


02. 프로그래밍에 대한 이해



목차

- 프로그래밍
 - 프로그래밍
 - 프로그래밍 언어
 - 코딩과 프로그래밍
- 도구
 - 여러 도구들
 - LLM의 활용

프로그래밍

프로그래밍

프로그램 program

+ 단어장 저장

형태 [[영어]program]

표준국어대사전

고려대한국어대사전

우리말샘



예문 열기

명사

1. **기본의미** 일의 진행 계획이나 순서. 또는 그 목록.

▼ 연수 프로그램

2. 방송이나 공연 따위의 진행 차례나 진행 목록.

▼ 텔레비전 프로그램

유의어 **프로²**

3. 전산 어떤 문제를 해결하도록 컴퓨터에 주어지는 자료 처리 방법과 순서를 기술한 일련의 명령문의 집합체.

▼ 컴퓨터 프로그램

유의어 **프로²**

프로그래밍

- 프로그램(Program)
 - 업무를 수행하기 위한 절차
- 프로그래밍(Program + -ing)
 - 어떤 목적을 달성하기 위해 필요한 절차를 기술하는 것

프로그래밍 언어

- 프로그래밍 언어(Programming Language)
 - 컴퓨터에게 명령을 내릴 수 있는 도구
 - 자연어와 흡사
 - **문법(Syntax):** **소통하기 위한 규칙**으로 자연어와는 다르게 이를 지키지 않으면 컴퓨터와 소통할 수 없다.
- 구조
 - **코어(Core):** **문법**과 **동작 원리**를 의미
 - **라이브러리(Library):** 프로그래머가 사용할 수 있는 **도구**
- 앞으로 여러분이 배워야 할 것은…
 1. 나의 **의도를 명확히 반영**하기 위해 어떤 문법을 써야 하는지
 2. 유용한 라이브러리는 무엇이고, **어떻게 사용**하는지

코딩과 프로그래밍

- 코드(Code)
 - 문법에 맞춰 적어놓은 명령어
- 코딩(Code + -ing)
 - 코드를 작성하는 것
- 프로그래밍과 코딩의 차이점
 - **프로그래밍:** 목적을 달성할 수 있게 논리적인 절차를 기술(**생각**)하는 것
 - **코딩:** 위의 절차를 바탕으로 컴퓨터가 알 수 있는 명령어로 옮기는 것

도구

여러 도구들

- **코드 에디터(Code Editor):** 코드 작성
 - e.g. [메모장](#), [Vim](#), [Visual Studio Code](#), [Sublime Text](#) 등
- **컴파일러(Compiler):** 코드를 컴퓨터가 이해할 수 있는 명령어로 변환[1]
 - e.g. [Roslyn](#), [Mono C# Compiler](#), [IL2CPP](#) 등
- **디버거(Debugger):** 버그를 쉽게 찾을 수 있도록 프로그램 중단 및 재개 지원
- **포매터(Formatter):** 코드 스타일 및 형식 통일하여 가독성을 높임
 - e.g. [clang-format](#), [Prettier](#), [StyleCop](#) 등
- **린터(Linter):** 잠재적 오류 및 비효율적 코드 감지
 - e.g. [clang-tidy](#), [Roslynator](#) 등
- **언어 서버(Language Server):** 코드 자동 완성, 정의 이동 등의 기능 지원[2]
 - e.g. [clangd](#), [IntelliSense](#) 등

[1]: 이를 [컴파일\(Compile\)](#)이라고 한다.

[2]: 자세한 것은 [Language Server Protocol](#) 참고

여러 도구들

- **프로파일러(Profiler):** 프로그램의 성능을 측정[1]
 - e.g. [gprof](#), [Valgrind](#), [perf](#), [dotTrace](#), [dotMemory](#) 등
- 통합개발환경(IDE; Integrated Development Environment)
 - 여러 프로그래밍 도구를 하나의 프로그램으로 통합시킨 것
 - e.g. [Android Studio](#), [IntelliJ IDEA](#), [Xcode](#), [Visual Studio](#), [Rider](#) 등

LLM의 활용

- 현대 개발에서 AI는 필수불가결
 - 기업 자소서에서도 AI 활용에 대해서 묻고 있음(e.g. NCSoft, Com2us 등)
- AI를 어떻게 활용하냐에 따라 생산성이 달라짐
- 올바르게 사용하지 못하면 대체될 뿐
 - 특히, 프로그래머는 사고력이 높아야하고 이를 향상하기 위해 끊임없이 사고해야 함
 - AI에게 사고를 넘기는 잘못을 범하면 안 됨. 다시 말해 사고의 주체는 항상 나여야 함
 - AI로부터는 사고하기 위한 지식을 건네받거나, 내가 보지 못하는 영역을 찾는데 사용해야 함

LLM의 활용

- 사람은 설계자이자 최종 결정권자가 되어야 한다.
 - 해야 할 일
 - 시스템 아키텍처 설계와 큰 그림 그리기
 - 핵심 알고리즘과 로직 구현
 - 품질 기준 설정과 최종 검토
 - 하지 말아야 할 일
 - 단순 반복 코드 직접 작성
 - 자동화 가능한 작업 수동 진행

LLM의 활용

- AI는 조수로 활용해야 한다.
 - 해야 할 일
 - 설계에 따른 코드 구현
 - 테스트 케이스 작성
 - 문서 초안 생성
 - 리팩토링 제안
 - 하지 말아야 할 일
 - 아키텍처 결정
 - 알고리즘 선택
 - 최종 품질 판단

부록

더 나아가기

- 프로그래밍이란 무엇인가요?
- 앞으로 수업을 들으면서 어떤 것을 중점적으로 배워나가야 할까요?
- 이전에 코딩과 프로그래밍을 동일한 것으로 생각했나요? 수업을 듣고 난 뒤, 생각이 어떻게 바뀌었나요?
- 앞으로 AI에게 맡길 부분과 맡기지 않을 부분은 무엇인가요?

참고자료

- [한 권으로 읽는 컴퓨터 구조와 프로그래밍 | 조너선 스타인하트 - 교보문고](#)
- [Human Resource Machine on Steam](#)