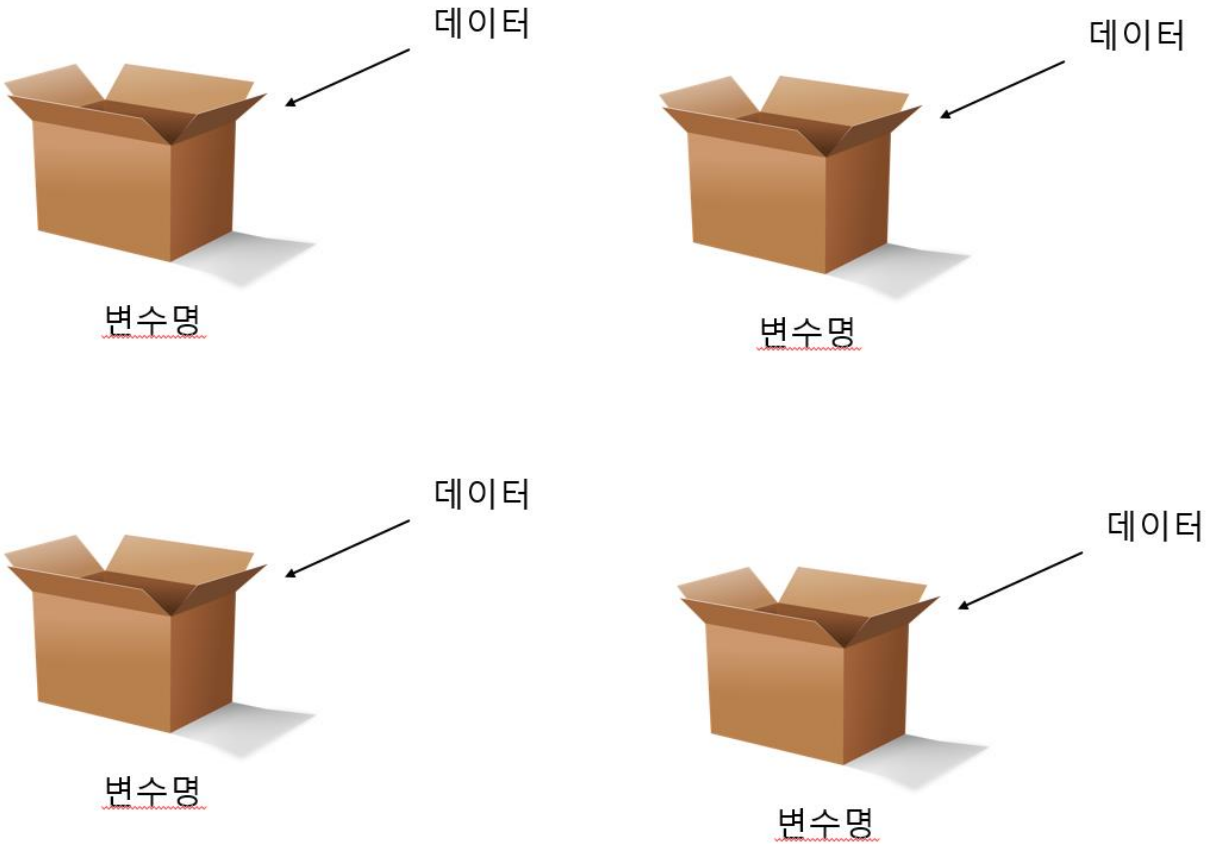
03-1 자료형(리스트)



03-1 자료형(리스트)

▪ 리스트(LIST)란?

- ✓ 자료구조 형태중 하나
- ✓ 대괄호[ ]로 묶어 값을 입력하며, 내부 값은 콤마(,) 로 구분
- ✓ 순서가 있는 수정 가능한 객체들의 집합
- ✓ 추가, 수정, 삭제 가능

03-1 자료형(리스트)

- 리스트명 = [요소1, 요소 2, 요소3, ...]
  - ✓ a = [ ] >> 빈값
  - ✓ a = [1, 2, 3] >> 숫자
  - ✓ a = ['python', 'is', 'fun'] >> 문자
  - ✓ a = [1, 2, 'python'] >> 숫자 + 문자
  - ✓ a = [1, 2, ['python', 'is', 'fun']] >> 숫자 + 리스트

03-1 자료형(리스트)

- 리스트의 인덱싱 : '가르킨다'라는 의미

  ✓ 리스트[인덱스 번호] : 인덱스 번호에 위치한 값을 반환

  ✓ num = [1, 2, 3]

  >> num[0] =?

  >> num[-1] = ?

  >> num[0] + a[2] = ?

03-1 자료형(리스트)

▪ 연습문제(인덱싱)

- ✓ list_ex1 = ['a', 'b', 'c', [1, 2, 3]] 이라는 리스트가 있다.
- ✓ 1) 변수 number에 [1, 2, 3]을 담아 출력하라
- ✓ 2) number에 담은 값 중 1과 3을 더해 4를 출력하라

03-1 자료형(리스트)

- 리스트의 슬라이싱 : '잘라낸다'라는 의미

  ✓ 리스트[시작:끝] : 시작 <= 원하는 값 < 끝

  ✓ num = [1, 2, 3, 4, 5]

  >> num[0:2]

  >> num[:2]

  >> num[3:]

03-1 자료형(리스트)

▪ 리스트 더하기

- ✓ list_poke_01 = ["피카츄", "라이츄"]
- ✓ list_poke_02 = ["파이리", "꼬부기"]
- ✓ list_poke_all = list_poke_01 + list_poke_02
- ✓ print(list_poke_all) –> ?

03-1 자료형(리스트)

▪ 리스트 반복하기(곱하기)

- ✓ list_num = [1, 2, 3, 4]
- ✓ list_result = list_num * 3
- ✓ print(list_result) -> ?

▪ 리스트 길이 구하기(내장함수)

- ✓ len(리스트명)

03-1 자료형(리스트)

▪ 리스트 값 수정하기

- ✓ list_num = [1, 3, 5, 7, 9, 10]
- ✓ 위 리스트에서 3번째 값을 정수 4로 수정하라.
- ✓ 위 리스트에서 2~5번째 값을 10, 20, 30, 40으로 수정하라.

03-1 자료형(리스트)

▪ 리스트 요소 개수 세기(count)

- ✓ 리스트.count(값)
- ✓ list_num = [1, 2, 3, 4, 5, 1, 2]
- ✓ 리스트에 포함된 값들 중 1이 몇 개 있는지 세어보시오.

03-1 자료형(리스트)

▪ 리스트 값 삭제하기(del)

- ✓ del 리스트명[인덱스] OR del 리스트명[시작:끝]
- ✓ list_num = [1, 3, 5, 7, 9, 10]
- ✓ 위 리스트에서 3번째 값을 삭제하라.
- ✓ 위 리스트에서 2~5번째 값을 삭제하라

03-1 자료형(리스트)

- 리스트 값 삭제하기(remove)
  - ✓ 리스트.remove(값)
  - ✓ list_num = [1, 3, 5, 7, 9, 10]
  - ✓ 위 리스트에서 원하는 값을 삭제하라.

03-1 자료형(리스트)

- 리스트 값 추가하기(append, insert, extend)
  - ✓ 리스트명.append(값)
  - ✓ list_number = ["one", "two", "three", "four", "five"]
  - ✓ "six"을 리스트에 추가
  - ✓ ['one', 'two', 'three']을 리스트에 추가

03-1 자료형(리스트)

- 리스트 값 추가하기(append, insert, extend)
  - ✓ 리스트명.insert(인덱스번호, 값)
  - ✓ list_number = ["one", "two", "three", "four", "five"]
  - ✓ 리스트 [2]번째에 "2.5"을 추가
  - ✓ 리스트 [3]번째에 ["3", "4"]을 추가

03-1 자료형(리스트)

- 리스트 값 추가하기(append, insert, extend)
  - ✓ 리스트명.extend([요소1, 요소2, ...])
  - ✓ list_number = ["one", "two", "three", "four", "five"]
  - ✓ 리스트 마지막에 "six", 7, "eight"을 추가
  - ✓ 리스트 [1]번째에 2, 3, 4, 5를 추가

03-1 자료형(리스트)

▪ 리스트 값 정렬하기(sort)

- ✓ 리스트.sort() -> 오름차순 정렬
- ✓ list_name = ["박효신", "이수", "김범수", "나얼"]
- ✓ ㄱㄴㄷㄹ 이름순으로 정렬하라

03-1 자료형(리스트)

- 리스트 뒤집기(reverse)
  - ✓ 리스트.reverse()
  - ✓ list_name = ["박효신", "이수", "김범수", "나얼"]
  - ✓ 해당 리스트를 역순으로 뒤집어라.

03-1 자료형(리스트)

▪ 연습문제(sort_정렬)

- ✓ list_lang = ['banana', 'cat', 'egg', 'cat', 'apple', 'door']
- ✓ 위와 같은 리스트가 있다.
- ✓ 이를 내림차순 정렬하라.

03-1 자료형(리스트)

▪ 리스트 위치반환(index)

- ✓ 리스트.index(값)
- ✓ list_poke = ["피카츄", "라이츄", "파이리", "꼬부기"]
- ✓ 리스트 중 "꼬부기"는 몇번째 문자인가?
- ✓ stra = “Courage is very important. Like a muscle, it is strengthened by use
- ✓ 위 문구중 b는 몇번째에 위치해 있는가?

03-1 자료형(리스트)

▪ 리스트 요소 끄집어내기(pop)

- ✓ 리스트.pop()
- ✓ 리턴값이 리스트가 아닌 값이다.
- ✓ list_name = [“김돌쇠”, “김갑돌”, “김갑순”, “김철수”]
- ✓ 리스트 중 마지막 요소만을 꺼내고 남겨라

03-2 자료형(튜플)

- 튜플(tuple)이란?
  - ✓ 자료구조 형태중 하나
  - ✓ 소괄호() 로 묶어 값을 입력하며, 내부 값은 콤마(,) 로 구분
  - ✓ 순서가 있는 수정 불가능한 객체들의 집합
  - ✓ 추가, 수정, 삭제 불가능

03-2 자료형(튜플)

- 튜플명 = (요소1, 요소 2, 요소3, ...)
  - ✓ a = ( ) >> 빈값
  - ✓ a = (1,) >> 숫자
  - ✓ a = ('python', 'is', 'fun') >> 문자
  - ✓ a = 1, 2, 'python' >> 숫자 + 문자
  - ✓ a = (1, 2, ('python', 'is', 'fun')) >> 숫자 + 리스트

03-2 자료형(튜플)

튜플 (특징 : 불변)

- ✓ tuple_a = (1,2,3)
- ✓ 길이, 값 고정

리스트 (특징 : 가변)

- ✓ list_a = [1,2,3]
- ✓ 길이, 값 수정 가능

03-2 자료형(튜플)_튜플&리스트의 공통점과 차이점

▪ 공통점

- ✓ 타입과 상관없이 다양한 자료형(요소)를 갖을 수 있다
- ✓ 인덱스 번호를 통해 요소들의 순서를 관리한다

▪ 차이점

- ✓ 리스트는 가변적(mutable)이며, 튜플은 불변적(immutable)이다.
- ✓ 리스트는 요소가 몇 개 들어갈지 명확치 않을 때, 튜플은 알고 있을 경우 사용
- ✓ 바뀌면 안되는 보호되어야 하는 정보는 튜플을 사용하여 명시

03-2 자료형(튜플)

- 튜플의 값 변경(불가능)
  - ✓ t1 = (1, 2, 'a', 'b')
  - ✓ t1[2] = 3

03-2 자료형(튜플)

- 튜플의 값 삭제(불가능)
  - ✓ t1 = (1, 2, 'a', 'b')
  - ✓ del t1[0]

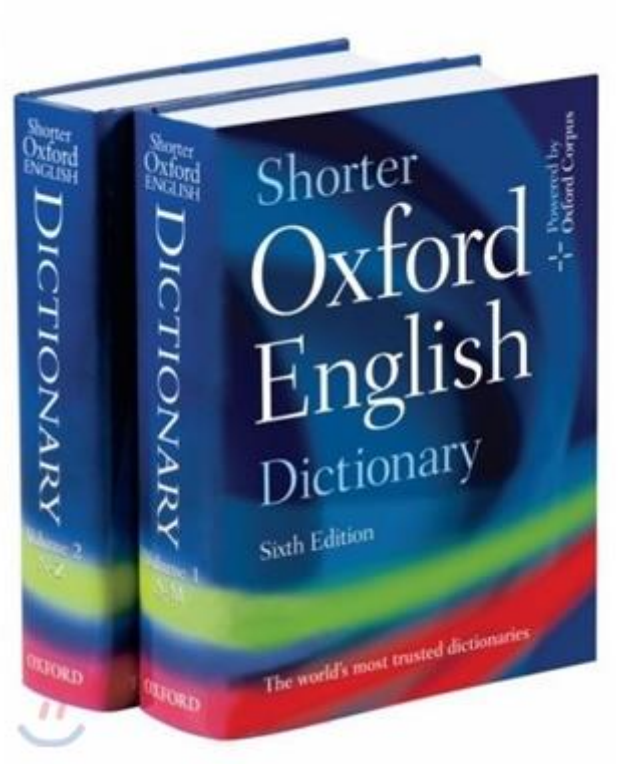
03-2 자료형(튜플)

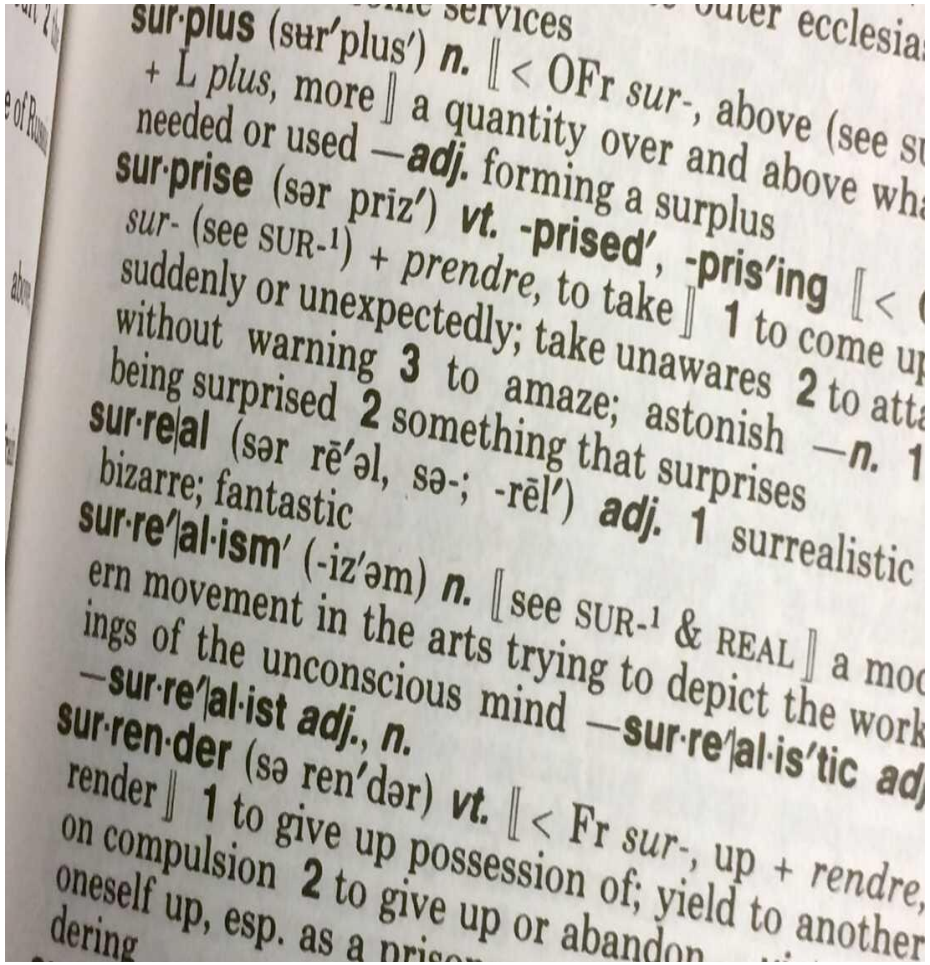
- 튜플의 인덱싱, 슬라이싱(가능)
  - ✓ t1 = (1, 2, 'a', 'b')
  - ✓ print(t1[3])
  - ✓ print(t1[0:3])
  - ✓ print(t1)

03-2 자료형(튜플)

- 튜플의 더하기, 곱하기
  - ✓ t1 = (1, 2, 'a', 'b')
  - ✓ t2 = (3, 4, 'c', 'd')
  - ✓ print(t1 + t2)
  - ✓ print(t1 * 2)

03-3 자료형(딕셔너리)


Shorter
Oxford
English
Dictionary
Sixth Edition
The world's most trusted dictionaries
Powered by Oxford Corpus
OXFORD


sur·plus (sʉr′plus′) n. ‖ < OFr sur-, above (see su
+ L plus, more ‖ a quantity over and above wha
needed or used —adj. forming a surplus
sur·prise (sər prīz′) vt. -prised′, -pris′ing ‖ <
sur- (see SUR-1) + prendre, to take ‖ 1 to come up
suddenly or unexpectedly; take unawares 2 to atta
without warning 3 to amaze; astonish —n. 1
being surprised 2 something that surprises
sur·re|al (sər rē′əl, sə-; -rēl′) adj. 1 surrealistic
bizarre; fantastic
sur·re′|al·ism′ (-iz′əm) n. ‖ see SUR-1 & REAL ‖ a mod
ern movement in the arts trying to depict the work
ings of the unconscious mind —sur·re|al·is′tic adj
—sur·re′|al·ist adj., n.
sur·ren·der (sə ren′dər) vt. ‖ < Fr sur-, up + rendre,
render ‖ 1 to give up possession of; yield to another
on compulsion 2 to give up or abandon
oneself up, esp. as a
dering

03-3 자료형(딕셔너리)

- Key + value 로 이루어진 자료구조
  - ✓ Hash (ruby)
  - ✓ Map (java)
  - ✓ Object (javascript)
  - ✓ JSON형태

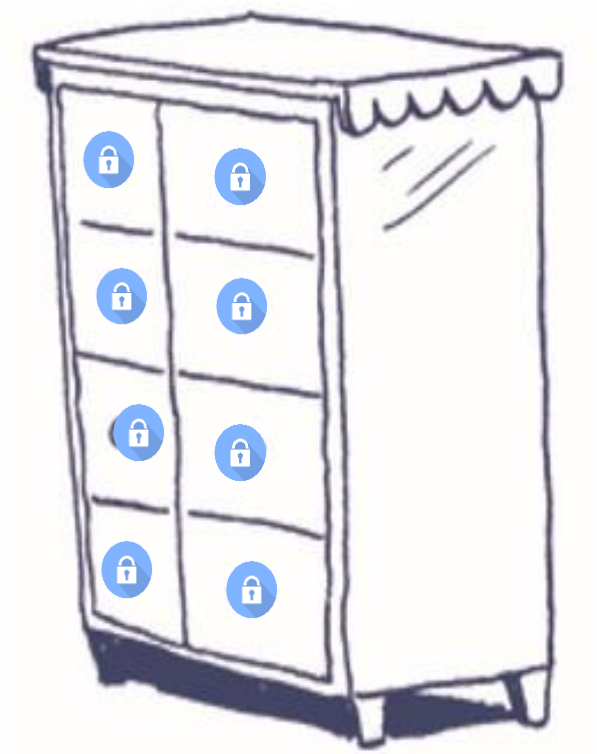
03-3 자료형(딕셔너리)

▪ 딕셔너리란?

- ✓ 사전에 단어:뜻으로 매칭되듯 {KEY:VALUE}를 한쌍으로 갖는 자료형
- ✓ 불변적(immutable)인 KEY와 가변적(mutable)인 VALUE로 매핑되어 있는

  순서가 없는 자료구조
- ✓ 키는 중복이 허용되지 않고, 값은 중복이 허용된다.
- ✓ 리스트/튜플처럼 인덱스(순서)를 통해 값을 가져 오지 않고 Key를 통해 값을 가져온다.
- ✓ Key를 이용하여 값의 수정, 삭제, 추가 가능하다.

03-3 자료형(딕셔너리) : {key:value, ...}



key	value
Name	Python
Age	30
Address	Netherlands
Phone	8282
school	no

▪ 선언

✓ 변수명 = {'key1' : 'value1', 'key2' : 'value2', 'key3' : 'value3'}

03-3 자료형(딕셔너리)

▪ 딕셔너리 생성 및 추가

- ✓ a = {'이름' : '아무개'} # [선언]
- ✓ 변수명[key] = value # [추가이자 수정]
- ✓ '신분':'학생' 의 쌍을 추가하라.

▪ 딕셔너리 삭제

- ✓ del 변수명[key] // 변수명.pop('key')
- ✓ Key가 이름인 한쌍을 지워라.

03-3 자료형(딕셔너리)

▪ 딕셔너리 확장

- ✓ dict_01 = {'a':1, 'b':2, 'c':3}
- ✓ dict_02 = {'a':1, 'd':4, 'e':5}
- ✓ 변수명.update('합치고자 하는 딕셔너리 변수명')
- ✓ 키값이 동일한 딕셔너리끼리 합친다면?

03-3 자료형(딕셔너리)

▪ 생성시 주의사항

- ✓ key가 중복된다면 어떻게 될까?
- ✓ value가 중복된다면 어떻게 될까?

03-3 자료형(딕셔너리)

- Key를 통해 value얻기(인덱싱과 같은...)
  - ✓ 딕셔너리 변수명[key]
  - ✓ dict_profile = {'이름':'홍길동', '나이':25, '성별':'여'}
  - ✓ print(dict_profile['이름'])
  - ✓ 또 다른 호출 방법 : 변수명.get(key)

03-3 자료형(딕셔너리)_리스트&튜플과의 차이

- 딕셔너리는 a={key:value}, 호출시 a[key]
- 리스트는 a=[value], 호출시 a[인덱스 번호(순서)]
- 튜플은 a=(value), 호출시 a[인덱스 번호(순서)]

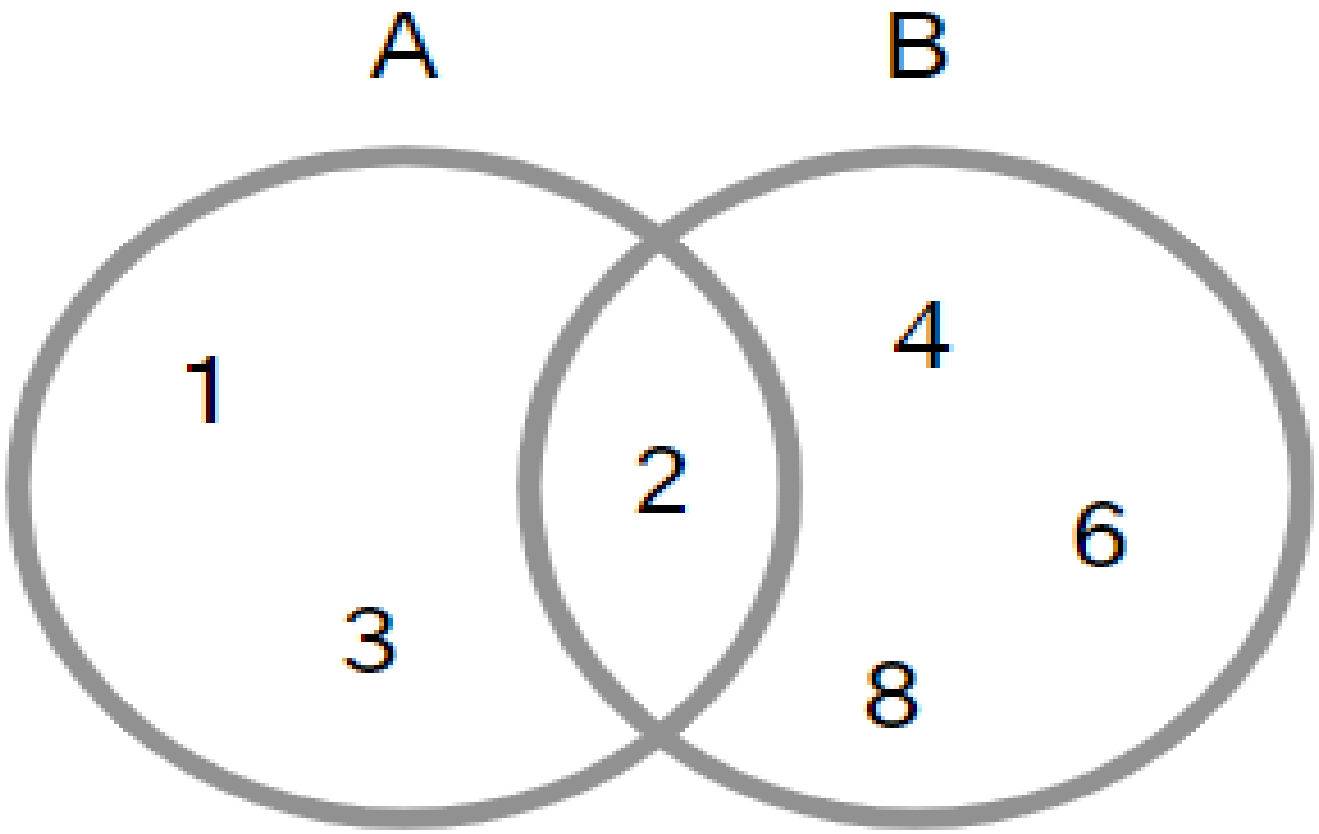
03-3 자료형(딕셔너리)

- Key 리스트 뽑아내기
  - ✓ 딕셔너리 변수명.keys()
- Value 리스트 뽑아내기
  - ✓ 딕셔너리 변수명.values()

03-3 자료형(딕셔너리)

- Key, Value 쌍 얻기
  - ✓ 변수명.items()
  - ✓ Key,value 쌍을 튜플로 묶어 dict객체로 반환
- Key, Value 쌍 모두 지우기(clear)
  - ✓ 변수명.clear()

03-4 자료형(집합)



03-4 자료형(집합)

- 셋(set)이란?
  - ✓ 자료구조 형태중 하나
  - ✓ 중괄호 {}로 묶어 값을 입력하며, 내부 값은 콤마(,) 로 구분
  - ✓ 순서가 없는 수정 가능한 객체들의 집합
  - ✓ 추가, 수정, 삭제 가능
  - ✓ 중복을 허용하지 않음

03-4 자료형(집합)

- ✓ 변수 = set([리스트형] OR “문자열”)
- ✓ 중복되지 않는 자료의 모임
- ✓ set([1, 1, 3, 2]) -> {1, 3, 2}
- ✓ 변수명[0] 과 같이 인덱싱 접근 불가.
- ✓ 인덱싱으로 접근하기 위해서는 리스트형으로 형변환 후 접근

03-4 자료형(집합)

- set명 = {요소1, 요소 2, 요소3, ...}
  - ✓ 변수명 = {1, 2, 3} >> 숫자
  - ✓ 변수명 = {'1', '2', '3'} >> 문자
  - ✓ 변수명 = set([리스트] OR "문자열")
  - ✓ set([1, 1, 3, 2]) -> {1, 3, 2}

03-4 자료형(집합)

- 집합 자료형
  - ✓ s = set([1,2,3])
  - ✓ s = {1,2,3}
- 순서가 없다.
  - ✓ s[0] 불가

03-4 자료형(집합)

■ 합집합

- ✓ s1 = set([1,2,3,4,5,6])
- ✓ s2 = set([4,5,6,7,8,9])
- ✓ s1 | s2
- ✓ s1.union(s2)

03-4 자료형(집합)

■ 교집합

- ✓ s1 = set([1,2,3,4,5,6])
- ✓ s2 = set([4,5,6,7,8,9])
- ✓ s1 & s2
- ✓ s1.intersection(s2)

03-4 자료형(집합)

- 차집합
  - ✓ s1 = set([1,2,3,4,5,6])
  - ✓ s2 = set([4,5,6,7,8,9])
  - ✓ s1 – s2
  - ✓ s1.difference(s2)

03-4 자료형(집합)

- 값 1개 추가(add)
  - ✓ set변수명.add(값)
- 값 여러 개 추가(update)
  - ✓ set변수명.update({값1, 값2})

03-4 자료형(집합)

- 값 삭제(remove, discard)

  ✓ set변수명.remove(값)

  없는 값 제거시 에러 발생

  ✓ set변수명.discard(값)

  none 리턴

03-5 자료형(불)

- 참(True)
  - ✓ a = True
  - ✓ b = true?
- 거짓(False)
  - ✓ a = False
  - ✓ b = false

목적	연산자	의미
비교	<	좌항이 우항보다 작다면 참
비교	>	좌항이 우항보다 크다면 참
비교	<=	우항이 좌항보다 크거나 같다면 참
비교	>=	좌항이 우항보다 크거나 같다면 참
비교	==	좌항과 우항이 같다면 참
비교	!=	좌항과 우항이 다르다면 참
논리	and	논리 and 논리(두논리 다 참일때만 참)
논리	or	논리 or 논리(두 논리중 하나만 참이어도 참)
논리	not	not 논리 : True -> False, False -> True

03-5 자료형(불)

값	참 or 거짓
“python”	참
“”	거짓
[1,2,3]	참
[]	거짓
()	거짓
{}	거짓
1	참
0	거짓
None	거짓

03-5 자료형(불)

- if문과 in을 이용한 조건문
  - ✓ a = [1,2,3,4]
  - ✓ if 1 in a :
  - ✓ print(a.pop())
  - ✓ print(a)
- While문과 pop함수 특성을 이용한 반복문
  - ✓ while a :
  - ✓ print(a.pop())