

생성형 AI를 활용한 프로그래밍 교육용 코드 생성 프롬프트 개발

Developing Code Generation Prompts for Programming Education with Generative AI

김슬기[†]
Seulki Kim[†]

요 약

딥러닝 기반의 AI 기술, 특히 생성형 AI는 최근 급격한 발전으로 사회에 많은 영향을 미치고 있다. 이로 인해 AI 교육 및 정보 교육이 강조되고 학생들의 역량을 기르기 위한 많은 교육 연구들이 이루어지고 있다. 본 연구는 프로그래밍 교육을 위한 교수 학습 환경 개선을 위해 생성형 AI의 적용 가능성을 확인하고자 하였다. 특히 생성형 AI의 텍스트 생성 기능을 활용하여 코드 및 교수학습 자료를 생성하기 위해 교수학습 전략과 프롬프트 엔지니어링 기법을 적용한 프롬프트를 개발하였다. 그 결과, 개발된 프롬프트는 교수학습 환경에 적합한 응답 결과를 도출하고 학습자 수준을 고려한 코드와 연습 문제를 생성할 수 있었으며, 일반적인 프롬프트와 비교하여 전체적으로 유사도가 높은 양질의 코드를 생성함을 확인하였다. 본 연구의 결과가 생성형 AI를 활용하여 다양한 수준의 학습자에게 개별 교육이 필요한 AI교육 및 정보교육 환경을 개선하는데 도움이 될 수 있기를 기대한다.

주제어: ChatGPT, SW · AI교육, 프로그래밍, 파이썬, 생성형 AI

ABSTRACT

Deep learning-based AI technologies, especially generative AI, have been profoundly impacting society due to their rapid advancements. This has led to an emphasis on AI and informational education, with numerous educational research endeavors undertaken to nurture students' capabilities. This study aimed to explore the potential of applying generative AI to enhance the learning environment for programming education. Specifically, we utilized the text generation capability of generative AI to produce code and instructional materials by implementing instructional strategies and prompt engineering techniques. The developed prompts were found to generate responses suitable for the educational environment, considering learners' proficiency levels to produce code and practice questions. Compared to standard prompts, they yielded high-quality code with a higher overall similarity. We hope that our findings will aid in improving the AI and informational educational environment, providing individualized education to learners of varying levels using generative AI.

Keywords: ChatGPT, SW · AI Education, Programming, Python, Generative AI

1. 서론

딥러닝을 활용한 인공지능(Artificial Intelligence, AI) 기술의 발전은 사회에 많은 영향을 주고 있으며 다양한 산업에서 활용되고 있다. 최근에는 인터넷 공간을 통해 수집되고 공유 및 활용이 용이해진 데이터

와 이를 처리할 수 있는 컴퓨팅 성능의 발전이 AI의 급속한 발전을 이끌고 있다[1].

다양한 AI 기술 중 생성형 AI는 최근 강력한 텍스트 생성 기능을 중심으로 급속도로 활용 범위가 넓어지고 있으며 사회적으로 큰 영향을 미치고 있다. 특히 방대한 텍스트 학습을 통해 생성된 모델은 사용자가

[†]정 회 원: 안산원곡초등학교 교사

논문접수: 2023년 07월 21일, 심사완료: 2023년 08월 25일, 게재확정: 2023년 08월 30일

입력하는 프롬프트에 따라 다양한 목적과 수준에 따른 응답을 도출할 수 있으며 전문 지식 및 코드 생성 등의 다양한 분야에서 활용할 수 있는 전천후 어플리케이션으로 각광받고 있다[2].

AI 기술이 사회에 미치는 영향이 커짐에 따라 기술에 대한 이해와 이를 활용할 수 있는 기초 기본 역량이 강조되고 있으며 관련 역량 교육에 대한 사회적 요구도가 높아지고 있다[1]. 이에 AI 교육과 정보교육이 강조되고 효과적인 교육 방법 중 하나인 프로그래밍 교육과 관련된 연구들이 많이 이루어졌다. 그 결과 교수학습 환경에서의 학습자가 겪는 어려움과 효과적인 교수학습 방법 개발 등의 유의미한 연구 결과들이 발표되고 교육 현장에 도움을 주고 있다[3-5].

생성형 AI의 텍스트 생성 기능은 프로그래밍 교육에서 다양한 교육용 코드 및 학습 문제와 같은 교수학습 자료를 생성하는데 활용될 수 있을 것이다. 이는 실시간으로 개별 학습자 수준에 따른 자료를 제공하고 교수학습 환경을 개선하는데 큰 도움을 줄 수 있을 것으로 예상된다[6]. 이에 본 연구는 생성형 AI를 활용하기 위한 프롬프트를 교수학습 전략과 프롬프트 엔지니어링 측면에서 연구하고 생성 결과를 분석하여 적용 가능성을 탐색하고자 한다.

2. 연구 배경

2.1 생성형 AI(Generative AI)

생성형 AI(Generative AI)는 딥러닝 기술을 통해 데이터의 패턴과 분포를 도출하고 학습하여 비디오, 이미지/그래픽, 텍스트, 오디오, 동영상 등 기존 디지털 콘텐츠를 인공적으로 생성할 수 있는 기술을 의미한다[6,7]. 최근에는 인간의 언어를 이해하고 생성할 수 있는 자연어 처리(Natural Language Processing, NLP) 분야에서 생성형 AI가 혁신적인 역할을 하고 있다[2]. 특히 트랜스포머(Transformer) 아키텍처가 기존의 순환 신경망(Recurrent Neural Network, RNN)의 연산 속도 문제를 극복하기 위한 대안으로 활용되면서 다양한 길이의 시퀀스와 문맥 정보를 가진 자연어를 효율적으로 처리할 수 있게 되었다[2]. 이에 트랜스포머 아키텍처를 활용하여 대규모 데이터를 사전 학습한 모델(Pre-Trained Model)은 수많은 종류의 데이터를 입력받고 확률적으로 생성하여 활용할 수 있게 되었다[2]. 특히 다양한 모델 중 방대한 언어 데이터를 중심으로 학습하고 이를 바탕으로 구성된 대형 언어 모

델(Large Language Model, LLM)은 인간의 언어를 확률적으로 이해하고 가장 최적의 응답을 생성하여 제공할 수 있게 되면서 다양한 산업 분야에서 폭넓게 활용되고 있으며 빠른 속도로 개선 및 발전하고 있다[8].

대표적인 LLM 서비스 중 하나인 ChatGPT는 GPT(Generative Pre-Trained Transformer) 모델을 활용한 챗봇 서비스로 트랜스포머 아키텍처를 통해 레이블이 지정되지 않은 방대한 언어 데이터를 기반으로 사전학습 되었다[9,10]. ChatGPT는 뛰어난 성능으로 출시 이후 엄청난 인기를 얻으며 의료 및 교육, 챗봇, 금융, 엔터테인먼트, 사이버 보안, 마케팅, 비전 작업, 프로그래밍 등 다양한 애플리케이션과 분야에 접목되어 활용되고 있다[2].

2.2 생성형 AI의 코드 생성

생성형 AI를 활용한 코드 생성은 생성형 AI, 특히 LLM을 활용하여 프로그램에 대한 자연어 설명이나 목적 및 사양과 같은 상위 수준의 표현을 통해 전체 코드를 자동으로 생성하거나 일부를 완성하는 과정을 의미한다[11]. 이는 프로그래밍 효율성을 개선하고 오류를 해결하는데 도움을 줄 수 있으며 ChatGPT와 같은 대형 언어 모델과 코드를 생성하는데 초점을 맞춘 Github의 CoPilot, Amazon의 CodeWhisperer 등이 대표적인 서비스로 활용되고 있다[12].

Biswas(2023)는 ChatGPT가 프로그래밍을 지원하기 위해 복잡한 개념과 기술에 대한 이해를 도울 수 있음을 주장하고 리소스 및 기술 문제를 진단하고 문제 해결을 위한 설명 및 예제와 지침 생성이 가능함을 확인하였다. 특히 특정 목적의 코드 전체나 재사용이 가능한 작은 조각의 스니펫 생성 및 오류 수정, 코드 최적화에 활용할 수 있는 방안에 대해 제안하였다[13].

Liu 외(2023)는 ChatGPT를 활용한 코드 생성 성능을 확인하기 위해 Text to Code, Code to Code의 관점에서 ChatGPT에게 과업을 제시하거나 구체적인 정보를 제시하는 등의 다양한 입력의 방식을 적용하고 그에 따른 코드의 품질을 분석하였다[8].

Yetişiren 외(2023)는 대표적인 코드 생성 LLM인 CoPilot과 CodeWhisperer, ChatGPT의 성능을 비교하기 위해 코드의 유효성, 정확성, 신뢰성 등의 관점에서 생성된 코드를 분석하였으며 전체적으로 ChatGPT가 높은 성능을 보이며 코드 생성을 위해서는 정확하고 명확한 문제 설명이 필수적임을 도출하였다[12].

Cao 외(2023)는 딥러닝 프로그램의 디버깅과 복구

를 위한 목적에서 ChatGPT를 활용했을 때 얻을 수 있는 이점과 한계를 연구하였다. 수정하고자 하는 코드의 위치, 데이터셋, 코드 목적에 대한 추가 설명 등의 입력 방식에 따라 결과의 차이를 분석하고 템플릿을 개발하여 일반화하였다[14].

선행 연구들은 생성형 AI의 코드 생성 능력의 범위를 확인하고 산업 현장에서 활용하기 위해 각각도로 결과를 분석하고 활용 가능성을 확인하였다. 이러한 코드 생성 기능은 교수학습 환경에 적합한 교육 목적의 코드를 생성하는데 활용될 수 있을 것이며 이를 위해 생성형 AI에게 입력하는 방식에 대한 면밀한 설계가 선행되어야 함을 도출할 수 있다.

2.3 프롬프트 엔지니어링(Prompt Engineering)

LLM의 맥락에서 프롬프트는 사용자가 제공한 입력 또는 쿼리(Query)를 의미하며, 이는 모델의 응답 생성 프로세스의 시작점 역할을 한다. 프롬프트는 단순한 단어나 구문, 질문에서부터 보다 정교하고 상세한 지침에 이르기까지 복잡성이 다양할 수 있다[15].

프롬프트 엔지니어링(Prompt Engineering)은 대규모 언어 모델의 출력 성능을 높이기 위해 프롬프트 혹은 입력 방법을 만드는 관점으로 입력과 알고리즘에 따라 결론을 도출하는 기존의 프로그래밍과 비교하여 비교적 덜 결정론적이지만 모델 혹은 프로그램에 지시를 내리는 역할을 하게 된다는 관점에서 공통점을 가진다[15,16].

Giray(2023)는 언어 생성 및 텍스트 기반 AI 모델에서 프롬프트 엔지니어링의 중요성을 강조하고 프롬프트 설계 시 고려해야 할 사항에 대해서 맥락과 명확성, 편향성의 완화, 구체성과 유연성의 균형, 윤리적 고려사항, 모델의 제한점 이해 등을 중심으로 다양한 요소를 제시하였다[16].

White 외(2023)는 양질의 결과를 얻기 위해 프롬프트를 문서화하여 체계적으로 정리하였다. 프롬프트 엔지니어링은 특정 맥락에서 반복되는 문제에 재사용 가능하며 소프트웨어 프로그래밍이 체계적인 접근 방식을 통해 개발 과제를 해결하는 것처럼 프롬프트 패턴을 통해 LLM과 사용자 간의 체계적인 상호작용이 필요함을 주장하였다. 특히 Table 1과 같이 프롬프트를 6개의 범주와 16개의 패턴 원리로 유목화 하여 제시하였다[17].

Table 1. Prompt Patten Category

Pattern Category	Prompt Pattern
Input Semantics	Meta Language Creation
Output Customization	Output Automater Persona Visualization Generator Recipe Template
Error Identification	Fact Check List Reflection
Prompt Improvement	Question Refinement Alternative Approaches Cognitive Verifier Refusal Breaker
Interaction	Flipped Interaction Game Play Infinite Generation
Context Control	Context Manager

Avila-Chauvet 외(2023)는 소프트웨어의 동작에 초점을 맞춰 원하는 과제를 달성하기 위한 프로그래밍 도구로서의 ChatGPT 활용 가능성을 탐색하고 정확도 높은 답변을 얻기 위해 명확하고 간결한 질문, 과제에 대한 맥락과 세부 정보 제공, 구체적인 설명 기준 등의 프롬프트 설계 원리를 제시하였다[18].

Chen 외(2023)는 프롬프트를 통해 ChatGPT와 같은 LLM을 프로그래밍 문맥에 따라 질문에 대답할 수 있는 튜터로 사용할 수 있음을 확인하고 프롬프트에 상세한 추가 정보를 입력하여 정확도 높은 응답을 생성하는 방법을 연구하였다[19].

Polak 외(2023)는 LLM을 활용하여 연구 결과에서 데이터를 추출하기 위한 프롬프트를 설계하였다. 명시적으로 원하는 과업에 대한 사전 정보를 제공하지 않는 제로샷(Zero-shot) 프롬프트와 사전 정보를 제공하고 이를 바탕으로 결과를 도출하는 소수샷(Few-shot) 프롬프트의 결과를 비교하였으며 정확한 응답을 도출하기 위해서는 명확하고 맥락이 풍부한 정보를 제공하는 것이 중요함을 주장하였다[20].

선행 연구를 통해 방대한 데이터로 사전 학습된 LLM은 입력되는 정보에 따라 응답의 종류와 질이 결정되며 프롬프트 엔지니어링 원리를 통해 면밀하게 설계된 입력 혹은 질문이 필요함을 도출할 수 있다.

2.4 프로그래밍 교육

AI가 사회에 미치는 영향이 커짐에 따라 컴퓨팅의 기본적인 수단이 되는 프로그래밍 교육에 대한 요구

가 높아지고 있으며 양질의 교육을 제공하기 위한 연구들이 지속적으로 이루어졌다.

Piteira 외(2013)는 프로그래밍 교육을 접하는 초보 학습자들의 어려움에 초점을 맞춰 어려움을 유발하는 요소를 학습 개념, 학습 환경, 교육 방법, 교육 주제, 교육 자료로 구분하였다. 특히 학습자와 교사의 의견을 수렴하였으며 학습자는 교육 방법 측면에서 혼자 학습하는 부분을 어려워했으며, 학습 자료의 경우 코드 예제와 인터넷 튜토리얼 등의 콘텐츠를 선호함을 도출하였다[21].

최정원 외(2014)는 초보 학습자에게 적절한 교육용 프로그래밍 언어를 활용하여 구조화된 학습 단계가 있는 콘텐츠를 제공하고 학습자가 직면하는 어려움에 대해서 분석하였다. 그 결과 프로그래밍의 개념을 내면화하기 위한 전략의 필요성과 정교화된 알고리즘 작성 연습, 다양한 문제 해결 경험의 제공 등이 필요함을 주장하였다[3].

김진숙 외(2015)는 효율적인 정보 교과 교육을 위해 지식, 기능, 태도 중 기능 영역을 중심으로 수업 모델을 개발하고 컴퓨팅 사고력 신장의 목표를 달성하기 위해 수업 모델의 각 단계에 컴퓨팅 사고력 구성 요소가 포함될 수 있도록 설계하였다. 그 결과 시연 중심, 재구성 중심, 개발 중심, 디자인 중심, CT 요소 중심의 5가지 수업 모델과 그 예시를 제안하였다[4].

김갑수(2015)는 프로그래밍 학습의 자기 조절 단계를 문제 이해, 설계, 구현, 시험, 유지 보수의 5단계로 구분하였다. 이를 활용하여 주어진 코드를 그대로 따라 하며 기능을 익히는 모방(Copying) 단계, 주어진 코드를 바탕으로 자신의 아이디어를 추가하여 새로운 결과를 만들어 내는 변형(Changing) 단계, 완전히 새로운 프로그램을 만드는 창작(Creating) 단계, 주어진 문제나 프로젝트에 자신만의 독특한 접근법을 활용해 문제를 해결하는 도전(Challenging) 단계로 이루어진 학습 모델을 개발하였다[5].

선행 연구를 종합하면 효과적인 프로그래밍 교육을 위해 학습자의 인지 부담을 최소화하고 학습자 수준에 적합한 개념과 실습 및 문제 해결 경험을 단계적으로 제공해야 함을 도출할 수 있다. 이에 본 연구는 학습자에게 효과적인 프로그래밍 교육을 제공할 수 있는 교수학습 환경을 구축하기 위한 방법으로 생성형 AI의 활용 가능성을 확인하고자 한다. 선행 연구를 바탕으로 교수학습 전략과 프롬프트 엔지니어링 원리를 활용하여 교육에 적합한 프롬프트를 개발하고 이를 통해 생성된 코드 및 학습 자료를 분석하여 구체적

인 적용 가능성을 확인하는 연구를 진행하고자 한다.

3. 연구 방법

3.1 연구 도구

본 연구에서 코드 및 학습 자료 생성을 위해 활용하는 생성형 AI는 코드 생성에 강점을 보이며 다양한 방면에서 널리 활용되고 있는 ChatGPT를 활용하였다[12]. 또한 생성형 AI의 접근성을 높이기 위해 회원 가입만으로 바로 활용 가능한 GPT-3.5 모델을 선정하였으며 생성하는 프로그래밍 언어는 ChatGPT의 학습 데이터로 사용되고 프로그래밍 교육에 많이 활용되는 파이썬으로 선정하였다[10,11].

3.2 연구 방법

ChatGPT가 생성한 코드를 평가하기 위해 교수학습 상황을 가정하여 학습자가 따라할 수 있는 단계인 ‘시연 및 실습 단계’와 문제 해결 경험 제공으로 내면화를 위한 ‘연습 문제 단계’로 나누었으며 프롬프트 엔지니어링 원리를 도입하여 해당 단계에서 활용할 수 있는 한글 프롬프트를 개발하였다[4,5].

생성형 AI는 같은 프롬프트 입력에도 세션마다 서로 다른 응답을 생성하는 특성을 가지고 있다[12]. 이에 프로그래밍 교육 교수학습 환경에서 생성형 AI를 활용하기 위해서는 모든 환경에서 모든 학생들에게 동일한 양질의 코드를 제공할 수 있는 관점에서 생성된 코드의 유사도에 대한 평가가 선행될 필요가 있다.

연구를 위해 ChatGPT API를 활용하여 코드 생성을 자동화하였으며 생성 모델은 ChatGPT 챗봇과 동일한 성능을 가지는 ‘gpt-3.5-turbo’ 모델을 활용하였다. 실제적인 유사도 측정을 위해 본 연구를 통해 개발된 프롬프트를 입력하고 세션 당 5개의 코드를 100회 반복하여 생성하였으며, 대조군으로 단순 코드 생성을 요청하는 일반적인 프롬프트를 입력하여 동일 횟수의 코드를 생성하고 유사도를 비교 분석하고자 하였다.

코드 간의 유사도를 측정하기 위한 방법으로는 언어 모델인 BERT(Bidirectional Encoder Representations from Transformers)를 기반으로 코드의 구조와 패턴을 파악할 수 있도록 학습된 CodeBERT를 활용하였다. CodeBERT를 통해 생성된 코드를 임베딩하여 코드의 전체적인 문맥 및 정보를 담은 벡터를 생성하고 세션 및 세션 간 벡터의 코사인 유사도를 분석하였다[22].

4. 연구 결과

4.1 프롬프트 개발

시연 및 실습 단계에서 활용하기 위한 코드 생성을 위해 ‘예시 코드 생성’ 키워드를 중심으로 프롬프트를 설계하였다. 특히 ‘페르소나 패턴(Persona pattern)’ 원리를 적용하여 코드 생성의 목적을 파이썬 교육으로 한정하였으며 API에 적용하기 위해 시스템 명령어를 활용하여 ‘파이썬을 가르치는 선생님’으로 설정하였다. 페르소나 패턴은 생성하고자 하는 결과물에 대해서 어떤 세부 사항에 집중할 것인지 설정할 수 있으며 생성형 AI가 취할 관점에 대해 직업, 직책, 가상의 인물 등으로 표현할 수 있다[17]. 또한 프롬프트의 맥락과 명확성은 정확도 높은 응답을 도출할 수 있다는 선행 연구를 바탕으로 프로그래밍 교육을 위한 코드 생성에 필요한 정보로 ‘개념’, ‘수준’, ‘선행 개념’의 학습자 중심 키워드를 추가하여 학습자의 단계 및 수준을 고려할 수 있도록 하였다[16-19]. 특히 ‘선행 개념’은 학생들의 인지 부담을 줄여주고 학습한 범위 내의 개념만 활용하여 생성할 수 있도록 제한하였다[3]. 완성된 시연 및 실습 단계의 예시 코드 생성 프롬프트는 Table 2와 같다.

Table 2. Demonstration and Practice Example Codes Generation Prompt

Type	Detail
Prompt 1	Be a teacher of Python programming and answer the following questions
Prompt 2	Make <i>[Number]</i> example codes for <i>[Level]</i> education using <i>[Concept]</i> base on only <i>[Precedent Concept]</i>
Example	Be a teacher of Python programming and answer the following questions Make 5 example codes for basic education using for loop base on only input and output, for loop

연습 문제 단계를 위한 프롬프트 또한 페르소나 패턴의 원리를 적용하였으며 ‘문제 생성 요청’과 ‘입력 코드 정오 확인’, ‘예시 정답 코드 생성’의 단계로 나누어 개발하였다. 학생들이 반복적인 코드 입력에서 벗어나 주어진 문제에 대해 자유롭게 사고하고 컴퓨팅 사고력을 바탕으로 실제 문제를 해결해 보는 경험을 제공해주기 위해 ‘실생활’과 ‘이야기’의 키워드를 추가하여 설계하였다[4,5,17].

또한 ‘대체 접근 방식 패턴’ 원리를 활용하여 학

생들이 문제를 해결하기 위해 작성한 코드의 유효성과 코드의 수정 방향을 요청하는 프롬프트를 구성하였으며[17], ChatGPT가 생성한 각 문제에 대한 예시 정답 코드를 요청하는 프롬프트를 통해 자신이 작성한 코드와 비교하거나 결과를 확인할 수 있도록 하였다. 완성된 프롬프트는 Table 3과 같다.

Table 3. Exercise Questions Generation Prompt

Type	Detail
Prompt 1	Be a teacher of Python programming and answer the following questions
Prompt 2	Make <i>[Number]</i> <i>[Level]</i> exercise questions <i>[Concept]</i> programming in the form of stories using real-life topics base on only <i>[Precedent Concept]</i>
Prompt 3	When I made the code for question <i>[Number]</i> as below, check whether I made the correct answer and suggest what you can improve the code <i>[Code]</i>
Prompt 4	Make an example code for question <i>[Number]</i>
Example	Be a teacher of Python programming and answer the following questions
	Make 5 basic exercise questions for loop programming in the form of stories using real-life topics base on only input, output, and for loop
	When I made the code for question 1 as below, check whether I made the correct answer and suggest what you can improve the code <i>n = int(input("Enter number of times to greet: "))</i> <i>for i in range(n):</i> <i>print("Hello, User!")</i>
	Make an example code for question 1

4.2 시연 및 실습 단계 예시 코드 생성 결과 분석

개발된 시연 및 실습 단계 예시 코드 생성 프롬프트를 ChatGPT에 입력한 결과의 일부는 Figure 1과 같

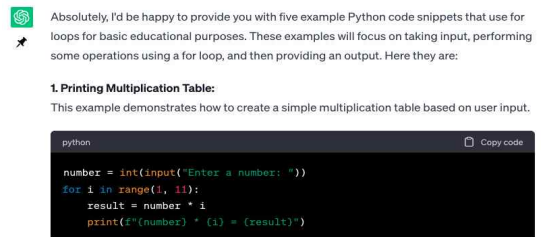


Figure 1. Result of Generating Demonstration and Practice Example Codes with Experimental Prompts

다.

생성된 예시 코드는 프롬프트의 설정에 따라 제한된 선행 개념을 활용하였으며 주요 개념을 중심으로 5개의 짧고 핵심적인 코드로만 구성된 기초 예시 코드를 생성한 것을 확인할 수 있었다.

개발된 프롬프트를 통해 생성된 코드 간의 유사도를 정량적으로 비교하기 위하여 Table 4와 같이 대조군으로 프로그래밍의 개념 정보만 제공하는 일반적인 프롬프트를 입력하여 비교 분석하였다.

Table 4. Demonstration and Practice Example Code Generation Prompt

Prompt	Detail
Experimental Prompt	<ul style="list-style-type: none"> - Be a teacher of Python programming and answer the following questions - Make 5 example codes for basic education using for loop using base on only input and output for loop
General Prompt	<ul style="list-style-type: none"> - Make 5 example codes using for loop

각 프롬프트를 통해 100회의 세션을 실행하여 500개 코드를 생성하였으며 모든 코드를 CodeBERT를 활용하여 임베딩 하였다. 코드의 유사도를 정량적으로 비교하기 위해 10개의 임의의 세션을 샘플링 하였으며 50개의 코드에 대한 임베딩 결과의 코사인 유사도 행렬을 도출하고 Fig 2, 3과 같이 시각화하였다.

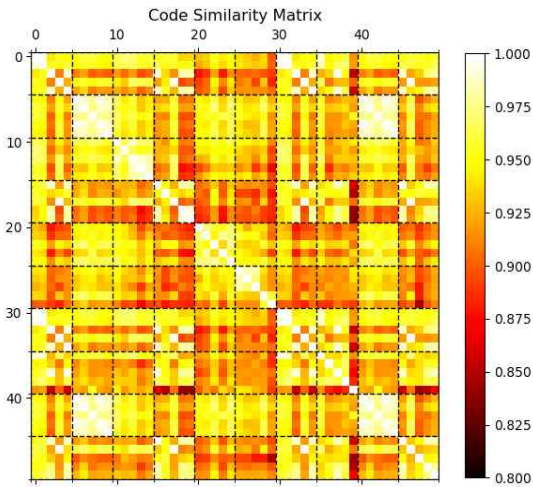


Figure 2. Demonstration and Practice Example Codes Generation Experimental Prompt Similarity Matrix

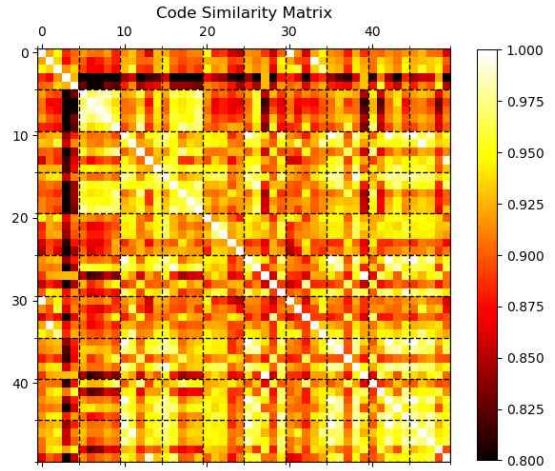


Figure 3. Demonstration and Practice Example Codes Generation General Prompt Similarity Matrix

샘플링 된 임의의 10개 세션의 경우 두 종류의 프롬프트 모두 0.8 이상의 비교적 높은 유사도를 나타내었으나, 본 연구에서 개발된 프롬프트의 경우 상대적으로 더 고른 분포의 높은 유사도 결과를 나타내었다. 특히 일반적인 프롬프트로 생성된 코드 중 유사도가 상대적으로 낮게 나타난 세션을 확인해 본 결과 파이썬과 자바스크립트, C 언어 등의 서로 다른 언어로 코드를 생성한 것을 확인할 수 있었다.

100회 세션 전체에 대한 유사도를 측정하기 위해 500개의 코드 중 임의의 코드 1개를 선택하고 생성 결과의 정확성과 교육 목적의 적합성을 확인한 뒤 기준 코드로 설정하였다. 이를 활용하여 각 세션의 코드와 코사인 유사도를 순차적으로 비교하였으며, 500개 코드에 대한 모든 결과를 도출하고 기준 코드에 대한 1개의 유사도 평가 결과를 제외한 뒤 Table 5와 같이 평균, 표준편차, 최댓값, 최솟값을 정리하였다.

Table 5. Demonstration and Practice Example Codes Generation Cosine Similarity Results

Prompt	Mean	Std	Min	Max
Experimental Prompt	0.940	0.024	0.822	0.993
General Prompt	0.932	0.026	0.754	0.995

두 프롬프트 모두 평균 0.9 이상의 높은 유사도를 보였으며 평균과 표준편차, 최댓값에서는 큰 차이를

보이지 않았다. 최솟값의 경우 일반적인 프롬프트에서는 0.754로 상대적으로 낮은 유사도를 보이는 코드가 있음을 확인할 수 있었고 개발된 프롬프트에서는 최솟값이 0.822로 비교적 높은 유사도를 보였다.

세션 간 차이점을 분석하기 위해 각 세션의 코사인 유사도 평균과 최댓값, 최솟값을 차트로 시각화한 결과는 Figure 4, 5와 같다

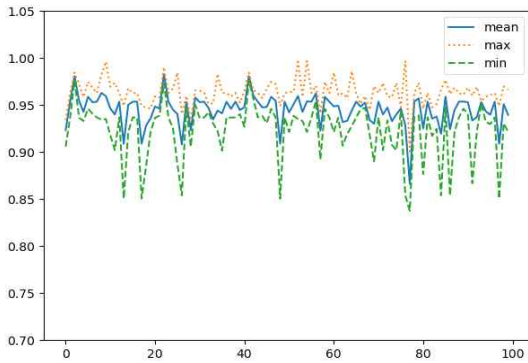


Figure 4. Visualization Result of Demonstration and Practice Example Codes Generation by Experimental Prompt each Session

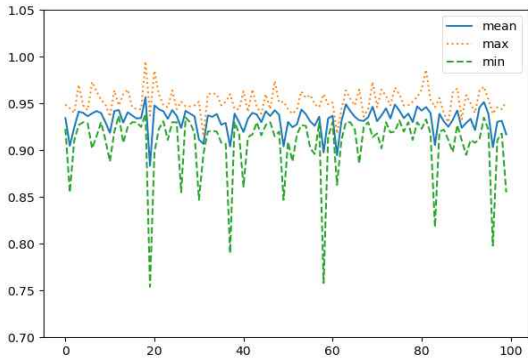


Figure 5. Visualization Result of Demonstration and Practice Example Codes Generation by General Prompt each Session

일반적인 프롬프트에서는 세션별 최솟값이 0.75에서 0.93 사이의 넓은 범위로 나타났으며 다른 세션과 큰 차이를 보이는 최솟값의 빈도도 더 자주 나타난 것을 확인할 수 있었다. 이와 비교하여 개발된 프롬프트에서는 상대적으로 고른 코사인 유사도 결과를 나타내었다. 이는 일반적인 프롬프트와 비교하여 개발된 프롬프트에서 전체적으로 기준 코드와 유사한 의미를 가지며 교육 목적에 적합한 코드를 더 안정적으로 생성하였다고 해석할 수 있었다.

4.3 연습 문제 단계 생성 결과 분석

개발한 연습 문제 생성 프롬프트를 ChatGPT에 입력하여 생성한 결과의 일부는 Figure 6과 같다.

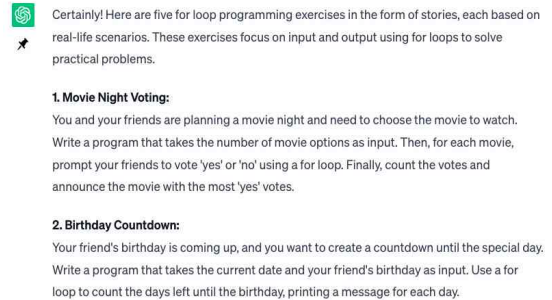


Figure 6. Results of Generating Exercise Questions with Experimental Prompts

개발된 프롬프트를 통한 연습 문제는 생활 속의 주제를 통한 이야기 형태의 문제로 제시하였으며 프롬프트에서 입력한 개념 및 선행 개념을 활용하여 연습 문제를 생성한 것을 확인할 수 있었다.

연습 문제의 유사도를 비교하기 위해 Table 6과 같이 대조군으로 일반적인 프로그래밍 개념 정보만 제공하여 연습 문제를 생성하는 프롬프트를 설정하였으며, 각 연습 문제의 예시 정답 코드를 요청하는 프롬프트는 동일하게 구성하였다.

Table 6. Exercise Questions and Codes Generation Prompt

Prompt	Detail
Experimental Prompt	<ul style="list-style-type: none"> - Be a teacher of Python programming and answer the following questions - Make 5 basic exercise questions for loop programming in the form of stories using real-life topics base on only input, output, and for loop - Make an example code for question 1-5
General Prompt	<ul style="list-style-type: none"> - Create 5 programming exercises using for loop - Make an example code for question 1-5

연습 문제의 정량적 유사도 평가를 위해 ChatGPT가 제시하는 각 연습 문제의 예시 정답 코드를 분석하며 100회의 세션, 500개의 예시 정답 코드를 CodeBERT를 활용하여 임베딩 하였다. 또한 10개의 임의의 세션을 샘플링 하여 임베딩 결과의 코사인 유사도 행렬을 도출하고 Figure 7, 8과 같이 시각화하였다.

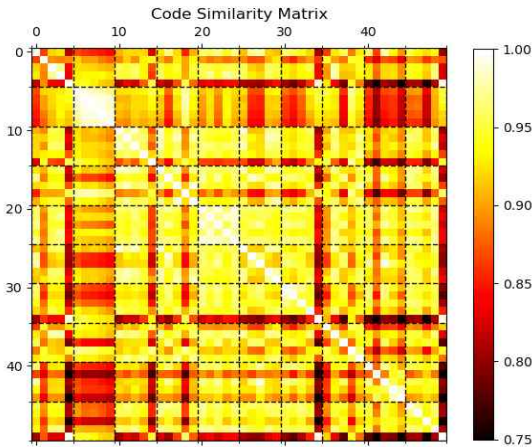


Figure 7. Experimental Prompt Exercise Questions and Codes Generation Code Similarity Matrix

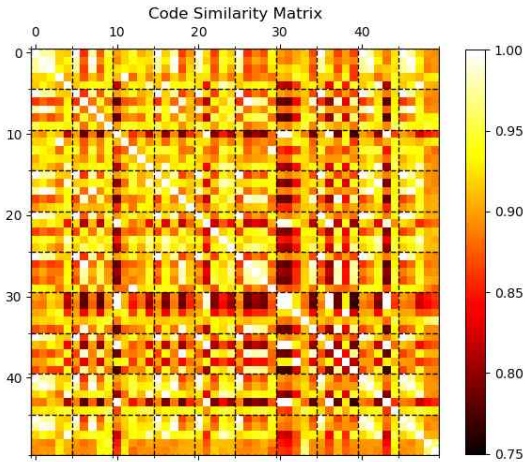


Figure 8. General Prompt Exercise Questions and Codes Generation Code Similarity Matrix

4.2 시연 및 실습 단계 예시 코드 생성 결과 분석과 비교하여 연습 문제의 예시 정답 코드는 두 프롬프트 모두 비교적 유사도가 낮은 세션이 더 많이 나타났다. 특히 일반적인 프롬프트를 통해 생성된 연습 문제의 코드는 개발된 프롬프트와 비교하여 더 많은 세션에서 낮은 유사도 결과를 나타내었다.

시연 및 실습 단계 분석과 동일한 방식으로 프롬프트를 통해 전달된 정보를 정확하게 포함하며 교육 목적에 적합한 임의의 코드 1개를 기준 코드로 설정하고 나머지 코드와의 코사인 유사도를 도출하였다. 코사인 유사도 전체의 평균, 표준편차, 최댓값과 최솟값의 결과는 Table 7과 같다.

Table 7. Exercise Questions and Codes Generation Cosine Similarity Results

Prompt	Mean	Std	Min	Max
Experimental Prompt	0.926	0.04	0.749	0.998
General Prompt	0.906	0.06	0.600	0.996

시연 및 실습 단계 예시 코드와 비교하여 전체적으로 다소 낮은 유사도 결과를 보임을 확인할 수 있었다. 특히 일반적인 프롬프트를 통해 만들어진 연습 문제 예시 정답 코드의 유사도는 최솟값이 0.600으로 매우 낮게 나타났으며, 앞선 결과와 비교하여 평균과 표준 편차 또한 더 큰 차이를 보임을 확인할 수 있었다.

세션 간 차이점을 분석하기 위해 각 세션의 코사인 유사도 평균과 최댓값, 최솟값을 차트로 시각화하였으며 결과는 Figure 9, 10과 같다.

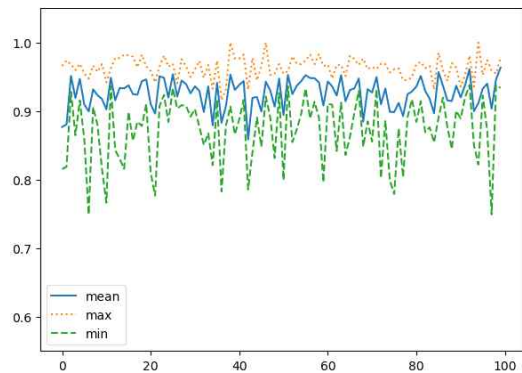


Figure 9. Visualization Result of Exercise Questions and Codes Generation by Experimental Prompt each Session

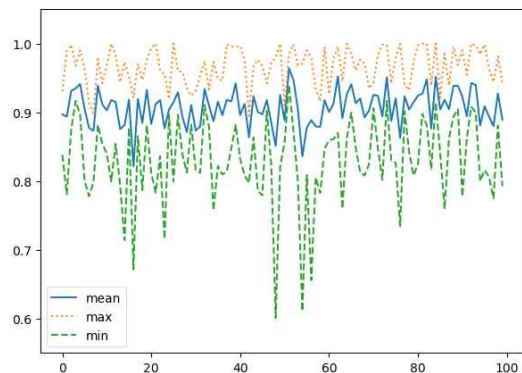


Figure 10. Visualization Result of Exercise Questions and Codes Generation by General Prompt each Session

일반적인 프롬프트와 비교하여 개발된 프롬프트는 전체적으로 높은 유사도 결과를 보이고 있었으며 특히 최솟값의 분포가 더 안정적으로 나타남을 확인할 수 있었다. 앞선 실험 결과였던 간단한 예시 코드에 대한 요청과 비교하여 연습 문제와 예시 정답 등 비교적 복잡한 결과물을 요청하는 경우 개발된 프롬프트가 훨씬 안정적이며 높은 유사도 결과를 나타내었다.

프롬프트의 차이점을 정량적으로 비교하기 위해 입력된 프롬프트의 토큰을 기준으로 확인한 결과 개발된 프롬프트의 경우 API에 117개의 토큰이 입력되었고 일반적인 프롬프트는 56개의 토큰이 입력되어 상당한 차이를 보임을 확인할 수 있었다. 이는 Polak(2023)와의 연구에서와 같이 프롬프트를 통해 맥락이 풍부한 상세 정보를 제공함으로써 더 정확한 응답 결과를 얻을 수 있었던 것으로 해석되었다[20].

5. 결론 및 제언

최근 교육 현장을 비롯한 다양한 영역에서 생성형 AI를 활용한 접근 방법들이 개발 및 활용되고 있다. 이에 본 연구는 생성형 AI를 활용하여 교육용 코드 및 교수자료를 생성하고 분석하여 프로그래밍 교육에의 적용 가능성을 확인하고자 하였다[8-10].

생성형 AI는 입력되는 프롬프트에 따라 응답의 결과가 매우 다양하게 나타나며 양질의 정보 제공을 통해 정확도 높은 출력 결과 도출이 가능함을 선행 연구를 통해 확인할 수 있었다[8,12]. 선행연구를 바탕으로 교수학습 전략과 프롬프트 엔지니어링의 원리를 적용하여 교수학습 환경에 적합한 프롬프트를 설계하였으며, 그 결과 프로그래밍 교육에 활용 가능한 예시 코드와 연습 문제를 생성할 수 있음을 확인하였다. 그리고 개발된 프롬프트의 성능을 정량적으로 평가하기 위해 예시 코드와 연습 문제 정답 코드의 유사도를 도출하였으며 일반적인 프롬프트와 비교하여 전체적으로 높은 유사도를 나타냄을 확인하였다[16-18].

본 연구에서 개발된 프롬프트를 활용하여 생성된 코드는 학습자에게 다양한 형태의 유사도 높은 코드를 제공하여 교실 전체의 실습 환경을 개선하는데 활용될 수 있을 것이다. 또한 프롬프트에 입력되는 키워드의 변경을 통해서 개별 학습자의 수준과 환경에 적합한 코드와 연습 문제를 생성할 수 있어 수준별 학습 제공의 어려움을 해결하고 학습자에게 다양한 문제 해결 경험, 컴퓨팅 사고력 중심의 교수학습 자료 제공에도 도움을 줄 수 있을 것으로 예상된다[3,4,5,21].

본 연구는 생성형 AI를 프로그래밍 교육 교수학습 환경에 적용할 수 있도록 교육 현장에 적합한 프롬프트를 개발하고 생성된 코드의 결과를 정량적으로 측정하여 활용 가능성을 확인하였다는 점에서 의의가 있다. 본 연구의 한계와 제안은 다음과 같다.

첫째, ChatGPT로 한정하여 연구를 진행하여 교육 대상의 연령에 따라 모든 교육 현장에서 활용할 수 없다는 한계를 가진다[10]. ChatGPT는 현재 가장 많은 사용자를 가지고 있으며 뛰어난 성능으로 널리 활용되고 있지만 교육 목적으로 활용하기 위해서는 교육 대상자를 고려하여 사용자 연령 제한 및 응답의 정확성, 응답의 윤리성 등에 대해 면밀한 사전 검토가 선행되어야 한다.

둘째, AI교육과 정보교육의 교육 방법 중 하나인 프로그래밍 교육에만 초점을 맞추었기 때문에 교육의 전반에서 활용하기 위한 추가적인 연구가 필요하다. 코드 생성뿐만 아니라 AI교육과 정보교육의 지식, 기능, 태도의 함양 전반에서 활용할 수 있는 방안과 다양한 학습자에게 적합한 결과를 분석할 필요가 있다.

셋째, ChatGPT와 같은 생성형 AI는 대화 혹은 세션을 기준으로 계속된 질문과 응답을 바탕으로 지속적인 맥락을 형성하고 정확도를 높일 수 있다는 특징을 가지고 있다[16-19]. 본 연구는 입력된 프롬프트를 통한 일회성 응답 결과만 활용하였기 때문에 생성형 AI에게 지속적인 맥락을 전달한 결과를 확인할 수 없었다. 추후 프롬프트의 지속적인 입력을 통한 맥락 형성과 관련된 면밀한 연구를 진행하여 그 효과성을 상세히 분석할 필요가 있다.

생성형 AI는 새로운 AI 도구로서 교육 현장에 다방면으로 활용될 수 있는 가능성을 가지고 있다. 특히 다양한 수준을 가지는 학습자에게 개별 교육이 필요한 AI 교육과 정보교육에서 교수학습 환경에 획기적인 개선을 가져다줄 수 있을 것으로 예상된다. 본 연구의 결과가 생성형 AI를 활용하여 프로그래밍 실습 환경을 포함하는 AI교육 및 정보교육 환경을 개선하는데 도움이 될 수 있기를 기대한다.

참고문헌

- [1] Korean Government. (2022). 2nd AI Strategy Summit. <https://www.msit.go.kr/bbs/view.do?sCode=user&mlId=113&mPid=112&pageNo=94&rtSeq=318123&searchOpt=AI&searchTxt=>

- [2] Harii, W. (2023). Unlocking the Potential of ChatGPT: A Comprehensive Exploration of its Applications, Advantages, Limitations, and Future Directions in Natural Language Processing. *arXiv preprint arXiv:2304.02017*.
- [3] Choi, J., Lee, Y. (2014). The analysis of learners' difficulties in programming learning. *The Journal of Korean Association of Computer Education*, 17(5), 89-98.
- [4] Kim, J., Han, S., Kim, S., Jung, S., Yang, J., Jang, E., Kim, J. (2015). *A Research on the Development of Teaching and Learning Models for SW Education*. KEDI, Korea.
- [5] Kim, K. (2015). A Self-regulated Learning Model Development in Computer Programming Education. *Journal of The Korean Association of Information Education*, 19(1), 21-30
- [6] Baidoo-Anu, D., & Owusu Ansah, L. (2023). Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning. *Available at SSRN* 4337484.
- [7] Gui, J., Sun, Z., Wen, Y., Tao, D., & Ye, J. (2021). A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Transactions on Knowledge and Data Engineering*. DOI: 10.1109/TKDE.2021.3130191.
- [8] Liu, C., Bao, X., Zhang, H., Zhang, N., Hu, H., Zhang, X., & Yan, M. (2023). Improving ChatGPT Prompt for Code Generation. *arXiv preprint arXiv:2305.08360*.
- [9] Adamson, V., & Bägerfeldt, J. (2023). *Assessing the effectiveness of ChatGPT in generating Python code*.
- [10] OpenAI (2022), *'Introducing chatgpt'*. Accessed on: 18-April-2023. URL: <https://openai.com/blog/chatgpt>
- [11] Tian, H., Lu, W., Li, T. O., Tang, X., Cheung, S. C., Klein, J., & Bissyandé, T. F. (2023). Is ChatGPT the Ultimate Programming Assistant--How far is it?. *arXiv preprint arXiv:2304.11938*. ISO 690
- [12] Yetiştirilen, 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*.
- [13] Biswas, S. (2023). Role of ChatGPT in Computer Programming: ChatGPT in Computer Programming. *Mesopotamian Journal of Computer Science*, 2023, 8-16.
- [14] Cao, J., Li, M., Wen, M., & Cheung, S. C. (2023). A study on prompt design, advantages and limitations of chatgpt for deep learning program repair. *arXiv preprint arXiv:2304.08191*.
- [15] Jiang, E., Olson, K., Toh, E., Molina, A., Donsbach, A., Terry, M. & Cai, C. J. (2022), *Promptmaker: Prompt-based prototyping with large language models*, in *'Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems'*, CHI EA '22, Association for Computing Machinery, New York, NY, USA. URL: <https://doi.org/10.1145/3491101.3503564>
- [16] Giray, L. (2023). Prompt Engineering with ChatGPT: A Guide for Academic Writers. *Annals of Biomedical Engineering*, 1-5. ISO 690
- [17] White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*.
- [18] Avila-Chauvet, L., Mejía, D., & Acosta Quiroz, C. O. (2023). *Chatgpt as a support tool for online behavioral task programming*. Available at SSRN 4329020.
- [19] Chen, E., Huang, R., Chen, H. S., Tseng, Y. H., & Li, L. Y. (2023). GPTutor: a ChatGPT-powered programming tool for code explanation. *arXiv preprint arXiv:2305.01863*.
- [20] Polak, M. P., & Morgan, D. (2023). Extracting Accurate Materials Data from Research Papers with Conversational Language Models and Prompt Engineering—Example of ChatGPT. *arXiv preprint arXiv:2303.05352*.
- [21] Piteira, M., & Costa, C. (2013, 7). *Learning computer programming: study of difficulties in learning programming*. In Proceedings of the 2013 International Conference on Information Systems and Design of Communication, Lisboa. ACM
- [22] Feng, Z., Guo, D., Tang, D., Duan, N., Feng, X., Gong, M., Shou, L., Qin, B., Liu, T., Jiang, D., & Zhou, M. (2020). Codebert: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155*.



김슬기

2008년 경인교육대학교
초등교육학과(교육학학사)
2016년 경인교육대학교
융합교육학과(교육학석사)
2023년 한국교원대학교
컴퓨터교육과(교육학박사)

2008년 ~ 현재 경기도 교육청 초등교사
관심분야: 정보(SW·AI) 교육, 프로그래밍 교육, 컴퓨팅사고력,
데이터 과학, 데이터 리터러시
E-Mail: tmfrlska85@gmail.com

부 록

〈표 2〉 시연 및 실습 예시 코드 생성 프롬프트

종류	상세 사항
프롬프트 1	파이썬 프로그래밍을 가르치는 선생님이 되어서 다음 질문에 답해줘
프롬프트 2.	[개념]을 활용한 [수준] 교육용 예시 코드를 오직 [선행 개념]에 기초해서 [숫자]개 만들어줘
예시1	파이썬 프로그래밍을 가르치는 선생님이 되어서 다음 질문에 답해줘 for loop를 활용한 기초 교육용 예시 코드를 오직 입력과 출력, for loop에 기초해서 5개 만들어줘
예시2	파이썬 프로그래밍을 가르치는 선생님이 되어서 다음 질문에 답해줘 if를 활용한 상급 교육용 예시 코드를 오직 입력과 출력, for loop, while, if에 기초해서 10개 만들어줘

〈표 3〉 연습 문제 생성 프롬프트

종류	상세 사항
프롬프트 1	파이썬 프로그래밍을 가르치는 선생님이 되어서 다음 질문에 답해줘
프롬프트 2.	[개념] 프로그래밍 [수준] 연습 문제를 오직 [선행 개념]에 기초해서 실생활 속 주제를 활용한 이야기 형태로 [숫자]개 만들어줘
프롬프트 3.	문제 [숫자]의 코드를 아래와 같이 작성했을 때, 정답이 맞는지 확인하고 코드를 개선할 수 있는 부분은 무엇인지 제안해줘 [Code]
프롬프트 4.	[숫자]번 문제의 정답 예시 코드를 작성해줘
예시1	파이썬 프로그래밍을 가르치는 선생님이 되어서 다음 질문에 답해줘 for loop 프로그래밍 기초 연습 문제를 오직 입력, 출력 for loop에 기초해서 실생활 속 주제를 활용한 이야기 형태로 5개 만들어줘 문제 1의 코드를 아래와 같이 작성했을 때, 정답이 맞는지 확인하고 코드를 개선할 수 있는 부분은 무엇인지 제안해줘 n = int(input("Enter number of times to greet: ")) for i in range(n): print("Hello, User!") 1번 문제의 정답 예시 코드를 작성해줘

	파이썬 프로그래밍을 가르치는 선생님이 되어서 다음 질문에 답해줘
예시2	리스트 프로그래밍 중급 연습 문제를 오직 입력, 출력 for loop, if에 기초해서 실생활 속 주제를 활용한 이야기 형태로 3개 만들어줘 문제 2의 코드를 아래와 같이 작성했을 때, 정답이 맞는지 확인하고 코드를 개선할 수 있는 부분은 무엇인지 제안해줘 def find_fruit(store_fruits, jinhee_favorite): for fruit in store_fruits: if fruit == jinhee_favorite: return "있어요" return "없어요" store_fruits = ['사과', '바나나', '포도'] jinhee_favorite = '바나나' result = find_fruit(store_fruits, jinhee_favorite) print(result) 2번 문제의 정답 예시 코드를 작성해줘



물론이지, Python 프로그래밍의 기초를 이해하기 위한 for 루프 예제 5개를 제공해 드릴게요. 이 예제들은 입력을 받고, for 루프를 사용하여 몇 가지 작업을 수행한 후 결과를 출력하는 것에 중점을 둔 예제입니다. 이제 시작해볼까요?

1. 구구단 출력:

사용자로부터 숫자를 입력받아 해당 숫자의 구구단을 출력하는 프로그램입니다.

```
python
number = int(input("숫자를 입력하세요: "))
for i in range(1, 10):
    result = number * i
    print(f"{number} * {i} = {result}")
```

〔그림 1〕 개발된 프롬프트를 활용한 시연 및 실습 단계 예시 코드 생성 결과



물론이에요! 다음은 실생활 상황을 기반으로 한 다섯 가지 for 루프 프로그래밍 연습 문제입니다. 각 문제는 입력 및 출력을 중심으로 하며, 실제 문제를 해결하기 위해 for 루프를 사용하는 내용을 다룹니다.

1. 영화 밤 투표:

당신과 친구들은 영화 밤을 계획 중이며, 볼 영화를 선택해야 합니다. 영화 옵션의 개수를 입력으로 받는 프로그램을 작성하세요. 그런 다음, 각 영화마다 for 루프를 사용하여 친구들에게 '예' 또는 '아니오'로 투표하도록 안내하세요. 마지막으로 투표 수를 계산하고 '예' 투표가 가장 많은 영화를 발표하세요.

2. 생일 카운트다운:

친구의 생일이 다가오고 있으며, 그 특별한 날까지 카운트다운을 만들고 싶습니다. 현재 날짜와 친구의 생일을 입력으로 받는 프로그램을 작성하세요. for 루프를 사용하여 생일까지 남은 날을 세고 매일 메시지를 출력하세요.

〔그림 6〕 개발된 프롬프트를 활용한 연습 문제 생성 결과