested modifications or hints.
	4.2 Instruct students to briefly formulate their self-explanations, focusing on the content of the modification, the reasoning behind it, and the expected outcomes.
	4.3 If the self-explanation is insufficient, have the AI prompt additional questions to supplement and scaffold the explanation.
5. Self Regulated Learning Support Principle [25, 26]	5.1 Enable students to visually monitor their current performance stage and progress within the task.
	5.2 Guide students to decompose the overall task into sub-goals and execute them progressively.
	5.3 When stagnation in the problem-solving process is detected, suggest specific remedial actions, such as checking variable values or verifying conditional statements.
6. Emotional Safety Support Principle [27, 28]	6.1 Utilize polite and encouraging conversational expressions within the AI interface so that learners can request assistance without psychological burden.
	6.2 Present performance levels using descriptive narratives focused on current proficiency and future directions for improvement, rather than scores or rankings.
	6.3 Frame help-seeking behavior not as a failure, but as an active learning strategy for problem-solving.

해당 설계원리가 적용된 AI 에이전트는 상용 LLM/IDE 연동형 AI 코딩 보조도구, 기존 교육용 AI 프로그래밍 도구와 다음과 같은 차별성을 가진다. 먼저, 상용 LLM/IDE 연동형 AI 코딩 보조도구는 코드 자동완성, 코드 생성, 오류 수정, 코드 리뷰 등을 통해 개발자의 생산성을 높이는 데 초점을 둔다. 이러한 도구도 사용자의 프롬프트에 따라 단계적 힌트나 소크라테스식 질문을 제공할 수 있으나, 기본적으로 학습자의 문제 분석, 선행 코드 작성, 자기설명, 성찰을 요구하는 학습 절차가 구조화되어 있다고 보기는 어렵다. 즉, 학습 지원의 방향과 수준이 도구 자체의 교육적 설계보다 사용자의 프롬프트 작성 방식에 크게 의존한다. 다음으로 기존 선행연구를 통해 개발된 교육용 AI 프로그래밍 도구는 상용 코딩 보조도구보다 학습 맥락을 더 많이 고려한다. 이러한 도구는 학습자의 질문에 응답하거나, 단계적 힌트를 제공하며, 경우에 따라 정답 코드의 직접 제공을 제한하는 방식으로 프로그래밍 학습을 지원한다. 그러나 대체로 특정 질문이나 오류 상황에 대한 도움 제공에 초점을 두며, 학습자의 과제 수행 전 과정을 하나의 학습 흐름으로 구조화하는 데에는 한계가 있다. 이에 비해 본 연구의 AI 에이전트 프로토타입은 C 프로그래밍 입문 학습자의 형성평가 수행 과정에 맞추어 학습 지원 전략을 시스템 수준에 반영하였다. 학습자는 과제의 요구 조건, 입력, 처리 과정, 출력 형식을 먼저 확인한 뒤 코드를 작성하도록 안내받으며, 첫 코드 작성 또는 문제 분석 전에는 완성 코드나 직접 정답을 제공받지 않는다. 또한 AI의 도움은 힌트, 개념 설명, 절차 안내, 의사코드, 제한적 예시 코드의 순서로 확장되며, 오류 유형, 반복 시도, 힌트 요청 빈도에 따라 피드백 수준이 조절된다. 나아가 AI의 힌트나 코드 리뷰를 적용하기 전 수정 이유와 예상 결과를 설명하도록 하여 AI 응답의 무비판적 수용을 방지하였다.

3.3 전문가 타당화

프로토타입의 타당성과 설계 적절성을 검토하기 위하여 전문가 타당화를 실시하였다. 전문가 타당화는 문헌 분석 및 사례 분석을 통해 도출한 설계원리와 이를 반영하여 개발한 초기 프로토타입이 C 프로그래밍 학습을 지원하는데 적절한지 확인하고, 프로토타입의 개선 방향을 도출하기 위한 목적으로 수행하였다.

전문가 타당화에는 Table 2와 같이 컴퓨터교육, 교육공학, AI 에이전트 개발 관련 연구 또는 실천 경험을 보유한 전문가 7명이 참여하였다. 전문가들은 프로토타입의 주요 기능, 설계원리, 시스템 아키텍처, UI/UX, 세부 기능의 적절성 등을 검토하였다.

전문가 타당화 도구는 크게 두 영역으로 구성하였다. 첫째, 프로토타입의 전반적 타당성을 검토하기 위하여 타당성, 설명력, 유용성, 보편성, 이해도에 관한 문항을 구성하였다. 둘째, 프로토타입의 설계 적절성을 검토하기 위하여 AI 에이전트의 주요 기능, 설계원리, 설계지침, 시스템 아키텍처, UI/UX의 적절성에 관한 문항을 구성하였다. 각 문항

은 4점 Likert 척도(4점=매우 그렇다, 3점=그렇다, 2점=그렇지 않다, 1점=매우 그렇지 않다)로 응답하도록 하였으며, 프로토타입의 강점, 약점, 보완할 점을 자유롭게 작성할 수 있는 개방형 문항을 포함하였다. 수집된 응답은 문항별 평균과 표준편차를 산출하여 분석하였으며, 전문가 수를 고려하여 내용타당도 지수(Content Validity Index, CVI)를 함께 산출하였다. 전문가가 7명인 경우 CVI가 .78 이상이면 내용타당도가 확보된 것으로 판단할 수 있다[29]. 개방형 응답은 프로토타입의 강점, 약점, 개선 요구로 구분하여 정리하고, 반복적으로 제시된 의견과 설계원리 및 기능 개선과 직접 관련된 의견을 중심으로 수정 사항을 도출하였다.

Table 2. Participants in the Expert Validation

Participant	Position	Degree	Experience	Major
P1	Professor	Ph.D.	9 years	AI
P2	Professor	Ph.D.	10 years	Educational Technology
P3	Researcher	Ph.D.	15 years	Computer Education
P4	Teacher	Ph.D.	7 years	Computer Education
P5	Teacher	Ph.D.	24 years	Educational Technology
P6	Teacher	M.S.	12 years	AI Integrated Education
P7	Teacher	M.S.	18 years	AI Integrated Education

3.4 사용성 평가

다음으로 전문가 타당화 결과를 반영하여 수정·보완한 프로토타입의 실제 사용 가능성을 확인하기 위하여 사용성 평가를 실시하였다. 사용성 평가에는 Table 3과 같이 C 프로그래밍 기초 문법을 이해하고 있는 A대학교 컴퓨터교육과 학생 10명이 참여하였다. 참여자는 변수, 자료형, 입출력문, 조건문, 반복문 등 C 언어의 기본 문법을 학습한 경험이 있는 학습자로 선정하였다. 사용성 평가는 과제 수행 기반으로 진행하였다. 먼저 참여자에게 연구의 목적과 프로토타입의 주요 기능을 안내한 후, AI 모드 설정, 과제 확인, 코드 작성 및 실행, AI 튜터를 통한 힌트 요청, 코드 리뷰 확인, 성찰 및 채점 기능을 순차적으로 사용하도록 하였다. 이후 참여자는 제시된 4개의 C 프로그래밍 과제를 수행하면서 필요에 따라 AI 에이전트와 상호작용하였으며, AI의 단계적 힌트, 오류 피드백, 코드 리뷰, 자기설명 유도 기능을 경험하였다. 과제 수행이 종료된 후에는 사용성 평가 설문에 응답하였다.

Table 3. Participants in the Usability Test

Participant	Gender	C Programming Proficiency Level	Interview Participant
S1	Male	Intermediate	●
S2	Male	Novice	●
S3	Female	Novice	
S4	Male	Novice	
S5	Male	Novice	●
S6	Male	Intermediate	
S7	Female	Novice	●
S8	Female	Novice	
S9	Male	Novice	
S10	Male	Novice	●

사용성 평가의 선택형 문항은 도구의 사용성을 평가하는 Morville[30] 허니콤 모델을 번안하여 사용하였다. 프로토타입의 용이성(2문항), 접근성(2문항), 신뢰성(2문항), 매력성(2문항), 유용성(2문항), 가치성(2문항)을 확인하는 총 12문항으로 5점 Likert 척도(1점=전혀 그렇지 않다, 5점=아주 그렇다)로 응답하도록 하였다. 개방형 문항은 프로토타입의 장점, 단점, 개선 방안에 대해 자유롭게 작성하도록 제시하였다. 수집된 자료는 선택형 문항의 경우 문항별 평균과 표준편차를 산출하여 분석하였다. 개방형 응답은 반복적으로 제시된 의견을 중심으로 범주화하였다.

이어서 개방형 의견에 주요 의견을 제시한 참여자 5명을 대상으로 면담을 실시하였다. 프로토타입의 주요 효과성과 개선점을 심층적으로 파악하기 위해 반구조화된 질문지를 사용하여 약 30분 동안 개별 심층 면담을 진행하였다. 면담 데이터를 녹음 후 전사하여, 반복적 비교분석법을 통해 주요 시사점을 도출하였다. 사용성 평가 결과는 최종 프로토타입을 수정·보완하기 위한 근거로 활용하였다.

4. 연구 결과

4.1 프로토타입 개발

4.1.1 시스템 설계

본 연구에서는 문헌 분석 및 사례 분석을 통해 도출한 설계원리를 바탕으로 프로토타입 초안을 개발하였다. 개발된 프로토타입은 대학교 C 언어 입문 학습자가 프로그래밍 과제(형성평가)를 수행하는 과정에서 과제 이해, 코드 작성, 실행, 디버깅, 코드 리뷰, 제출, 성찰을 통합적으로 수행할 수 있도록 설계된 AI 에이전트 기반 학습 지원 시스템이다.

본 연구에서 개발한 AI 에이전트는 자체적으로 언어모델을 개발하거나 학습한 시스템이 아니라, 기존 LLM/API를 활용하여 C 프로그래밍 학습을 지원하도록 설계한 웹 기반 학습 지원 프로토타입이다. 따라서 본 연구의 개발 범위는 도출된 설계원리를 반영하여 C 프로그래밍 학습 맥락에 적합한 에이전트 역할 구조, 스캐폴딩 및 피드백 규칙, 코드

실행 결과와 AI 피드백의 연계 방식, 학습에 적합한 UI 등을 설계하고 이를 웹 기반 시스템으로 구현하는 데 있다. 기존 LLM과 외부 라이브러리 및 서비스는 AI 응답 생성, 코드 작성 환경, 데이터 저장, 코드 실행, AI 응답 품질 관리를 위한 기반 기술로 활용하였다.

프로토타입은 화면의 좌측에 과제 제시 영역, 중앙의 코드 작성 영역, 우측의 AI 상호작용 영역, 데이터 관리 영역으로 구성하였다. 과제 제시 영역에서는 학습 목표, 과제 설명, 입출력 조건, 제약 사항 등을 제시하여 학습자가 해결해야 할 문제 상황을 명확히 파악할 수 있도록 하였다. 중앙의 코드 작성 영역에는 Microsoft가 개발한 웹 기반 코드 에디터 라이브러리인 Monaco Editor를 활용하여, 학습자가 웹에서 C 코드를 작성하고 수정할 수 있도록 하였다. 우측의 AI 상호작용 영역에는 대화, 힌트, 코드 리뷰, 성찰 기능을 배치하여 학습자가 필요에 따라 AI의 도움을 요청하고 자신의 코드를 점검할 수 있도록 하였다. 데이터 관리 영역은 화면에는 보이지 않고 학습 데이터를 저장하고 학습 상태를 추적하는 기능을 수행하도록 하였다.

프로토타입의 기술 스택은 웹 기반 실습 환경과 AI 에이전트 기능을 안정적으로 구현할 수 있도록 구성하였다. 프론트엔드는 Next.js, React, TypeScript 기반으로 개발하였으며, 코드 에디터는 Monaco Editor를 활용하여 C 문법 하이라이팅과 코드 작성 환경을 제공하였다. C 코드 실행은 브라우저 기반 WASM (WebAssembly) 실행 환경을 기본으로 하였다. 백엔드와 데이터 저장은 Supabase를 활용하여 학생 프로필, 과제, 제출물, 대화 기록, 학습 이벤트, 숙달도, 오개념, 개입 기록 등을 저장할 수 있도록 구성하였다. AI 응답 생성 및 에이전트 오케스트레이션은 Vercel AI SDK와 Claude Agent SDK를 활용하여 구현하였으며, Langfuse를 활용하여 AI 호출 이력, 프롬프트 성능, 비용, 응답 품질을 추적할 수 있도록 하였다.

AI 에이전트는 단일 챗봇이 모든 기능을 수행하는 방식이 아니라, 역할이 분화된 멀티 에이전트 구조로 설계하였다. 학습자의 요청 유형, 코드 상태, 실행 결과, 오류 메시지, 학습 이력 등을 종합하여 적절한 하위 에이전트를 선택하고, 해당 에이전트가 정해진 역할과 정책에 따라 응답을 생성하도록 설계하였다. 구체적으로 총괄 에이전트는 학생의 요청이나 시스템 이벤트를 분석하여 요청 유형을 분류하고, 해당 기능을 담당하는 하위 에이전트에 작업을 전달하는 역할을 수행한다. 하위 에이전트는 코딩 도우미, 리뷰어, 디버거, 평가, 상태 추적, 안전 에이전트로 구성하였다. 코딩 도우미 에이전트는 학습자의 질문에 대해 단계적 힌트와 소크라테스식 질문을 제공한다. 리뷰어 에이전트는 학생이 작성한 C 코드를 정확성, 스타일, 메모리 안전성 측면에서 검토하고 개선점을 안내한다. 디버거 에이전트는 실행 결과와 오류 메시지를 해석하여 학생 수준에 맞는 설명을 제공한다. 평가 에이전트는 제출 코드와 성찰 내용을 바탕으로 루브릭 기반 평가를 수행한다. 상태 추적 에이전트는 학생의 시도, 오류, 힌트 요청, 자기설명, 제출 이력을 바탕으로 학습 상태를 갱신한다. 안전 에이전트는 정답 유출, 과도한 코드 제공, 개인정보 노출, 부적절한 응답을 방지하기 위한 안전장치로 작동한다.

4.1.2 설계원리를 적용한 세부 기능 구현

도출된 설계원리는 프로토타입의 기능과 상호작용 방식에 다음과 같이 반영하였다. 첫째, 학습자 주도 문제해결 원리는 AI 응답 제한 방식으로 구현하였다. 학습자가 첫 코드를 작성하거나 최소한의 문제 분석을 수행하기 전에는 AI가 완성 코드나 직접적인 정답을 제공하지 않고, 문제 재진술, 요구 조건 확인, 입력·처리·출력 구조 점검 질문을 우선 제공하도록 하였다. 이를 통해 학습자가 AI에게 문제 해결을 위임하기보다 스스로 문제를 분석하고 해결 전략을 수립하도록 유도하였다.

둘째, 단계적 도움 제공 원리는 AI 튜터의 힌트 제공 방식에 반영하였다. AI의 지원은 개념 설명, 절차 안내, 제한적 예시의 순서로 점진적으로 제공되도록 설계하였다. 첫 번째 도움 요청에서는 문제 해결 방향이나 확인할 조건을 중심으로 안내하고, 반복 시도, 같은 오류가 확인될 때 더 구체적인 개념 설명이나 절차 안내를 제공하도록 하였다. 예시 코드는 학습자가 충분히 시도하고 자신의 정체 지점을 설명한 이후에만 일부 코드를 제한적으로 제공하였다.

셋째, 오류 기반 피드백 원리는 코드 실행 결과와 코드 리뷰 기능에 반영하였다. AI는 컴파일러 메시지를 그대로 전달하는 데 그치지 않고 오류가 발생한 이유, 확인해야 할 코드 등을 학생 수준에 맞게 설명한다. 또한 정답 코드를 직접 제시하기보다 점검 질문을 제공하여 학습자가 직접 오류를 수정하도록 지원하였다.

넷째, 자기설명 기반 AI 수용 원리는 AI 제안 적용 과정에 반영하였다. 학생이 AI가 제시한 힌트, 수정 제안, 코드 리뷰 결과를 적용하기 전에는 해당 제안을 왜 수락하려는지 간단히 설명하도록 하였다. 이러한 자기설명은 단순히 AI의 답을 수동적으로 수용하는 것을 방지하고, 학생이 자신의 코드 수정 이유와 예상 결과를 명시적으로 점검하도록 하는 장치로 기능한다.

다섯째, 자기조절학습 지원 원리는 학습 진행 추적, 집중 시간 관리, 정체 상태 진단, 단계별 안내 기능으로 구현하였다. 학습자는 현재 자신이 과제 이해, 코드 작성, 실행 및 디버깅, 코드 리뷰, 제출, 성찰 중 어느 단계에 있는지 확인할 수 있다. 이를 통해 학습자가 자신의 학습 과정을 점검하고 조절할 수 있도록 지원하였다.

여섯째, 정서적 안정 지원 원리는 AI 피드백의 표현 방식과 학습 수준 제시 방식에 반영하였다. 학생의 질문이나 요청에 AI는 단정적 표현보다 부드러운 표현을 사용하도록 설계하였다. 학습의 평가 결과는 경쟁적 점수/순위보다 수행 수준과 다음 개선 방향을 중심으로 제시하였다.

4.2 전문가 타당화 결과

초기 프로토타입의 전반에 대한 타당성 평가 결과, Table 4와 같이 평균은 3.14~3.71로 나타났으며, CVI는 0.86~1.00으로 산출되었다. 모든 문항에서 CVI가 0.86 이상으로 나타나 내용타당도 기준을 충족하였다.

Table 4. Results of the Expert Validation of AI Agent Prototype

Category	Questionnaire Statement	M	SD	CVI
Validity	The prototype is valid for supporting C programming learning.	3.57	.49	1.00
Explanatory	The prototype clearly explains the knowledge and performance required to support C programming learning.	3.57	.49	1.00
Usefulness	The prototype can be usefully utilized in C programming learning.	3.71	.45	1.00
Universality	The prototype can be universally utilized in C programming learning.	3.14	.64	.86
Understanding	The prototype is comprehensibly presented to facilitate the execution of C programming learning.	3.43	.49	1.00

초기 프로토타입의 설계 적절성에 대한 타당성 평가 결과, Table 5와 같이 평균은 3.43~3.86으로 나타났으며, CVI는 0.86~1.00으로 산출되었다. 모든 문항에서 CVI가 0.86 이상으로 나타나 내용타당도 기준을 충족하였다.

Table 5. Results of the Expert Validation of Design Appropriateness of AI Agent Prototype

Category	Questionnaire Statement	M	SD	CVI
Design Principle	Are the design principles validly constructed to support students' C programming learning?	3.71	.45	1.00
	Are the design guidelines derived from the principles validly constructed to support students' C programming learning?	3.86	.35	1.00
Core Function	Are the core functions appropriately implemented in accordance with the design principles?	3.43	.73	0.86
System Architecture	Is the prototype's system architecture appropriately designed and implemented to realize the corresponding functions?	3.71	.45	1.00
UI/UX	Is the prototype's UI/UX appropriately designed and implemented to support learning?	3.57	.49	1.00

전문가의 의견을 분석한 결과, 프로토타입의 장점은 다음과 같다. 첫째, 최근 AI에 의존하여 프로그래밍 학습을 하는 경우가 많은데, 프로토타입은 단계적 힌트와 질문을 제공하여 학습자의 주도적인 사고 과정을 지원한다는 점이 긍정적으로 평가되었다. 둘째, UI/UX가 직관적이어서 초보 학습자도 쉽게 학습에 참여할 수 있는 점이 긍정적으로 평가되었다. 셋째, 코드를 작성하는 라인과 AI 챗봇이 서로 연결되어 피드백을 매우 구체적으로 제공한다는 점이 높게 평가되었다.

반면 보완이 필요한 사항도 제시되었다. 첫째, 초보 학습자가 AI의 피드백을 이해하기 어려울 수 있으므로 오류 설명을 더 쉽고 명확하게 제시할 필요가 있다는 의견이 제시되었다. 둘째, AI 모드별 차이를 사용자가 직관적으로 이해할 수 있도록 모드 설명을 화면에 더 명확히 제공할 필요가 있다는 의견이 제시되었다. 셋째, 코드 리뷰 결과가 지나치게 많은 정보를 한 번에 제시할 경우 학습자의 인지적 부담이 증가할 수 있으므로, 핵심 오류를 우선적으로 제시하고 필요할 때 추가 설명을 제공하는 방식으로 개선할 필요가 있다는 의견이 제시되었다.

4.3 사용성 평가 결과

전문가 타당화 결과를 반영하여 수정·보완한 프로토타입의 사용성을 확인하기 위한 사용성 평가 결과, 용이성(M=4.20, SD=.75), 접근성(M=4.05, SD=.74), 신뢰성(M=4.40, SD=.58), 매력성(M=4.35, SD=.48), 유용성(M=4.60, SD=.49), 가치성(M=4.40, SD=.73)으로 나타났다. 모든 영역의 평균이 4.00 이상으로 나타나, 참여자들은 프로토타입의 사용성을 전반적으로 긍정적으로 인식한 것으로 확인되었다. 구체적으로 유용성이 가장 높게 나타났으며, 이는 참여자들이 프로토타입이 C 프로그래밍 과제 수행 과정에서 실질적인 도움을 제공한다고 인식하였음을 의미한다. 또한 신뢰성, 가치성, 매력성도 높게 나타나, 학습자들이 AI 에이전트의 응답과 학습 지원 기능을 비교적 신뢰할 수 있으며, 코딩학습에 활용할 만한 가치가 있다고 평가한 것으로 볼 수 있다. 반면 접근성은 모든 영역 중 가장 낮게 나타났다. 이는 프로토타입의 기능별 접근 경로, 사용 방법, 화면 구성 측면에서는 개선의 여지가 있음을 시사한다.

사용성 평가의 개방형 문항에 대한 응답과 심층 면담 내용을 통합하여 분석한 결과, 참여자들은 프로토타입의 강점을 다음과 같이 제시하였다. 첫째, 즉각적인 피드백 제공이다. 참여자들은 C 프로그래밍 학습 과정에서 어려움이 발생했을 때 AI 에이전트에게 바로 질문하고 도움을 받을 수 있다는 점을 긍정적으로 인식하였다. 둘째, 오류 원인에 대한 이해 지원이다. 참여자들은 AI 에이전트가 단순히 정답이나 수정 코드를 제시하는 것이 아니라, 오류가 왜 발생했는지 설명해 준다는 점을 장점으로 제시하였다. 셋째, 심리적 부담 완화이다. 참여자들은 프로그래밍 과제를 수행하다가 어려움에 직면하더라도 AI 에이전트와 상호작용하면서 문제 해결을 계속 시도할 수 있다고 응답하였다. 이는 프로토타입이 초보 학습자의 좌절감을 완화하고, 학습을 중단하지 않도록 돕는 학습 파트너로 기능할 수 있음을 보여준다.

참여자들은 프로토타입의 단점을 보완하기 위하여 다음과 같은 개선 사항을 제안하였다. 첫째, AI 의존 가능성을 줄일 필요가 있다. 참여자들은 AI가 명시적인 답을 제공하지 않도록 설계되었더라도, 도움 수준이 지나치게 구체적이

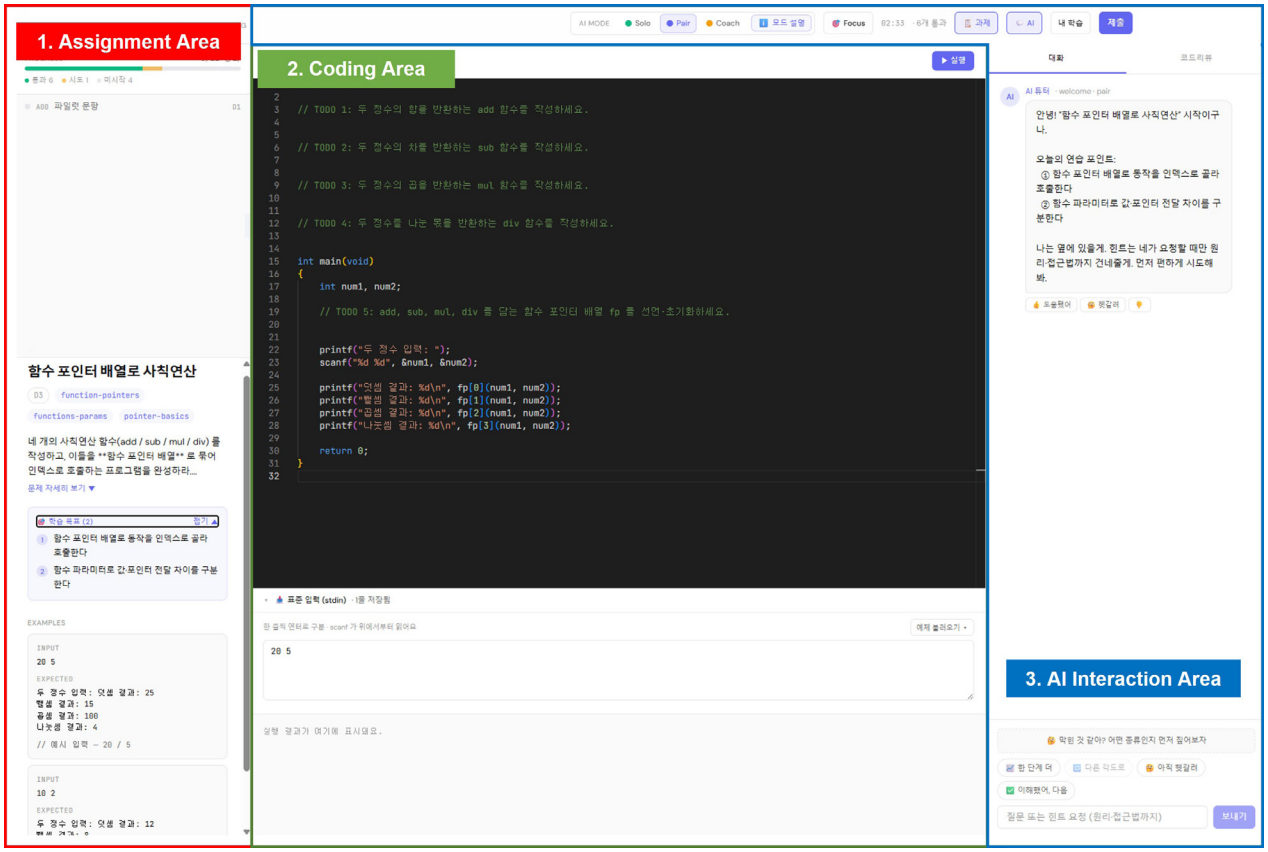

Figure 1. Screen Layout and Functions of Final AI Agent Prototype for Supporting Programming Learning

거나 친절할 경우 학습자가 스스로 사고하기보다 AI의 안내에 의존할 가능성이 있다고 응답하였다. 특히 문제 풀이 초기에 AI 챗봇을 바로 사용할 경우 쉽게 해결책을 얻으려는 태도가 형성될 수 있다는 우려가 제시되었다. 이에 따라 프로토타입에는 학습자가 일정 시간 동안 스스로 문제를 분석하고 일부 코딩을 시도한 이후 AI 도움을 요청하도록 하는 AI 사용지연 기능이 추가될 필요가 있다고 제안하였다. 둘째, 맞춤형 피드백을 제공할 필요가 있다. 학습자의 코드 작성 상황, 오류 유형, 반복 시도 횟수, 힌트 요청 빈도, 등을 종합적으로 고려하여 피드백의 구체성 수준을 조절할 필요가 있다. 예를 들어 저수준 학습자에게는 오류 메시지를 구체적인 용어로 풀어 설명하고 확인해야 할 코드 위치를 라인별로 명확히 안내하고, 고수준 학습자에게는 직접적인 설명보다 점검 질문이나 대안 탐색 질문을 제공하는 방식이 적절하다. 셋째, 응답 속도와 사용 편의성을 개선할 필요가 있다. 일부 참여자들은 프로토타입의 AI 챗봇이 상용 LLM에 비해 응답 속도가 느리고, 프로토타입의 기능별 버튼 위치나 사용 경로가 더 직관적으로 제공될 필요가 있다고 응답하였다. 이에 따라 사용자가 자주 활용하는 기능을 쉽게 찾고 실행할 수 있도록 UI/UX를 개선할 필요가 있다. 예를 들어 AI 모드 설정, 힌트 요청, 코드 리뷰, 성찰 및 채점 기능의 위치를 명확히 구분하고, 각 기능의 역할을 짧은 안내 문구나 아이콘으로 제시할 수 있다. 사용성 평가 결과를 토대로 최종 프로토타입을 수정·보완하였다.

4.4 최종 프로토타입 개발

전문가 타당화와 사용성 평가 결과를 반영하여 최종 프로토타입을 개발하였다. 최종 프로토타입은 Figure 1과 같이 과제 제시 영역, 코드 작성 영역, AI 상호작용 영역, 데이터 관리 영역으로 구성하였다.

첫째, 과제 제시 영역은 학습자가 문제를 이해하고 해결 방향을 설정할 수 있도록 학습 목표, 과제 설명, 입출력 조건을 제공한다. 이를 통해 학습자는 AI의 도움을 요청하기 전에 과제를 통해 이해해야 할 개념과 수행 목표, 해결해야 할 문제, 입출력의 형식, 사용해야 할 문법 요소와 평가 기준을 확인할 수 있다. 이는 학습자가 문제를 충분히 분석한 뒤 코드 작성을 시작하도록 지원하기 위한 것으로, 학습자 주도 문제해결 원리와 자기조절학습 지원 원리를 반영한 것이다.

둘째, 코드 작성 영역은 학습자가 C 코드를 직접 작성하고 실행하며, 실행 결과와 오류 메시지를 확인할 수 있도록 구성하였다. C 에디터는 학습자가 직접 코드를 작성하고 수정할 수 있는 공간으로 제공되며, 코드 실행 기능은 작성한 C 코드의 실행 결과를 확인하도록 지원한다. 결과 및 오류 출력 기능은 컴파일 오류, 실행 결과 등을 제시하여 학습자가 자신의 코드 상태를 점검할 수 있도록 한다. 또한 과제 수행 후에는 성찰 질문을 제공하여 학습자가 문제 해결 과정을 돌아볼 수 있도록 하였다. 이 영역은 학습자가 코드를 직접 작성하고 오류를 수정하는 과정을 반복하도록 지원함으로써 학습자 주도 문제해결 원리, 오류 기반 피드백 원리, 자기

조절학습 지원 원리를 반영한다.

셋째, AI 상호작용 영역은 AI 모드 설정, AI 튜터, 코드 리뷰, 채점 기능으로 구성하였다. AI 모드 설정 기능은 학습자가 AI의 개입 수준을 3수준으로 선택하고 도움 요청 정도를 조절할 수 있도록 지원한다. AI 튜터 기능은 학습자의 질문, 코드 상태, 오류 상황을 바탕으로 단계적 힌트, 개념 설명, 절차 안내를 제공한다. AI 의존 가능성을 줄이기 위하여 최종 프로토타입에서는 학습자가 문제를 먼저 분석하고 일부 코드를 시도한 이후 AI 도움을 요청하도록 흐름을 조정하였다. 코드 리뷰 기능은 학습자가 작성한 코드를 정확성, 코딩 스타일, 메모리 안전성 측면에서 분석하고 개선점을 제시하도록 구성하였다. 채점 기능은 제출 코드와 성찰 내용을 바탕으로 루브릭 기반 평가 결과와 수행 수준을 제시한다. 이 영역은 단계적 도움 제공 원리, 오류 기반 피드백 원리, 자기설명 기반 AI 수용 원리, 정서적 안정 지원 원리를 반영한다.

넷째, 데이터 관리 영역은 화면에는 노출되지 않지만, 학습자의 활동 데이터를 저장하고 AI 에이전트의 응답 생성과 학습 상태 추적을 지원하는 시스템 내부 영역으로 구성하였다. 백엔드에서는 코드 작성 이력, 실행 결과, 힌트 요청 이력, AI와의 대화 기록, 코드 리뷰 결과, 제출물, 성찰 응답 등을 학습 이벤트로 저장한다. 저장된 데이터는 AI 에이전트가 학습자의 현재 코드 상태와 이전 시도 이력, 오류 유형, 힌트 요청 빈도 등을 파악하여 맞춤형 피드백을 생성하는 데 활용된다. 또한 백엔드는 코드 실행 결과와 AI 피드백을 연결하는 역할을 수행한다. 학습자가 코드를 실행하면 실행 결과와 오류 메시지가 백엔드로 전달되고, 디버거 에이전트와 리뷰어 에이전트는 이를 바탕으로 오류 원인, 확인해야 할 코드 요소, 수정 방향을 학습자 수준에 맞게 설명한다. 이때 AI 응답은 단순히 정답 코드를 제공하는 방식이 아니라, 설계원리에 따라 질문, 점검, 단계적 힌트, 코드 리뷰, 성찰 피드백의 형태로 제공되도록 제한된다. 따라서 백엔드 영역은 학습자의 입력과 코드 실행 결과를 AI 에이전트의 피드백 생성 과정에 연결하고, 응답 결과를 다시 학생 화면의 AI 상호작용 영역으로 반환하는 중간 처리 구조로 작동한다.

Table 6. Functions of the Final AI Agent Prototype for Supporting Programming Learning

Domain	Functional Element	Description
Task Presen-tation	Learning Objectives	Present the core concepts to be understood through the task and the performance goals.
	Task Description	Present the problem scenario to be solved, requirement conditions, and execution procedures.
	I/O Conditions	Present the format of input values, output formats, and sample input/output examples.
	Constraints	Present the syntax elements to be used, restrictive conditions, and evaluation criteria.
Code Gene-ration	C Editor	Provide an editor where learners can directly write and modify C code.
	Code Execution	Support the verification of execution results for the written C code.
	Execution Results & Error Output	Present compiler errors, execution results, and test failure status.
	Reflection	Provide post-task reflection questions to support learners in monitoring their problem-solving processes.
AI Inter-action	AI Mode Configu-ration	Support learners in selecting the level of AI intervention and adjusting the degree of help-seeking.
	AI Tutor	Provide progressive hints, conceptual explanations, and procedural guidance based on the learner's queries, code state, and error contexts.
	Code Review	Analyze the learner's written code in terms of correctness, coding style, and memory safety, and suggest areas for improvement.
	Grading	Present rubric-based evaluation results and performance levels based on the submitted code and reflection responses.
Data Manage-ment	Learning Log Storage	Store learning process logs, including code generation, execution, error occurrences, hint requests, code reviews, submissions, and reflections.
	Dialogue Log Storage	Store Q&A exchanges and feedback content between the learner and the AI agent.
	Learning State Tracking	Track the learner's performance state based on repetitive errors, frequency of hint requests, and submission history.
	Feedback Data Storage	Accumulate AI responses, code review outcomes, grading results, and reflection data to support personalized feedback and prototype optimization.

아울러 백엔드 데이터 관리 영역은 교사 지원을 위한 기초 자료를 생성한다. 학습자의 반복 오류, 힌트 요청 빈도, 제출 여부, 성찰 응답, AI 피드백 활용 이력은 학습 상태 추적에 활용되며, 이를 통해 학습자의 지원이 필요한 상황을 파악할 수 있다. 이러한 정보는 교사용 대시보드와 연계되어 교수자가 학습자의 현재 상태를 확인하고, 필요한 경우 개별 피드백이나 수업 중 개입을 제공하는 근거로 활용될 수 있다.

최종 프로토타입은 C 프로그래밍 입문 학습자가 AI를 활용하여 문제 분석, 코드 작성, 오류 수정, 코드 점검, 성찰 과정을 자기주도적으로 수행할 수 있도록 Table 6과 같은 기능 요소로 개발되었다. 이를 통해 학습자는 자신의 수준과 문제 상황에 맞는 도움을 받을 수 있으며, 동시에 AI에 과도하게 의존하지 않고 주도적으로 C 프로그래밍 과제를 해결할 수 있도록 지원받는다.

5. 결론 및 제언

본 연구는 C 프로그래밍 학습자의 학습을 효과적으로 지원하기 위한 AI 에이전트 프로토타입을 개발하고, 전문가 타당화와 사용성 평가를 통해 그 타당성과 활용 가능성을 확인하였다. 본 연구의 의의는 다음과 같다.

첫째, AI를 단순히 정답을 얻는 도구가 아니라 교수적 의도를 가진 학습 지원 도구로서 활용할 수 있는 가능성을 제시하였다. 기존의 생성형 AI 활용이 코드 생성, 오류 수정, 정답 확인에 초점을 두는 경향이 있었다면, 본 연구에서 개발한 AI 에이전트는 학습자가 문제를 먼저 이해하고, 코드를 직접 작성하며, 오류 원인을 분석하고, 자신의 수정 이유를 설명하도록 지원하였다. 이는 프로그래밍 학습에서 AI가 학습자의 사고 과정을 대체하는 것이 아니라, 문제해결 과정을 촉진하고 자기주도적 학습을 지원하는 교수적 도구로 설계될 수 있음을 보여준다.

둘째, 프로그래밍 학습 과정에서 필요한 다양한 지원을 체계적으로 통합하였다. 프로그래밍 학습자는 문제 이해, 코드 작성, 실행, 디버깅, 코드 리뷰, 제출, 성찰의 과정을 반복한다. 이 과정에서 필요한 지원은 단순한 정답 제공이 아니라 단계적 도움, 오류 기반 피드백, 자기설명, 자기조절 학습, 정서적 안정 지원 등으로 다양하다. AI 에이전트는 이러한 지원을 하나의 학습 흐름 안에 통합함으로써, 학습자가 프로그래밍 과제를 지속적으로 수행할 수 있도록 도울 수 있다.

셋째, 교수자가 제공하기 어려운 즉각적·개별화 피드백을 AI 에이전트로 보완할 수 있는 가능성을 제시하였다. 프로그래밍 수업에서는 학습자마다 이해 수준, 문제해결 속도, 도움 요청 시점이 다르기 때문에 교수자가 모든 학습자의 오류를 실시간으로 진단하고 맞춤형 피드백을 제공하는 데 한계가 있다. 본 연구에서 개발한 AI 에이전트는 학습자의 코드 작성 상태, 실행 결과, 오류 유형, 시도 이력 등을 바탕으로 단계적 힌트와 오류 피드백을 제공하도록 설계되었다. 이는 AI 에이전트가 학습자가 필요한 순간에 적절한 도움을 받을 수 있는 학습 환경을 조성할 수 있음을 시사한다.

본 연구의 한계점 및 제언은 다음과 같다.

첫째, 본 연구는 프로토타입의 개발, 전문가 타당화, 사용성 평가에 초점을 두었기 때문에 실제 학습 성과에 대한 효과 검증은 이루어지지 않았다. 후속 연구에서는 실제 C 프로그래밍 수업에 AI 에이전트를 적용하여, 문제해결력, 컴퓨팅 사고력, 자기효능감, 자기조절학습, AI 의존도 등에 미치는 효과를 실증적으로 분석할 필요가 있다.

둘째, AI 에이전트의 피드백 품질은 모델 성능, 프롬프트 설계, 코드 실행 환경, 학습 데이터의 양과 질에 따라 달라질 수 있다. 따라서 후속 연구에서는 AI 응답의 정확성, 적절성, 설명 가능성, 정서적 표현의 안정성을 지속적으로 검토할 필요가 있다.

셋째, 본 연구의 사용성 평가는 C 프로그래밍 기초 문법을 이해하고 있는 대학생 10명을 대상으로 수행되었으므로, 연구 결과를 다양한 학습자 집단과 교육 맥락으로 일반화하는 데에는 한계가 있다. 후속 연구에서는 학습자의 프로그래밍 수준, 전공, 학년, 학습 경험을 다양화하여 프로토타입의 사용성과 적용 가능성을 검토할 필요가 있다.

참고문헌

[1] Kim, S., & Jeong, I. (2011). The scratch programming learning attitude effects of scaffolding based learning strategy. *Journal of The Korean Association of Information Education, 15*(1), 39-49.

[2] Yoo, I., & Park, D. (2026). Exploring AI programming education methods using AI agents and educational robots. *Journal of The Korean Association of Information Education, 30*(1), 149-159. https://doi.org/10.14352/jkaie.2026.30.1.149

[3] Shute, V. (2008). Focus on formative feedback. *Review of Educational Research, 78*(1), 153-189. https://doi.org/10.3102/0034654307313795

[4] Choa, H., & Koo, D. (2026). AI Block Agent: Design and development of an AI agent to support block-based programming learning. *The Journal of Korean Association of Computer Education, 29*(3), 113-122. https://doi.org/10.32431/kace.2026.29.3.011

[5] Hwang, H., & Lee, J. (2025). Designing a Programming Course Model Using AI-Based Pair Programming Techniques. *The Journal of Korean Association of Computer Education, 28*(2), 13-22. https://doi.org/10.32431/kace.2025.28.2.002

[6] Ha, J. (2026). Programming Education Using Generative AI: Focusing on Evaluation of Code Generation Accuracy and Plagiarism Detection. *Journal of Digital Contents Society, 27*(1), 151-164. https://doi.org/10.9728/dcs.2026.27.1.151

[7] Kim, Y. (2025). A Generative AI-Based Personalized Programming Education System with Adaptive Rubric Evaluation. *Journal of The Korea Society of Computer and Information, 30*(10), 11-22. *https://doi.org/10.9708/jksci.2025.30.10.011*

[8] Lee, M., & Kim, H. (2024). Learning-by-Prompting: A Self-Regulated Learning Based Programming Education Framework Utilizing Generative AI Prompting. *Journal of The Korea Society of Computer and Information, 27*(9), 43-53. http://doi.org/10.32431/kace.2024.27.9.005

[9] Ryu, J., & Kim, K. (2026). The Effects of a Problem Solving Process-Oriented Programming Education Program Using Unity on General High School Students' Computational Thinking and Learning Flow: An Exploratory Study on the Potential Use of Generative AI as a Problem Solving Support Tool. *The Journal of Korean Association of Computer Education, 29*(3), 11-19. https://doi.org/10.32431/kace.2026.29.3.002

[10] Hannafin, M., Land, S., & Oliver, K. (2013). Open learning environments: Foundations, methods, and models. *In Instructional-design theories and models* (pp. 115-140).

Routledge.

[11] Yan, L., Martinez-Maldonado, R., Jin, Y., Echeverria, V., Milesi, M., Fan, J., Zhao, L., Alfredo, R., Li, X., & Gašević, D. (2025). The effects of generative AI agents and scaffolding on enhancing students' comprehension of visual learning analytics. *Computers & Education, 234*, 105322. https://doi.org/10.1016/j.compedu.2025.105322

[12] Richey, R., & Klein, J. (2014). *Design and development research: Methods, strategies,* and issues. Routledge.

[13] Kwon, K., & Cheon, J. (2019). Exploring problem decomposition and program development through block-based programs. *International Journal of Computer Science Education in Schools, 3*(1), 3–16. https://doi.org/10.21585/ijcses.v3i1.54

[14] Horváth, G. (2024, October). Teaching Task Specification Efficiently in Introductory Programming Courses in Higher Education. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives* (pp. 111-122). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-73474-8_9

[15] Sun, D., Zheng, Y., Xu, J., & Yang, Z. (2026). When generative AI meets Socratic method: Investigating programming learning dynamics through behaviours, interaction qualities and perceptions. *Journal of Computer Assisted Learning, 42*(2), e70210. https://doi.org/10.1002/jcal.70210

[16] Kazemitabaar, M., Ye, R., Wang, X., Henley, A. Z., Denny, P., Craig, M., & Grossman, T. (2024). CodeAid: Evaluating a classroom deployment of an LLM-based programming assistant that balances student and educator needs. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Article 650, pp. 1–20). ACM. https://doi.org/10.1145/3613904.3642773

[17] Rum, S., & Ismail, M. (2017). Metacognitive support accelerates computer assisted learning for novice programmers. *Journal of Educational Technology & Society, 20*(3), 170-181. https://www.jstor.org/stable/26196128

[18] ivers, K., & Koedinger, K. (2017). Data-driven hint generation in vast solution spaces: A self-improving Python programming tutor. *International Journal of Artificial Intelligence in Education, 27*(1), 37–64. https://doi.org/10.1007/s40593-015-0070-z

[19] Stachel, J., Marghitu, D., Ben Brahim, T., Sims, R., Reynolds, L., & Czelusnik, V. (2013). Managing cognitive load in introductory programming courses: A cognitive aware scaffolding tool. Journal of Integrated Design & Process Science, 17(1), 37–54. https://doi.org/10.3233/jid-2013-0004

[20] Becker, B. (2016). Effective compiler error message enhancement for novice programming students. *Computer Science Education, 2*6(2–3), 148–175. https://doi.org/10.1080/08993408.2016.1225464

[21] Cabo, C. (2026). Persistence of early misconceptions hinders progress in learning computer programming. *IEEE Transactions on Education, 69*(1), 31–39. https://doi.org/10.1109/TE.2025.3637125

[22] Chi, M., Bassok, M., Lewis, M., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science, 13*(2), 145–182. *https://doi.org/10.1207/s15516709cog1302_1*

[23] Hausmann, R. & VanLehn, K. (2010). The effect of self-explaining on robust learning. *International Journal of Artificial Intelligence in Education, 20*(4), 303–332. https://doi.org/10.3233/JAI-2010-010

[24] Tamang, L., Alshaikh, Z., Ait Khayi, N., Oli, P., & Rus, V. (2021). A comparative study of free self-explanations and Socratic tutoring explanations for source code comprehension. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (SIGCSE '21) (pp. 219–225). Association for Computing Machinery. https://doi.org/10.1145/3408877.3432423

[25] Peng, J., Wang, M., Sampson, D., & van Merriënboer, J. (2019). Using a visualisation-based and progressive learning environment as a cognitive tool for learning computer programming. *Australasian Journal of Educational Technology, 35*(2). https://doi.org/10.14742/ajet.4676

[26] Morrison, B., Margulieux, L., & Guzdial, M. (2015). Subgoals, context, and worked examples in learning computing problem solving. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research* (ICER '15) (pp. 21–29). Association for Computing Machinery. https://doi.org/10.1145/2787622.2787733

[27] D'Mello, S., & Graesser, A. (2013). AutoTutor and affective AutoTutor: Learning by talking with cognitively and emotionally intelligent computers that talk back. *ACM Transactions on Interactive Intelligent Systems, 2*(4), Article 23. https://doi.org/10.1145/2395123.2395128

[28] Jiménez, S., Juárez-Ramírez, R., Castillo, V., Licea, G., Ramírez-Noriega, A., & Inzunza, S. (2018). A feedback system to provide affective support to students. *Computer Applications in Engineering Education, 26*(3), 473–483. https://doi.org/10.1002/cae.21900

[29] Rubio, D., Berg-Weger, M., Tebb, S., Lee, E., & Rauch, S. (2003). Objectifying content validity: Conducting a content validity study in social work research. *Social work research, 27*(2), 94-104. https://doi.org/10.1093/swr/27.2.94

[30] Morville, P. (2005). Ambient findability: *What we find changes who we become.* O'Reilly Media, Inc.

김민지

· 2023년 서울대학교 교육학과 교육공학전공(교육학 박사)
· 2023년~2025년 서울대학교 학부대학 디지털교육 혁신센터 연구교수
· 2026년~현재 을지대학교 교양학부 조교수

관심분야 : AI융합교육, 학습과학, 교수설계, 에듀테크

mjkim@eulji.ac.kr

이동국

· 2019년 한국교원대학교 교육학과 교육공학전공 (교육학박사)
· 2025년~현재 경북대학교 정보·컴퓨터교육과 조교수

관심분야 : AI 융합교육, 컴퓨팅 사고력, 교사 전문성 개발, 첨단 학습환경 설계, AI 에이전트

dklee@knu.ac.kr

부 록

Table 1. 설계원리 및 설계지침

설계원리	설계지침
1. 학습자 주도 문제해결 원리 [13–16]	1.1. 학생이 문제를 확인하고 분석할 수 있도록 충분한 정보를 제공한다.
	1.2. 학생이 과제의 요구 조건, 입력, 처리 과정, 출력 형식을 먼저 파악하도록 안내한다.
	1.3. AI 피드백은 학생이 작성한 코드, 실행 결과, 시도 이력을 바탕으로 제공한다.
	1.4. AI는 정답 제시보다 질문, 점검, 방향 힌트를 우선 제공한다.
2. 단계적 도움 제공 원리 [17–19]	2.1. 학생의 학습 수준, 반복 시도, 동일 오류, 장시간 정체, 힌트 요청 빈도를 고려하여 피드백의 구체성 수준을 조절한다.
	2.2. 학습 지원은 힌트, 개념 설명, 절차 안내, 의사코드, 예시 코드의 순서로 확장한다.
	2.3. 예시 코드는 충분한 시도와 자기설명 이후에만 제한적으로 제공한다.
	2.4. 피드백은 한 번에 여러 내용을 제시하지 않고 하나의 핵심 오류나 개념 단위로 분절한다.
3. 오류 기반 피드백 원리 [16, 20, 21]	3.1. 컴파일러 메시지를 학생 수준에 맞게 풀어 설명한다.
	3.2. 오류가 발생한 이유와 확인해야 할 코드 요소를 함께 안내한다.
	3.3. 정답 코드를 바로 제시하기보다 학생이 직접 수정할 수 있는 점검 질문을 제공한다.
	3.4. 동일 오류가 반복될 경우 관련 개념을 다시 확인하도록 안내한다.
4. 자기설명 기반 AI 수용 원리 [22–24]	4.1. AI의 수정 제안, 힌트를 적용하기 전, 학생이 그 이유를 설명하도록 한다.
	4.2. 자기설명은 수정 내용, 수정 이유, 예상 결과를 중심으로 간단히 작성하도록 한다.
	4.3. 자기설명이 부족할 경우 AI가 추가 질문을 통해 설명을 보완하게 한다.
5. 자기조절학습 지원 원리 [25, 26]	5.1. 학생이 현재 과제의 수행 단계와 진행 상태를 시각적으로 확인할 수 있게 한다.
	5.2. 전체 과제를 작은 목표로 나누어 단계별로 수행하도록 안내한다.
	5.3. 문제해결 과정의 정체가 감지되면 변수 값 확인, 조건식 점검과 같은 구체적인 회복 행동을 제안한다.
6. 정서적 안정 지원 원리 [27, 28]	6.1. AI는 학습자가 부담 없이 도움을 요청할 수 있도록 친절하고 격려적인 대화 표현을 사용한다.
	6.2. 수행 수준은 점수나 등수보다 현재 수준과 다음 개선 방향 중심의 서술형 표현으로 제시한다.
	6.3. 도움 요청을 실패가 아닌 문제해결을 위한 학습 전략의 일부로 안내한다.

Table 2. 전문가 타당화 참여자 프로필

구분	직위	학력	경력	전공 및 연구 분야
P1	교수	박사	9년	인공지능
P2	교수	박사	10년	교육공학
P3	연구위원	박사	15년	컴퓨터교육
P4	교사	박사	7년	컴퓨터교육
P5	교사	박사	24년	교육공학
P6	교사	석사	12년	AI융합교육
P7	교사	석사	18년	AI융합교육

Table 3. 사용성 평가 참여자 프로필

구분	성별	C 프로그래밍 학습 수준	면담 참여
S1	남	중급	●
S2	남	초급	●
S3	여	초급	
S4	남	초급	
S5	남	초급	●
S6	남	중급	
S7	여	초급	●
S8	여	초급	
S9	남	초급	
S10	남	초급	●

Table 4. 프로토타입의 전반에 대한 타당성 평가 결과

구분	문항	M	SD	CVI
타당성	프로토타입은 C 프로그래밍 학습을 지원하는 데 타당하다.	3.57	.49	1.00
설명력	프로토타입은 C 프로그래밍 학습을 지원하는 데 필요한 지식과 수행을 명확히 설명한다.	3.57	.49	1.00
유용성	프로토타입은 C 프로그래밍 학습에 유용하게 활용될 수 있다.	3.71	.45	1.00
보편성	프로토타입은 C 프로그래밍 학습에 보편적으로 이용될 수 있다.	3.14	.64	.86
이해도	프로토타입은 C 프로그래밍 학습을 수행하는 데 이해하기 쉽게 표현되었다.	3.43	.49	1.00

Table 5. 프로토타입의 설계 적절성 평가 결과

구분	문항	M	SD	CVI
설계원리	설계원리는 학생의 C 프로그래밍 학습을 지원하는데 타당하게 구성되었는가?	3.71	.45	1.00
설계지침	설계원리에 따른 설계지침은 C 프로그래밍 학습을 지원하는데 타당하게 구성되었는가?	3.86	.35	1.00
주요 기능	설계원리에 따라 주요 기능이 적절하게 구현되었는가?	3.43	.73	0.86
시스템 아키텍처	프로토타입의 시스템 아키텍처는 해당 기능을 구현하는 데 적절하게 설계/구현되었는가?	3.71	.45	1.00
UI/UX	프로토타입의 UI/UX는 학습을 지원하는 데 적절하게 설계/구현되었는가?	3.57	.49	1.00

Table 6. 최종 프로토타입의 기능 요소

영역	기능 요소	내용
과제 제시	학습 목표	과제를 통해 이해해야 할 개념과 수행 목표 제시
	과제 설명	해결해야 할 문제 상황, 요구 조건, 수행 절차 제시
	입출력 조건	입력값의 형태, 출력 형식, 예시 입출력 제시
	제약 사항	사용해야 할 문법 요소, 제한 조건, 평가 기준 제시
코드 작성	C 에디터	학습자가 직접 C 코드를 작성하고 수정 에디터 제공
	코드 실행	C 코드의 실행 결과 확인 지원
	결과 및 오류 출력	컴파일 오류, 실행 결과, 테스트 실패 여부 제시
	성찰	과제 수행 후 성찰 질문을 제공하여 문제 해결 과정을 점검하도록 지원
AI 상호작용	AI 모드 설정	학습자가 AI의 개입 수준을 선택하고 도움 요청 정도를 조절할 수 있도록 지원
	AI 튜터	학습자의 질문, 코드 상태, 오류 상황을 바탕으로 단계적 힌트, 개념 설명, 절차 안내 제공
	코드 리뷰	학습자가 작성한 코드를 정확성, 코딩 스타일, 메모리 안전성 측면에서 분석하고 개선점 제시
	채점	제출 코드와 성찰 내용을 바탕으로 루브릭 기반 평가 결과와 수행 수준 제시
데이터 관리	학습 이력 저장	코드 작성, 실행, 오류 발생, 힌트 요청, 코드 리뷰, 제출, 성찰 등 학습 과정 이력 저장
	대화 기록 저장	학습자와 AI 에이전트 간 질의응답 및 피드백 내용 저장
	학습 상태 추적	반복 오류, 힌트 요청 빈도, 제출 이력 등을 바탕으로 학습자의 수행 상태 추적
	피드백 자료 저장	AI 응답, 코드 리뷰 결과, 채점 결과, 성찰 자료를 축적하여 맞춤형 피드백과 프로토타입 개선 지원