

Round 3

PRESS START





New Assignment







Let's Go





calculator.py

```
def add():
def sub():
def mul():
def div():
def rem():
```

def result_write() :
 def result_read() :

def start():

•



cal_module.py

def add:

def sub:

def div:

def mul:

cal_IO.py

def result_write():

def result_read():

calculator.py

import cal_module import cal_IO import cal_start

start()

cal_start.py

def start():



why?









Let's Go





example Java...

```
static Scanner sc = new Scanner(...);
int num1:
int num2:
try {
       num1 = sc.nextInt();
       num2 = sc.nextInt();
       System.out.println(num1/num2):
} catch (ArithmeticException e) {
       System.out.println("0으로 나누지 마!")
} catch (InputMismatchException e) {
       System.out.println("int형으로 넣어라")
} finally {
       System.out.println("try문이 끝났어.")
```



example Python...

```
try:
       num1 = int(input())
       num2 = int(input())
       print(num1/num2)
except ZeroDivisionError as e :
       print("0으로 나누지 마!")
except ValueError as e:
       print("int형으로 넣어라")
finally:
       print("try문이 끝났어.")
```





Let's Go





```
# 리스트 자료형
ex_list = [1, "이", ,[3]]

# 튜블 자료형
ex_tuple = (1, "이", [3])

# 딕셔너리 자료형
ex_dictionary = {1 : 1, 2 : "이", 3 : [3]}

# 집합 자료형
ex_set = set([1, "이", [3]])
```



List

리스트 자료형 ex_list = [1, "이", ,[3]]

요소 수정	요소 형태	특징
0	Anything	대괄호

- array(배열)과 list(리스트)의 차이

배열은 크기가 정해져 있으며, 기능이 존재하지 않는다. 리스트는 크기가 가변적이며 여러 기능이 존재한다.

배열 인덱스는 값에 대한 유일무이한 식별자. 리스트 인덱스는 몇 번째 데이터인가 정도의 의미.

배열은 요소를 삭제해도 순서가 그대로이다. 리스트는 중간 요소를 삭제하면 순서가 바뀐다.



Tuple

```
# 튜블 자료형
ex_tuple = (1, "이", [3])
```

요소 수정	요소 형태	특징
X	Anything	소괄호

- 멀쩡한 리스트를 두고 튜플은 왜 쓰는거지??

들어 있는 요소의 수를 사전에 정확히 알고 있으면 튜플이 유리하다.

리스트에 비해 조금 더 공간 효율적이다.

•••

그렇다고 한다…



Dictionary

```
# 딕셔너리 자료형
ex_dictionary = {1 : 1, 2 : "이", 3 : [3]}
```

요소 수정	요소 형태	특징
0	Key : Value(anything)	중괄호 index7t 없음

- 자바와 C++의 Map형태와 유사하다.
- Dictionary자료형은 다양한 함수를 지엄하기 때문에 알아두면 유용하다.
 key의 리스트를 만들어주는 keys()
 value의 리스트를 만들어주는 values()
 해당 Dictionary의 특정 key를 탐색하는 in
 - key값은 문자나 숫자. 튜플같은 변하지 않는 값만 들어감.



Set

```
# 집합 자료형
ex_set = set([1, "이", [3]])
```

요소 수정	요소 형태	특징
0	anything but set	set()을 통해 생성 중복 비허용 순서가 없음

- Set은 어떤 경우에 쓴담?

자료의 중복을 제거할 때(많이 쓰지는 않음)

자료의 집합 연산을 할 때(합집합, 차집합, 교집합) ->> 집합연산 필요 시 가장 유용하게 쓸 수 있는 자료구조



NEXT STAGE

