| 교육 제목 | 데이터 기반 인공지능 시스템 엔지니어 양성 과정\_파이썬 |
| --- | --- |
| 교육 일시 | 2021년 9월 24일 |
| 교육 장소 | YGL C-6 학과장 & 자택(디스코드 이용한 온라인) |
| **교육 내용** | |
| 오전 | 혼공파(혼자 공부하는 파이썬 5장 \_ 함수   1. 함수의 활용    * 재귀함수 : 자기 자신을 호출 하는 함수  ! 상황에 따라 같은 것을 기하급수적으로 반복하는 문제 -> 메모화(딕셔너리를 사용하여 한 번 계산한 값을 저장하고 돌려 주는 방법)로 해결    * 조기리턴 : 흐름 중간에 return 키워드를 사용하는 것   !!! 파이썬은 함수 내부에서 함수 외부에 있는 변수를 참조하지 못한다. → 함수 내부에서 함수 외부에 있는 변수라는 것을 설명하려면 “global 변수이름' 구문을 사용해야 한다. (global 키워드는 파이썬에만 있음)  !!! 가독성 높은 코딩 : 주석을 활용한다. (커뮤니케이션이 원활한 코딩)  !!! 유지 보수가 쉬운 코딩 : 변수를 활용한다. 함수를 활용한다.   1. 함수 고급 (파이썬 고유 문법)    * 튜플 : 함수와 함께 많이 사용되는 리스트와 비슷한 자료형으로, 리스트와 다른 점은 한 번 결정된 요소는 바꿀 수 없다는 것(내부 요소의 변경이 불가능) !!! 괄호 없이 여러 값을 할당하는 것은 튜플만 할 수 있다.    * 람다 : 매개변수로 함수를 전달하기 위해 함수 구문을 작성하는 것이 번거롭고, 코드 공간 낭비라는 생각이 들 때 함수를 간단하고 쉽게 선언하는 방법 ex) lambda 매개변수:리턴값 → 함수의 매개변수에 직접 넣을 수 있다.  ! map() : 리스트의 요소를 함수에 넣고 리턴된 값으로 새로운 리스트를 구성해 주는 함수 ex) map(함수, 리스트) ! filter() : 리스트의 요소를 함수에 넣고 리턴된 값이 True인 것으로, 새로운 리스트를 구성해 주는 함수 ex) filter(함수, 리스트)    * 파일 처리  ! 파일 열기 : 파일 객체 = open(문자열: 파일경로, 문자열: 모드)  !!! 모두 w(write) 새로 쓰기, a(append) 뒤에 이어 쓰기, r(read) 읽기 ! 파일 쓰기 : 파일 객체.write(문자열) ! 파일 읽기 : 파일 객체.read() ! 파일 닫기 : 파일 객체.close()   !!! with 키워드 : 파일을 닫지 않는 실수를 방지하기 위해 생겨난 구문, 구문이 종료될 때 자동으로 파일이 닫힘  ex) with open(“파일명.확장자명",”모드") as file:  --(탭)-- file.write(문자열)   * + 제너레이터 : 함수 내부에 yield 키워드를 사용하면 제너레이터함수로 변환됨 (이터레이터를 직접 만들 때 사용하는 코드) ! 제너레이터 함수는 제너레이터 객체를 리턴하며, 이는 next() 를 통해서 함수 내부의 코드를 실행 → 코드를 조금씩 실행할 때 사용 |
| 오후 | 혼공파(혼자 공부하는 파이썬 6장 \_ 예외 처리   1. 구문 오류와 예외    * 프로그래밍 언어의 오류 2가지  ! 프로그램 실행 전에 발생하는 오류 → 구문 오류 ! 프로그램 실행 중에 발생하는 오류 → 런타임 오류 혹은 예외(exception)    * 구문 오류 : 괄호의 개수, 들여쓰기 문제 등으로 프로그램 실행 전에 발생하는 오류 ex) SyntaxError    * 기본 예외 처리  ! 조건문을 사용하는 방법 ! try 구문을 사용하는 방법 try: # 단독 사용 불가 --(탭)-- 예외가 발생할 가능성이 있는 코드 except:  --(탭)-- 예외가 발생했을 때 실행할 코드 (pass 사용(?)) else: # 반드시 except: 구문 뒤에 사용  --(탭)-- 예외가 발생하지 않았을 때 실행할 코드  finally: --(탭)-- 무조건 실행할 코드 2. 예외 고급    * 예외 객체를 사용하여 예외를 구분하고 if, elif, …, else 조건문 처럼 차례대로 오류 검출하면서 확인 가능 단, 조건에 일치하지 않는 예외 발생 시 프로그램이 강제 종료 되는 것을 방지하기 위해 ‘모든 예외의 부모'라고 할 수 있는 Exception을 넣어서 프로그램이 죽지 않도록 처리함    * raise 구문 : 미구현 부분으로 고의로 예외를 발생시켜 프로그램을 죽게 만들어 상기토록 하는 것  ex) raise 예외 객체 |