

# 씽크 파이썬: 컴퓨터 과학자처럼 생각하며 배우는 파이썬. 제3판



## 1. 강사: 여봉 (呂鵬, Lyndon(En))

### 1.1. 기본 정보

여봉입니다. '씽크 파이썬' 강좌의 강사를 맡게 되었다. 현재 전주대학교에서 인공지능 박사 과정을 밟고 있음. 저의 배경에는 프로그래밍, AI 연구 및 교육에 관한 풍부한 경험이 있음.



### 1.2. 학력 배경

- 인공지능 박사 (진행 중) - 신경구조 재구성: SSM, MoE 및 신경미분방정식 3중 융합 아키텍처의 시스템 설계 및 대규모 AI 시스템에서의 적용
- 인공지능 석사 - 기계학습 알고리즘 및 데이터 마이닝 중점
- 소프트웨어 공학 학사 - 프로그래밍 원리와 소프트웨어 개발에 강한 기반 구축

### 1.3. 기술 전문 분야

- 프로그래밍 언어: 파이썬 (전문), C, C++, 자바, 안드로이드, Objective-C
- AI 및 머신러닝: 딥러닝, 컴퓨터 비전, NLP, 데이터 마이닝, 강화학습
- 연구 분야: LLM 개인화, 빅 데이터 분석, 스마트 시스템 개발 etc.

### 1.4. 교육 철학

저는 이론과 실제 응용을 연결하는 실습 중심 학습을 믿는다. 이 강좌를 통해 파이썬 프로그래밍 기초와 함께 문제 해결 능력과 컴퓨터적 사고 개발에 집중할 것이다.

### 1.5. 연락처 정보

- 전화: 010-9601-9911
- 연구실 위치: 공학관 1호관 5층 514호 (입구에서 두 번째 자리)
- 이메일: [lyupaif@jj.ac.kr](mailto:lyupaif@jj.ac.kr) (<mailto:lyupaif@jj.ac.kr>)
- 연구실 시간: 오전 8시부터 오후 10시까지
- 강의 자료: 모든 자료는 이 주피터 노트북 파일을 통해 제공될 예정임.

\*\*여러분의 파이썬 학습 여정을 안내하게 되어 기쁩니다. 프로그래밍은 단순히 코드를 작성하는 것이 아니라 새로운 사고 방식을 개발하는 것입니다!\*\*

## 2. 교재 소개: 씽크 파이썬. 제3판

O'REILLY®

Third  
Edition

# Think Python

How to Think Like a Computer Scientist



Allen B. Downey

## 2.1. 왜 "컴퓨팅사고와 SW코딩"인가?

오늘날 빠르게 진화하는 디지털 환경에서 컴퓨팅적 사고는 모든 분야에서 기본적인 기술이 되었다. "컴퓨팅사고와 SW코딩" 강좌는 단순히 프로그래밍 능력뿐만 아니라, 학생들이 체계적으로 문제에 접근하고 효과적인 해결책을 만들 수 있게 하는 필수적인 인지 프레임워크를 개발하는 것을 목표로 한다.

컴퓨팅적 사고는 코딩을 넘어서는 중요한 기술들을 포함한다:

- 복잡한 문제를 관리 가능한 구성요소로 분해하기
- 패턴과 추상화 인식하기
- 알고리즘과 단계별 절차 설계하기

# 씹크 파이썬

THINK  
PYTHON

앨런 다우니 지음  
조현태 옮김

개정 2판



- 테스트와 개선을 통한 해결책 평가하기

이러한 능력은 과학과 공학에서부터 비즈니스, 의학, 인문학에 이르기까지 사실상 모든 전문 분야에서 점점 더 가치가 높아지고 있다. 컴퓨팅적 사고를 마스터함으로써, 학생들은 학업과 전문적 경력 전반에 걸쳐 활용할 수 있는 전이 가능한 문제 해결 접근법을 개발하게 된다.

## 2.2. 왜 파이썬을 교육 언어로 선택했는가?

여러 설득력 있는 이유로 파이썬을 교육 언어로 선택했다:

1. **초보자를 위한 접근성:** 파이썬의 깔끔한 구문과 가독성은 새로운 프로그래머들에게 예외적으로 접근하기 쉽게 만들어, 학생들이 복잡한 구문보다는 개념에 집중할 수 있게 한다.
2. **산업적 관련성:** 파이썬은 전 세계적으로 가장 수요가 많은 프로그래밍 언어 중 하나로, 특히 데이터 과학, AI, 웹 개발, 자동화와 같은 빠르게 성장하는 분야에서 두각을 나타낸다.
3. **다재다능함:** 웹 애플리케이션부터 과학적 컴퓨팅까지, 파이썬의 광범위한 라이브러리 생태계는 학생들이 관심사에 기반한 다양한 응용 분야를 탐색할 수 있게 한다.
4. **커뮤니티 지원:** 파이썬의 방대한 커뮤니티는 모든 수준의 학습자들에게 풍부한 자원, 문서, 지원을 제공한다.
5. **현대적 도구와의 통합:** 파이썬은 가상 비서 및 대화형 컴퓨팅 환경을 포함한 현재 기술과 원활하게 작동하여 학습 경험을 향상시킨다.

## 2.3. 왜 "씹크 파이썬: 컴퓨터 과학자처럼 생각하며 배우는 파이썬"인가?

앨런 B. 다우니(Allen B. Downey)의 "Thinking Python: How to think like a computer scientist. 3rd Edition"은 여러 핵심적인 이유로 우리 강좌 목표와 완벽하게 일치한다:

1. **컴퓨팅적 사고에 중점:** 이 책은 단순히 구문을 가르치는 대신, 우리 강좌 제목이 약속하는 것처럼 프로그래밍 뒤에 있는 사고 과정을 강조합니다. 코딩을 배우는 동안 "컴퓨터 과학자처럼 생각하도록" 학생들을 훈련시킵니다.
2. **점진적 학습 경로:** 교재의 신중하게 구성된 접근법은 객체 지향 프로그래밍과 같은 더 복잡한 주제로 나아가기 전에 기본 개념부터 시작하여 지식을 점진적으로 구축한다.
3. **정확한 용어:** 저자는 프로그래밍 어휘를 세심하게 소개하고 정의하여, 학생들이 코드를 작성하고 효과적으로 소통하는 데 필요한 이중 유창성을 개발하도록 돕는다.
4. **실습 중심 접근법:** 각 장은 개념을 적용을 통해 강화하는 목표 지향적 연습 문제를 포함하여, 학생들이 이론적 이해와 함께 실용적인 기술을 개발하도록 보장한다.
5. **현대적 도구의 통합:** 3판은 특히 주피터 노트북과 ChatGPT와 같은 가상 비서 사용에 대한 지침을 통합하여 우리의 교육 방법론과 완벽하게 일치한다.
6. **테스트 강조:** 이 책은 초기에 소프트웨어 테스트 원칙을 소개하여 처음부터 좋은 개발 관행을 기르게 한다.
7. **간결한 표현:** 저자의 명확하고 효율적인 글쓰기 스타일은 인지적 부하를 최소화하여, 학생들이 프로그래밍 개념을 마스터하는 데 정신적 에너지를 집중할 수 있게 한다.

## 2.4. 학생들이 얻게 될 것

"파이썬으로 생각하기"를 통해 이 강좌를 완료함으로써, 학생들은 다음을 개발하게 된다:

- 다재다능하고 수요가 많은 언어인 파이썬의 기본적인 프로그래밍 기술
- 다양한 분야에 적용 가능한 체계적인 문제 해결 능력
- [주피터 노트북과 AI 비서 \(00 2. ChatGPT, Claude 등 대규모 언어 모델의 완벽한 쉬운 해석.ipynb#LLMs section\)](#)를 포함한 현대적인 개발 도구 활용 능력
- 소프트웨어 테스트 원칙과 관행에 대한 이해

## 3. 강의 계획서: 컴퓨팅사고와 SW코딩

### 3.1. 교재 개요

이 강좌는 앨런 B. 다우니(Allen B. Downey)의 "씹크 파이썬. 제3판"을 기반으로 하며, 교재는 프로그래밍 기술과 컴퓨팅적 사고 모두에 훌륭한 기초를 제공한다. 원래 교재는 기본 표현식부터 고급 객체지향 원리까지 프로그래밍 개념을 점진적으로 소개하는 19개 장으로 구성되어 있다.

## 3.2. 강좌 재구성 및 최적화

프로그래밍 경험이 없는 학생들을 위해 학습 경험을 최적화하도록 교재 내용을 신중하게 재구성했다. 수정된 15회 강좌는 필수 개념을 유지하면서 더 점진적인 학습 곡선을 만든다.

### 구조적 조정

#### 1. 장 재구성:

- 원래 3장은 함수 개념을 더 점진적으로 소개하기 위해 2회차와 3회차로 나누어졌습니다
- 18장(파이썬 추가 기능)의 선택된 내용이 튜플 교육을 보완하기 위해 10회차로 앞당겨졌다
- 16장은 더 체계적인 OOP 설명을 위해 14회차와 15회차로 나누어졌다

#### 2. 내용 향상:

- 수학적 예제는 프로그래밍 배경이 없는 학생들을 더 잘 수용하기 위해 단순화되었다
- 흥미를 높이기 위해 추가적인 실제 응용 사례가 통합되었다
- 재귀와 같은 도전적인 개념은 학생들이 적절한 기초 지식을 갖추도록 재배열되었다

#### 3. 실용적 개선:

- 각 세션에는 이제 목표지향적인 실습 연습이 포함된다
- 연습 순서는 간단한 것에서 복잡한 것으로 진행되어 학생들이 점진적으로 기술을 쌓을 수 있게 한다
- 복잡한 프로젝트는 관리 가능한 작업으로 나누어졌다

#### 4. 난이도 진행:

- 단원 1 (1-5회차): 기본 개념에 초점, 낮은 난이도
- 단원 2 (6-9회차): 데이터 구조 소개, 중간 난이도
- 단원 3 및 4 (10-15회차): 이전 지식을 기반으로 하는 더 복잡한 개념 제시, 중간 난이도 유지

## 3.3. 강좌 개요

### 단원 1: 프로그래밍 기초 (1-5회차)

#### 1회차: 컴퓨팅적 사고와 파이썬 기초 소개

- 컴퓨팅적 사고 개념
- 파이썬 환경 설정
- 기본 표현식과 문장
- 주피터 노트북 소개

#### 2회차: 변수, 표현식 및 함수 기초

- 변수와 할당
- 숫자 유형과 연산
- 간단한 함수 정의
- 내장 함수 사용하기

#### 3회차: 함수 설계 및 인터페이스

- 함수 매개변수와 인자
- 함수 구성
- 프로그램 구조와 흐름
- 문서화 및 테스트 기초

#### 4회차: 조건문과 재귀

- 부울 표현식

- 조건부 실행
- 재귀 소개
- 간단한 재귀 패턴

### **5회차: 반환 값과 함수 설계**

- 함수 반환 값
- 함수 설계 원칙
- 스택 다이어그램
- 점진적 개발

## **단원 2: 시퀀스와 반복 (6-9회차)**

### **6회차: 반복, 루프 및 검색**

- While과 for 루프
- 루프 패턴
- 검색 알고리즘
- 루프 디버깅

### **7회차: 문자열 연산과 정규 표현식**

- 문자열 메서드와 연산
- 문자열 형식 지정
- 패턴 매칭
- 정규 표현식 기초

### **8회차: 리스트와 시퀀스 연산**

- 리스트 생성과 접근
- 리스트 메서드와 연산
- 리스트 컴프리헨션
- 중첩 리스트

### **9회차: 딕셔너리와 매핑**

- 딕셔너리 생성과 접근
- 딕셔너리 메서드
- 해시 테이블과 성능
- 고급 딕셔너리 기법

## **단원 3: 데이터 구조와 파일 (10-12회차)**

### **10회차: 튜플과 데이터 구조**

- 튜플 기초와 응용
- 다중 할당
- 다중 값 반환
- 데이터 구조 선택

### **11회차: 텍스트 분석 및 생성**

- 텍스트 처리 기법
- 단어 빈도 분석
- 마르코프 모델

- 텍스트 생성 알고리즘

#### **12회차: 파일과 데이터베이스**

- 파일 연산
- 데이터 지속성
- 데이터베이스 기초
- 데이터 로딩 및 저장

### **단원 4: 객체지향 프로그래밍 (13-15회차)**

#### **13회차: 클래스와 함수**

- 객체지향 개념
- 클래스 정의
- 속성과 메서드
- 객체 생성

#### **14회차: 클래스와 메서드**

- 메서드 설계
- 특수 메서드
- 연산자 오버로딩
- 객체 상호작용

#### **15회차: 클래스, 객체 및 상속**

- 상속 기초
- 다형성
- 클래스 계층
- 디자인 패턴

### **3.4. 교육 목표**

이 강좌 구조는 프로그래밍 배경이 없는 학생들의 학습 곡선을 고려하여 신중하게 설계되었다. 전략적인 콘텐츠 배분과 타이밍을 통해 학생들이 15회 세션 내에 파이썬 프로그래밍 기초와 컴퓨팅적 사고 방법론을 체계적으로 마스터할 수 있도록 보장한다.

각 세션은 이전 지식을 기반으로 하면서 관리 가능한 속도로 새로운 개념을 소개한다. 실습 연습은 실제 적용을 통해 이론

## **4. 파이썬을 위한 교육 및 학습 도구**

### **주피터 노트북(Jupyter-Notebook): 우리의 주요 학습 환경**

이 강좌에서는 주로 주피터 노트북을 주요 교육 플랫폼으로 사용할 것이다. 주피터 노트북은 프로그래밍 초보자에게 특히 적합한 대화형 코딩 환경을 제공한다.

#### **4.1. 주피터 노트북이란?**

주피터 노트북은 다음과 같은 요소가 포함된 문서를 만들고 공유할 수 있는 오픈 소스 웹 애플리케이션이다:

- 실행할 수 있는 라이브 코드
- 풍부한 텍스트 설명
- 시각화 및 차트



- 대화형 요소

문서에 설명과 코드를 모두 작성할 수 있는 디지털 노트북이라고 생각하시면 되구요. 코드 블록을 개별적으로 실행하고 각 블록 바로 아래에서 결과를 즉시 볼 수 있다.

## 4.2. 주피터 노트북을 어떻게 사용할 것인가

이 강좌에서 주피터 노트북은 여러 목적으로 사용된다:

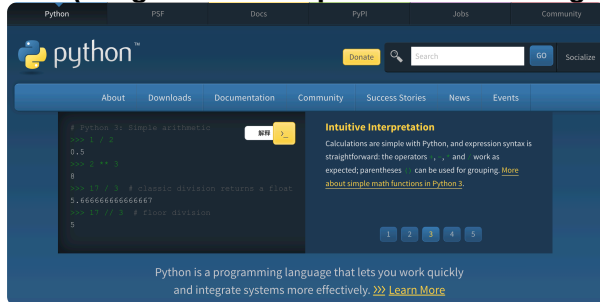
- **수업 내 시연:** 저의가 노트북을 사용하여 개념을 시연함
- **실습 연습:** 노트북에서 직접 코딩 연습
- **과제:** 노트북 파일로 과제를 제출
- **자기주도 학습:** 개념을 자신의 속도로 복습하기 위해 노트북을 반복적으로 방문할 수 있음

주피터에 대한 사전 경험이 필요하지 않습니다—첫 번째 세션에서 필요한 모든 것을 다룰 것이다.

## 4.3. 파이썬 설치

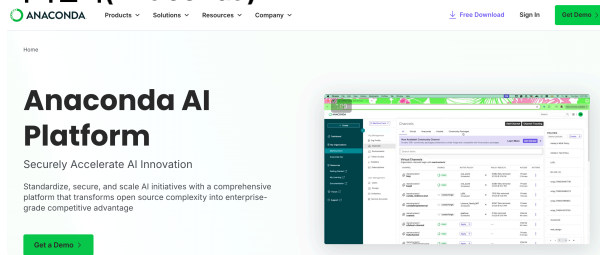
현재 주로 두 가지 설치 방법이 있다: Python 공식 웹사이트에서 다운로드해서 설치하기; Anaconda 통합 패키지 관리 플랫폼을 통해 설치하기.

### IDLE(Integrated Development and Learning Environment For Python)



- 파이썬과 함께 번들로 제공되는 기본 IDE
- 기능이 적은 간단한 인터페이스
- 매우 기본적인 프로그래밍 작업에 적합
- 다른 옵션에 비해 기능이 제한적

### 아나콘다(Anaconda)

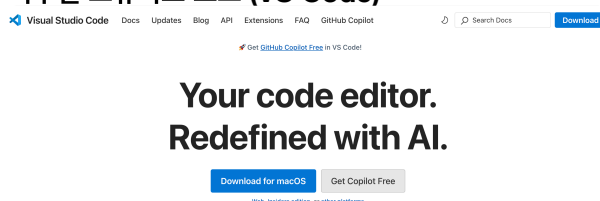


- 쉽게 시작: 파이썬과 데이터 분석에 필요한 모든 프로그램을 한 번에 설치.
- 복잡함 제로: 따로 설치할 필요 없이 바로 코딩을 시작할 수 있다.
- 정리왕: 여러 프로젝트를 동시에 진행할 때 서로 꼬이지 않도록 깔끔하게 관리해 준다.

## 4.4. 기타 파이썬 개발 플랫폼

이 강좌에서는 주피터 노트북에 중점을 둘 것이지만, 파이썬 프로그래머들이 사용하는 다른 인기 있는 개발 환경을 알아두는 것이 유용하다:

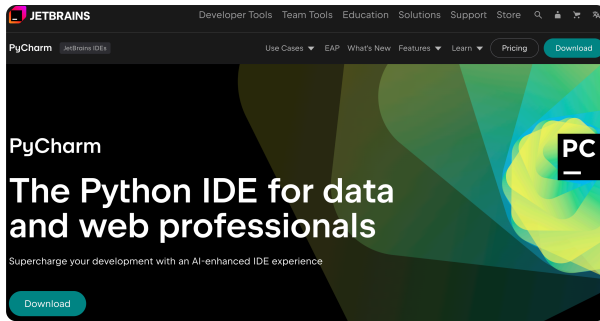
### 비주얼 스튜디오 코드 (VS Code)



- 가볍지만 강력한 코드 편집기
- 파이썬 외에도 많은 프로그래밍 언어 지원
- 추가 기능을 위한 확장 프로그램 제공
- 전문 개발자들 사이에서 인기
- 초보자를 위한 설정이 더 많이 필요

### 파이참 (PyCharm)





- 파이썬을 위해 특별히 설계된 종합적인 통합 개발 환경 (Integrated Development Environment, IDE)
- 고급 코드 완성 및 오류 감지 제공
- 내장된 디버깅 도구 포함
- 산업에서 널리 사용됨
- 초보자에게는 학습 곡선이 가파름

## 4.5. 교육 및 학습을 위해 주피터 노트북을 선택한 이유

이 강화를 위해 몇 가지 설득력 있는 이유로 주피터 노트북을 선택했다:

### 1. 초보자에게 이상적

- 낮은 진입 장벽: 복잡한 설정이 필요 없음
- 즉각적인 피드백: 코드를 실행하고 결과를 즉시 확인
- 시각적 학습: 출력(차트 및 표 포함)이 코드 바로 아래에 표시됨

### 2. 컴퓨팅적 사고 지원

- 단계별 실행: 개별 코드 셀을 실행하여 각 부분이 정확히 무엇을 하는지 확인
- 실험: 코드를 쉽게 수정하고 다시 실행하여 다른 결과 확인
- 문서화: 설명 텍스트와 코드를 결합하여 개념 강화

### 3. 교육적 이점

- 서술적 흐름: 설명과 코드를 혼합하는 교재의 접근 방식을 따름
- 혼합 콘텐츠: 텍스트, 코드, 방정식 및 시각화를 하나의 문서에 결합
- 자체 완결형 수업: 각 노트북에는 해당 수업에 필요한 모든 것이 포함됨

### 4. 실용적 이점

- 일관성: 모든 사람이 동일한 환경에서 작업하여 설정 문제 방지
- 쉬운 제출: 과제를 단일 노트북 파일로 제출 가능
- 진행 상황 추적: 작업이 진행되면서 저장되어 사고 과정을 보여줌

### 5. 산업 관련성

- 데이터 과학, 연구 및 학계에서 널리 사용됨
- 전문적인 환경으로 기술 전환
- 주피터 경험은 많은 분야에서 점점 더 가치를 인정받음

## 4.6. 시작하기

첫 번째 세션에서는 다음 사항을 안내할 것이다:

- 주피터 노트북 접근하기
- 노트북 만들기 및 저장하기
- 코드 셀 실행하기
- 텍스트 설명 추가하기
- 노트북 파일 관리하기

**연습 1:** Jupyter 노트북은 셀로 구성되어 있으며, 각 셀에는 텍스트 또는 코드가 들어 있다. 이 셀에는 텍스트가 들어 있다.

다음 셀에는 코드가 들어 있다.

```
In [1]: 1 print('Hello, world!')
```

Hello, world!

이전 셀을 클릭하여 선택한다. 왼쪽 상단 메뉴 표시줄에 원 안에 삼각형이 있는 버튼이 표시되며, 이 버튼은 '실행' 아이콘이다. 이 버튼을 누르면 Jupyter가 해당 셀에서 코드를 실행하고 결과를 표시한다.

노트북에서 코드를 처음 실행하는 경우 시작하는 데 몇 초가 걸릴 수 있다. 그리고 작성하지 않은 노트북인 경우 경고 메시지가 표시될 수 있다. 신뢰할 수 있는 출처(저를 포함해서)에서 노트북을 실행하는 경우, “계속 실행”을 누름.

'실행' 버튼을 클릭하는 대신, Shift 를 누른 상태에서 Enter 를 눌러 셀에서 코드를 실행할 수도 있다.

**연습 2:** 셀에 텍스트를 추가함. 위의 셀 메뉴를 사용하여 서식을 지정하거나 [마크다운](https://www.markdownguide.org/basic-syntax/)

(<https://www.markdownguide.org/basic-syntax/>)을 사용해 텍스트를 마크업할 수 있다. 완료했으면 Shift 를 누른 상태에서 Enter 를 누르면 방금 입력한 텍스트의 서식이 지정되고 다음 셀로 이동한다.

**연습 3:** Jupyter는 두 가지 모드를 가지고 있다:

- **명령 모드**에서는 전체 셀 추가 및 제거와 같이 셀에 영향을 주는 작업을 수행할 수 있다.
- **편집 모드**에서는 셀의 내용을 편집할 수 있다.

텍스트 셀을 사용하면 어떤 모드에 있는지 알 수 있다. 편집 모드에서는 셀이 세로로 분할되어 왼쪽에는 편집 중인 텍스트가, 오른쪽에는 서식이 지정된 텍스트가 표시된다. 그리고 상단에 텍스트 편집 도구가 표시된다. 명령 모드에서는 서식이 지정된 텍스트만 표시된다.

코드 셀의 경우 차이가 더 미묘하지만 셀에 커서가 있으면 편집 모드에 있는 것이다.

편집 모드에서 명령 모드로 이동하려면 ESC 를 누름. 명령 모드에서 편집 모드로 이동하려면 Enter 를 누름.

**연습 4:** 이제 노트북의 일반적인 기능을 알았으니 마지막으로 코드를 저장하고 작업하는 것을 항상 잊지 마시오!! 그렇지 않으면 심각한 결과를 초래할 수 있다!! 언제든지 Ctrl+S 를 사용해 저장하거나 왼쪽 상단 도구 모음에 있는 저장 기능을 사용할 수 있다.

이 도구의 목적은 여러분의 학습 여정을 지원하는 것이다. 주피터 노트북은 단순함과 기능의 완벽한 균형을 제공하여 복잡한 개발 환경과 씨름하지 않고 프로그래밍 개념을 익히는 데 집중할 수 있도록 도와준다.

## 5. 파이썬 프로그래밍 학습 결과 평가 및 출결 체크(試行, [실제 공부 상황 따라 변경할 수 있음](#) (./00 1. [수강생 평가 기준\(實行\).ipynb#Student-evaluation-criteria](#)))

### 5.1. 평가 구성 (총 100점)

#### 1. 일일 성과 (40%)

- 수업 참여: 10%
  - 수업 질문과 답변
  - 그룹 토론에 적극적인 참여
  - 수업 대화형 연습 완료
- 프로그래밍 과제: 20%
  - 4개의 소규모 프로그래밍 과제 (각 5%)
  - 각 단원 후 배정, 단원 내용과 밀접하게 관련됨

- 대부분의 학생들이 독립적으로 완료할 수 있는 중간 난이도
- 출석: 10%
  - 자세한 출석 규칙은 아래에 나열되어 있음

## 2. 중간고사 (25%)

- 8번째 수업 시간 후 실시
- 객관식 문제(40%)와 프로그래밍 문제(60%) 포함
- 시간: 90분
- 기본 구문과 간단한 알고리즘 구현에 중점

## 3. 최종 프로젝트 (35%)

- 그룹 프로젝트 (그룹당 4-5명)
- 프로젝트 제안서 (5%): 10번째 수업 시간까지 제출
- 최종 프로젝트 (25%): 15번째 수업 시간까지 제출
- 프로젝트 발표 (5%): 마지막 주에 진행, 각 그룹 5-8분

## 5.2. 출석 기준

### 1. 기본 규칙

1. 출석 방법: 전주대학교의 출결 앱 또는 교사 시스템 사용.
2. 결석 계산:

- 전체 출석: 10점
- 1-2회 결석: 9점
- 3-4회 결석: 7점
- 5-6회 결석: 5점
- 7회 이상 결석: 0점 (강좌 재수강 권장)

### 2. 인간적인 정책

1. 인증된 병가: 병원 증명서 제출 시 결석으로 간주하지 않음
2. 가족 긴급 상황: 사전 또는 사후에 통보, 학기당 최대 2회 허용
3. 지각 규칙:
  - 15분 지각: 지각으로 기록
  - 3회 지각은 1회 결석으로 간주
  - 15분 이상 지각: 결석으로 간주하지만, 계속해서 수업에 참여하도록 권장

### 3. 출석 인센티브 조치

1. 전체 출석 보상: 전체 출석한 학생들은 최종 종합 평가에서 추가 2점을 받음 (100점을 초과하지 않음)
2. 수업 문제의 답현에 참여 보너스: 출석을 보장하는 기초 위에, 수업 토론에 적극적으로 참여하면 추가 1-2점을 얻을 수 있음

## 5.3. 프로그래밍 과제 설계

### 과제 1 (5번째 수업 시간 후): 함수와 제어 흐름

- Turtle을 사용하여 간단한 재귀 패턴을 구현하는 기하학적 도형 그리기 프로그램 설계

## 과제 2 (9번째 수업 시간 후): 데이터 구조의 응용

- 주어진 텍스트에서 단어 빈도를 계산하고 다양한 방식으로 시각화하는 텍스트 분석 도구

## 과제 3 (12번째 수업 시간 후): 파일 처리 및 데이터 분석

- 파일 읽기/쓰기 및 데이터 처리 기능을 갖춘 간단한 학생 관리 시스템 설계

## 과제 4 (14번째 수업 시간 후): 객체지향 프로그래밍

- 클래스 기반의 작은 게임이나 시뮬레이션 프로그램 만들기

## 5.4. 최종 프로젝트 제안 주제[중간시험 끝난후]

1. 개인 회계 애플리케이션: 간단한 GUI 금융 추적 도구
  2. 간단한 텍스트 편집기: 기본 편집 및 형식 지정 기능 포함
  3. 강좌 관리 시스템: 학생들이 강좌, 과제 및 성적을 관리하는 데 도움
  4. 간단한 게임 개발: 단어 맞추기 게임, 행맨 또는 간단한 RPG와 같은 게임
  5. 데이터 시각화 프로젝트: 오픈 데이터 세트 분석 및 시각화
6. 게임 하스 도미니. 단어 교시 또는 지시 오저의 아가환느 데 드으

- Made by Lyndon.
- In 2025.3.1.
- Last updated on Sep 1.