|  |  |
| --- | --- |
| 교육 제목 | **Python 함수의 활용** |
| 교육 일시 | 21년 9월 24일 |
| 교육 장소 | YGL |
| **교육 내용** | |
| 오전 | 재귀 함수  -재귀 함수: 내부에서 자기 자신을 호출하는 함수를 말한다.  -팩토리얼 : n\*(n-1)\*(n-2) ... n 이라는 연속된 곱셈이다.  -팩토리얼 구하기 : 반복문이나 재귀 함수로 팩토리얼을 구할 수 있는데 반복문은 간단히 10! == 10\*9\*8...1까지 반복을 하면 된다. 하지만 재귀 함수는  -메모/메모화: 딕셔너리를 활용해 한 번 계산한 값을 저장, 메모된 값을 돌려주면서 코드의 속도를 빠르게 하는 테크닉.  -global 키워드: global 변수이름으로 파이썬에만 있는 특이한 구조로 다른 구조에서도 같은 변수를 쓰기 위해 밖에서 선언해 놓은 것으로 사용할 변수 앞에 global 키워드를 붙이면 밖에 미리 선언해 놓은 변수를 쓸 수 있다.  -조기 리턴(early return): 흐름 중간에 return 키워드를 사용해서 코드 들여쓰기를 줄이는 효과를 가져오는 테크닉.  -튜플: 리스트와 비슷한 자료형으로 한번 결정된 요소를 바꿀 수 없다는 것이 리스트와 다르다. 튜플은 ()를 쓰고 리스트는[]를 쓴다.  -filter() 함수, map()함수: 함수를 매개변수로 전달하는 대표적이 함수로 내장 함수라고 부르기도 한다. 맵 함수는 리스트의 요소를 맵함수 내의 함수에 넣고 리턴된 값으로 새로운 리스트를 구해주는 함수이고, 필터함수는 리스트의 요소를 필터 함수 내의 함수에 리턴된 값이 True일때 새로운 리스트를 구해준다. 즉, 맵과 필터 함수 둘다 결과는 리스트로 나온다.  \*map함수 없이 리스트의 요소들을 함수로 결과를 내고 싶을때  for item in 리스트  함수명(item) 을 해야하지만 맵을 쓰면 map(함수명,리스트)  -람다 : 매개변수로 함수를 전달하기 위해 파이썬에서 제공하는 기능으로 lamda 매개변수: 리턴값 으로 사용한다. 간단히 말해 한 줄정도의 간단한 함수를 쉽게 선언하는 것이고, 매개변수를 여러개를 사용할 수있다. 람다가 함수가 될 수 있다.  파일처리  -open()함수  \*모드:  - w : write모드 새로 쓰기 모드, 기존의 파이링 존재하면 기존 데이터 삭제 후 새로 생성, 존재하지 않으면 새로 생성한다  -a: append 모드 뒤에 이어서 쓰기 모드, 마지막에 추가  - r: read모드 읽기 모드, 읽기만 가능하고 존재하지 않으면 작동하지 않는다  -with 구문: 파일을 open 했으면 close를 해야 자원이 낭비되는 것을 막고 다른 데에서 의도치 않게 수정되는 것을 막을 수 있는데 with 구문을 쓰면 close를 쓰지 않아도 자동으로 닫는다 with open(파일명) as file : 내용  제너레이터: 이터레이터를 직접만들 때 사용하는 코드로 함수 내부에 yield 키워드를 사용하면 해당 함수는 제너레이터 함수가 되며 일반 함수와는 달리 함수를 호출해도 함수 내부의 코드가 실행되지 않는다.  -next()함수: yield 키워드 부분까지만 실행하며 넥스트 함수 리턴값으로 yield 키워드 뒤에 입력한 값이 출력된다.  오류의 종류: 실행 하기 전에 발생하는 구문 오류(syntax error)가 있고 프로그램 실행 중에 발생하는 예외(exception) 또는 런타임 오류(runtime error)가 있다.  예외처리 : 처리 방법에는 조건문을 사용하는 방법(기본 예외 처리)과 try 구문을 사용하는 방법이 있다. |
| 오후 | -try except 구문은 오류가 일어난 상황을 완벽히 이해하고 있지 않아도 프로그램이 완전히 죽는것을 막는다.  \*마지막 부분에 pass를 넣어서 아예 넘길 수도 있고 else를 붙여서 에러가 발생하지 않았을 때를 위한 코드를 넣을 수도 있다.  -finally 구문: 예외처리 구문에서 가장 마지막에 사용할 수 있는 구문으로 예외가 발생하든 발생하지 않든 구문에 있는 코드를 무조건 실행한다. \*else 문은 except 구문 뒤에 써야한다.  -예외 잡기; 예외 처리를 할 때에 예외 를 구분을 해서 처리를 할 수가 있는데 ValueError 와 IndexError는 값에 오류가 있는 경우와 리스트이 인덱스에 관련된 예외를 알려주는 것이고 모든 예외를 알려주는 Exception이 있다.  -raise '키워드' 구문: 아직 구현인 안된 부분에서 일부러 예외를 강제로 발생시키는 구문이다.  오늘 배운것 간단히 정리  함수의 활용 : 재귀함수 -메모리  제너레이터  map(함수명, 리스트)  filter(함수명, 리스트), lambda 매개변수: 리턴값  file처리하는 함수 open close read write  예외처리  try: ~except: ~else: ~finally:  python: 자료의 형태 -수, 문자열, 불리언  list [], 딕셔너리 {}  튜플()  list[ 반복문] : 리스트 내포  조건식 if ~elif ~else  반복문 for (변수 in 반복자료):  while  break 또는 continue  fun\_a(10, 20) -> def func(a,b) : a,b가 매개변수 |