1학년 파이썬 모듈 학습 프로젝트 계획

프로젝트 개요

- **대상:** 파이썬 문법 학습을 완료한 1학년 학생
- **목표:** 파이썬 모듈 사용법을 익혀 스스로 다양한 프로젝트를 진행할 수 있는 능력과 흥미를 키운다.
- **최종 목표:** Pygame 라이브러리를 활용하여 간단한 게임을 직접 만들어본다.

모듈 학습의 중요성

- 현대 개발 환경에서 AI 도구의 활용이 필수적이지만, AI 결과물을 이해하고 미세 조정하기 위해서는 기본적인 로직 및 모듈 활용 능력이 요구된다.
- 모듈 사용법을 익히면 파이썬의 강력한 기능을 활용하여 더 복잡하고 흥미로운 작업을 효율적으로 수행할 수 있다.

추천 모듈 학습 방법

1. **공식 문서 활용:** 모듈의 정확한 기능과 사용법을 익히는 가장 기본적인 방법입니다.
2. **검색 및 튜토리얼 참고:** 다양한 예제와 설명을 통해 실제 적용 방법을 학습합니다.
3. **직접 코딩하며 익히기:** 작은 예제나 미니 프로젝트를 통해 이론을 실제 코드로 구현해보는 것이 가장 중요합니다.

가상 환경 (Virtual Environment) 사용법

파이썬 프로젝트를 진행할 때 **가상 환경**을 사용하는 것은 매우 중요합니다.

가상 환경이란?

- 특정 파이썬 프로젝트만을 위한 독립적인 작업 공간입니다.
- 프로젝트별로 필요한 파이썬 패키지(모듈)들을 분리하여 관리할 수 있습니다.

왜 가상 환경을 사용해야 할까요?

- **의존성 관리:** 프로젝트 A에서는 `requests` 1.0 버전을 사용하고, 프로젝트 B에서는 `requests` 2.0 버전을 사용해야 할 경우, 가상 환경을 사용하면 각 프로젝트에 맞는 버전을 충돌 없이 설치하고 사용할 수 있습니다.
- **전역 환경 오염 방지:** 시스템에 설치된 기본 파이썬 환경에 여러 패키지를 설치하다 보면 복잡해지고 다른 프로젝트에 영향을 줄 수 있습니다. 가상 환경을 사용하면 이런 문제를 방지할 수 있습니다.
- **프로젝트 공유 용이:** 프로젝트에 사용된 패키지 목록을 쉽게 관리하고 공유하여 다른 사람이 동일한 개발 환경을 구축하기 쉽게 만듭니다.

가상 환경 사용 방법 (venv 모듈 사용)

파이썬 3.3+ 버전부터는 `venv` 모듈이 내장되어 있어 별도 설치 없이 가상 환경을 만들 수 있습니다.

1. **가상 환경 생성:** 프로젝트 폴더로 이동하여 다음 명령어를 실행합니다. `venv_name` 부분은 원하는 가상 환경 이름으로 변경하세요 (일반적으로 `.venv` 또는 `venv` 사용).

   ```
   python -m venv venv_name
   ```

   또는 (macOS/Linux)

   ```
   python3 -m venv venv_name
   ```

2. **가상 환경 활성화 (Activate):** 가상 환경을 사용하기 전에 활성화해야 합니다. 운영체제에 따라 명령어가 다릅니다.

   - **Windows:**

     ```
     venv_name\Scripts\activate
     ```

   - **macOS/Linux:**

     ```
     source venv_name/bin/activate
     ```

   활성화되면 터미널 프롬프트 앞에 가상 환경 이름이 표시됩니다 (예: `(venv_name) your_prompt>`).

3. **패키지 설치:** 가상 환경이 활성화된 상태에서 `pip`로 패키지를 설치하면 해당 가상 환경에만 설치됩니다.

   ```
   pip install matplotlib pygame
   ```

4. **가상 환경 비활성화 (Deactivate):** 가상 환경 사용을 마쳤으면 비활성화합니다.

   ```
   deactivate
   ```

프로젝트 시작 시: 해당 프로젝트 폴더로 이동하여 가상 환경을 활성화하고 작업합니다.

학습 예정 모듈

- **Matplotlib:** 시각적인 결과물을 빠르게 얻을 수 있어 학습 초기 흥미 유발에 좋습니다. 데이터 시각화의 기초를 다질 수 있습니다.
- (선택 사항) **Pandas:** 데이터 분석에 관심이 있다면 Matplotlib 학습 후 추가로 다룰 수 있습니다.
- **Pygame:** 게임 개발을 위한 라이브러리로, 앞서 배운 모듈 활용 능력을 바탕으로 최종 프로젝트를 진행합니다.

최종 프로젝트 목표

- Pygame을 사용하여 **간단한 아케이드 게임 (예: Pong 또는 Snake)** 구현을 목표로 합니다. Pygame의 핵심 요소(게임 루프, 이벤트 처리, 객체 움직임 등)를 익히는 데 집중합니다.

프로젝트 일정

- **기간:** ~ 8월 12일
- **정기 모임:** 주 1회 (매주 화요일)

상세 일정 및 계획 (예시)

목표: 8월 12일까지 Pygame을 활용한 간단한 게임의 기본 구현을 완료한다.

진행 방식: 매주 화요일 만나서 학습 내용을 공유하고, 다음 주 학습/실습 내용을 정한다. 주중에는 스스로 학습 및 실습을 진행한다.

주차별 계획:

- **1주차 (예: 7월 2일): 파이썬 모듈의 이해 및 활용 시작**
    - 모듈이란 무엇인가? 왜 사용하는가?
    - `pip`를 이용한 모듈 설치 방법
    - `import` 구문 사용법 (`import module_name`, `from module_name import function`, `import module_name as alias`)
    - 간단한 내장 모듈 (예: `math`, `random`) 사용 예제 실습
    - **주중 과제:** 몇 가지 내장 모듈을 사용해보고, `pip`로 간단한 외부 모듈 (예: `requests` - 웹 페이지 가져오기) 설치 및 사용 연습
- **2주차 (예: 7월 9일): 데이터 시각화 모듈 Matplotlib 기초**
    - Matplotlib 설치 및 기본 설정
    - `pyplot` 모듈을 이용한 기본적인 그래프 그리기 (선 그래프, 점 그래프)
    - 데이터 준비 (리스트, 넘파이 배열 등)
    - 그래프 제목, 축 이름 설정
    - **주중 과제:** 다양한 데이터를 준비하여 여러 종류의 간단한 선/점 그래프 그려보기
- **3주차 (예: 7월 16일): Matplotlib 활용 및 Pygame 소개**
    - 막대 그래프, 산점도 등 다른 그래프 유형 그리기
    - 그래프 커스터마이징 (색상, 스타일 등)
    - **Pygame 소개:** 게임 개발 라이브러리 Pygame의 특징 및 설치
    - Pygame으로 간단한 창 띄우기 예제
    - **주중 과제:** Matplotlib로 다양한 그래프를 연습하고, Pygame 설치 및 창 띄우는 코드 실행해보기
- **4주차 (예: 7월 23일): Pygame 기본 요소 학습**
    - 게임 루프(Game Loop)의 이해 (이벤트 처리, 화면 업데이트)
    - 이벤트 처리 (키보드, 마우스 입력 등)

    - 화면에 도형 및 색상 그리기 (`pygame.draw`)
    - **주중 과제:** Pygame 창에 다양한 도형을 그리고, 키보드 입력에 따라 화면에 변화를 주는 간단한 코드 작성 연습

- **5주차 (예: 7월 30일): Pygame 객체 및 움직임 구현**
    - 게임 객체 표현 방법 (클래스 활용 등)
    - 객체의 위치, 속도 관리
    - 객체 화면에 표시하기 (`blit`)
    - 객체 움직임 구현 (위치 업데이트)
    - **주중 과제:** 화면에 움직이는 객체 (예: 사각형)를 만들고, 키보드 입력으로 조종하는 연습

- **6주차 (예: 8월 6일): 간단한 게임 (Pong) 구현 시작**
    - Pong 게임의 기본 구조 설계 (패들, 공, 점수판)
    - 패들 객체 구현 및 키보드 조작 연결
    - 공 객체 구현 및 자동 움직임 구현
    - **주중 과제:** Pong 게임의 패들과 공을 화면에 띄우고 움직이게 만드는 기본 코드 작성

- **8월 12일:** 목표 달성 확인 및 프로젝트 마무리