

9주차

프로그래밍과 문제해결

▶ 리스트

프런티어창의대학
이경자교수



▶ 리스트

1교시 리스트 생성과 자료추가

2교시 리스트 조작함수

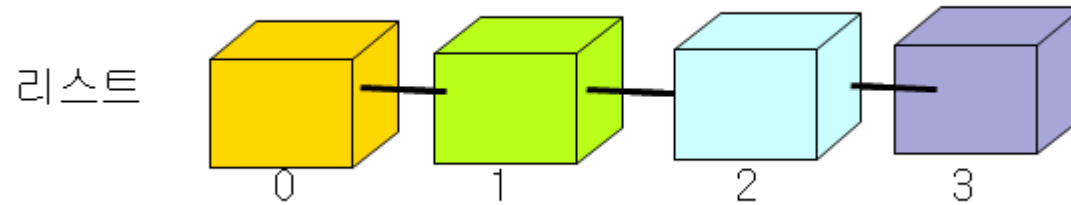
3교시 리스트 활용

리스트(List)










▶ 리스트

- 리스트는 항목(item)들을 저장하는 컨테이너로서 그 안에 항목들이 **순서를 가지고** 저장.
- 리스트는 어떤 타입의 항목도 저장가능. 유용하고 많이 사용.



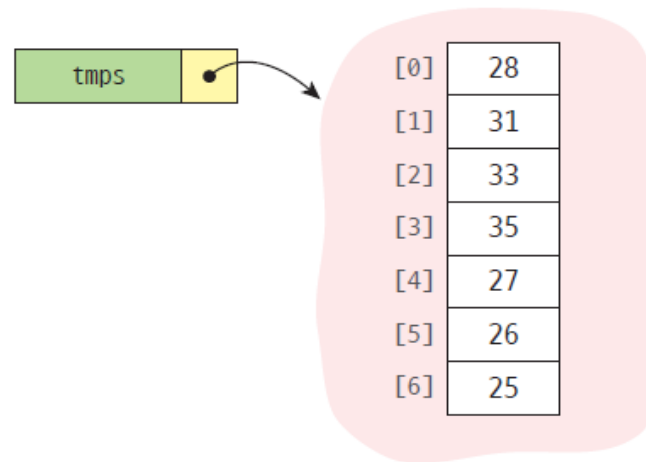
▶ 리스트를 사용하는 예

- 지난 1주일 중에서 가장 더운 날을 찾으려고 한다고 하자.

MON	TUE	WED	THU	FRI	SAT	SUN
						
28	31	33	35	27	26	25



```
temps = [28, 31, 33, 35, 27, 26, 25]
```



▶ 리스트 만들기

Syntax: 리스트

형식 리스트_이름 = [요소1, 요소2, ...]

예 temps = [28, 31, 33, 35, 27, 26, 25]
e = temps[3]

초기값을 가진 리스트를 생성한다.

리스트의 이름


대괄호를 사용하여 요소에 접근한다.

- 여러 개의 데이터를 하나로 묶어서 저장.
- 항목(Item)들을 **순차적**으로 모아놓음.
- [] 사이에 ,로 구분하여 항목 나열
- 서로 다른 data type 가능
 - 리스트도 리스트의 항목이 될 수 있음

▶ 리스트에 접근

```
>>> letters = ['A', 'B', 'C', 'D', 'E', 'F']
```

'A'	'B'	'C'	'D'	'E'	'F'
0	1	2	3	4	5



```
>>> print(letters[0])
```

A

```
>>> print(letters[1])
```

B

```
>>> print(letters[2])
```

C

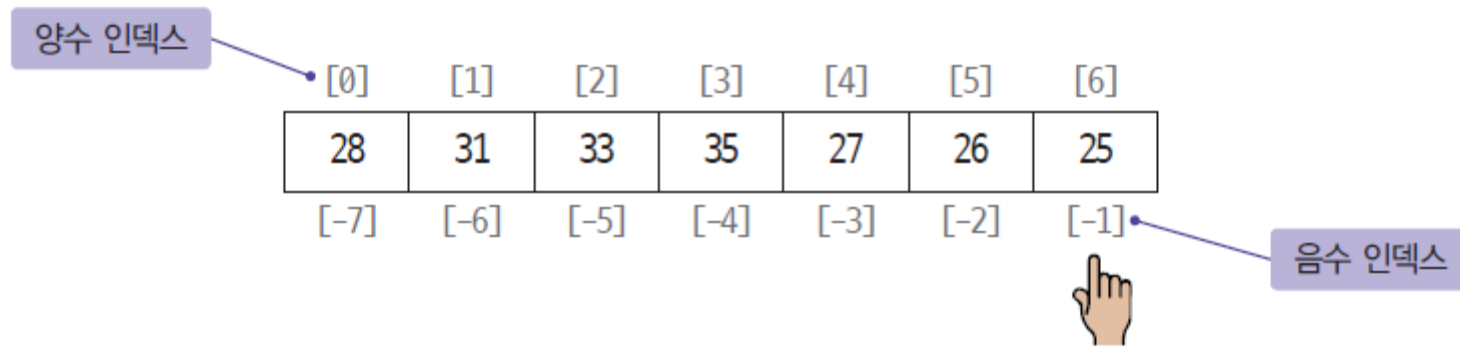
```
>>> print(letters[7])
```

Index error : list index out of range

▶ 음수인덱스

- 음수 인덱스는 리스트의 끝에서부터 매겨진다.

temps[5]==temps[-2]



▶ 리스트 조작함수

- 기본사용방법 : 리스트이름.함수명()

함수	설명	사용법
append()	리스트 제일 뒤에 항목을 추가한다.	리스트이름.append(값)
pop()	리스트 제일 뒤의 항목을 빼내고, 빼낸 항목은 삭제한다.	리스트이름.pop()
sort()	리스트의 항목을 정렬한다.	리스트이름.sort()
reverse()	리스트 항목의 순서를 역순으로 만든다.	리스트이름.reverse()
index()	지정한 값을 찾아서 그 위치를 반환한다.	리스트이름.index(찾을 값)
insert()	지정된 위치에 값을 삽입한다.	리스트이름.insert(위치, 값)
remove()	리스트에서 지정한 값을 제거한다. 단 지정한 값이 여러 개일 경우 첫 번째 값만 지운다.	리스트이름.remove(지울 값)
extend()	리스트 뒤에 리스트를 추가한다. 리스트의 더하기(+) 연산과 동일한 기능을 한다.	리스트이름.extend(리스트)
count()	리스트에서 찾을 값의 개수를 센다.	리스트이름.count(찾을 값)
del()	리스트에서 해당 위치의 항목을 삭제한다.	del(리스트이름[위치])
len()	리스트에 포함된 전체 항목의 개수를 센다.	len(리스트이름)

▶ 자료추가(1) : append()

- append(item) : 항목을 리스트의 **끝에** 추가

```
>>> heroes = [ ]  
>>> heroes.append("아이언맨")  
['아이언맨']
```

```
>>> heroes.append("닥터 스트레인지")  
>>> print(heroes)  
['아이언맨', '닥터 스트레인지']
```

추가해보기 → heroes=['아이언맨', '닥터 스트레인지', '헐크', '스칼렛 위치']

▶ 자료추가(1) : append()

```
aa=[ ]  
aa.append(0)  
aa.append(0)  
aa.append(0)  
aa.append(0)  
print(aa)
```

[0,0,0,0]

- 항목이 100개인 리스트를 만들어 모두 1로 채우기

```
aa=[ ]  
for i in range(100):  
    aa.append(1)  
  
print(aa)  
print(len(aa))
```

▶ 자료추가(2) : extend()

```
>>> nums = [1,2,3]
>>> nums.extend(['a', 'b', 'c'])
>>> print(nums)
[1, 2, 3, 'a', 'b', 'c']
```

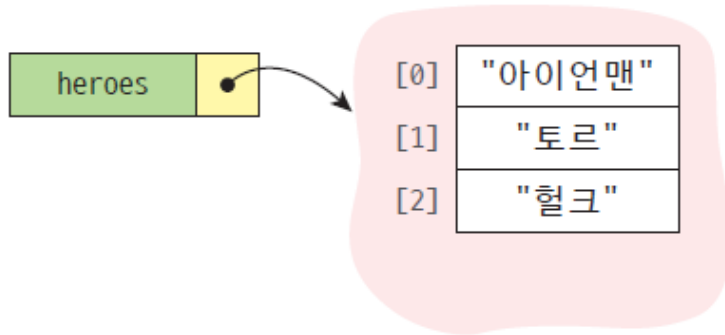
```
>>> nums = [1,2,3]
>>> letters = ['a', 'b', 'c']
>>> nums.extend(letters)
>>> print(nums)
[1, 2, 3, 'a', 'b', 'c']
```

```
>>> nums = [1,2,3]
>>> letters = ['a', 'b', 'c']
>>> new=nums+letters
>>> print(new)
[1, 2, 3, 'a', 'b', 'c']
```

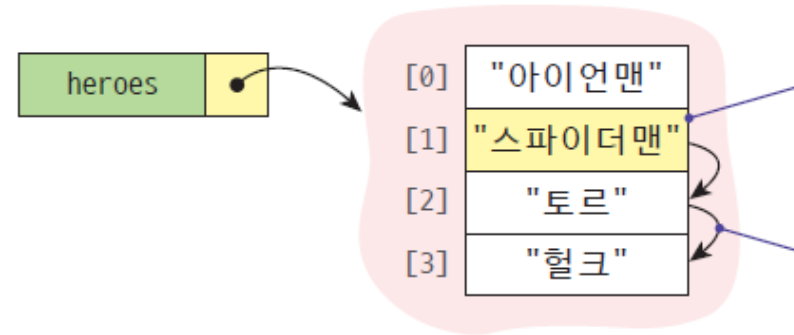
▶ 자료추가(3) : insert()

- **insert(index, item)** : 항목을 인덱스 위치에 추가

① `heroes=("아이언맨", "토르", "헐크")`



② `heroes.insert(1, "스파이더맨")`



▶ 자료추가(3) : insert()

- `insert(index, item)` : 항목을 인덱스 위치에 추가

```
>>> heroes.append("스파이더맨")
```

```
>>> print(heroes)
```

```
['아이언맨', '닥터 스트레인지', '헐크', '스칼렛 위치', '스파이더맨']
```

```
>>> heroes.insert(1, "배트맨")
```

```
>>> print(heroes)
```

```
['아이언맨', '배트맨', '닥터 스트레인지', '헐크', '스칼렛 위치', '스파이더맨']
```

▶ 반복문을 이용한 리스트 조회

- List를 sequence로 한 for loop

```
>>> letters = ['a', 'b', 'c', 'd']  
>>> for i in letters:  
    print(i)
```

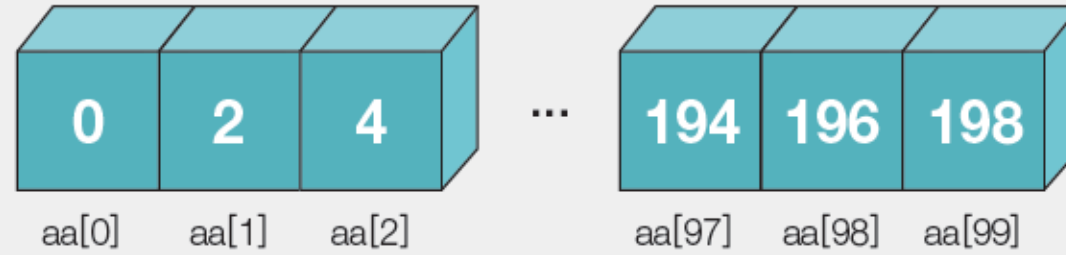
- List index를 이용한 for loop

```
>>> letters = ['a', 'b', 'c', 'd']  
>>> for i in range(len(letters)):  
    print(letters[i])
```

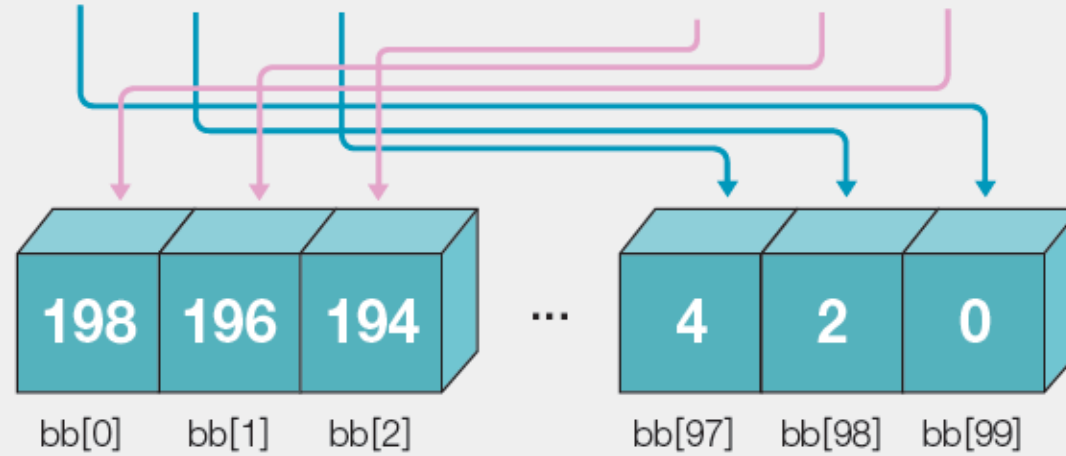
- Nested list의 for loop

```
>>> letters = [[1, 2], [3, 4], [5, 6]]  
>>> for i, j in letters:  
    print(i, j)
```

① 리스트 aa를
짝수로 초기화



② 리스트 bb에
역순으로 대입




```
aa=[]  
  
bb=[]  
  
value=0  
  
for i in range(0,100):  
    aa.append(value)  
    value=value+2  
  
for i in range(0,100):  
    bb.append(aa[99-i])
```

리스트 자료변경



▶ 리스트 자료변경

```
>>> heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
>>> heroes[1] = "닥터 스트레인지"  
>>> print(heroes)  
['아이언맨', '닥터 스트레인지', '헐크', '스칼렛 위치']  
>>> heroes[5] = "슈퍼맨"
```

생성되지
않은
인덱스는
변경불가능

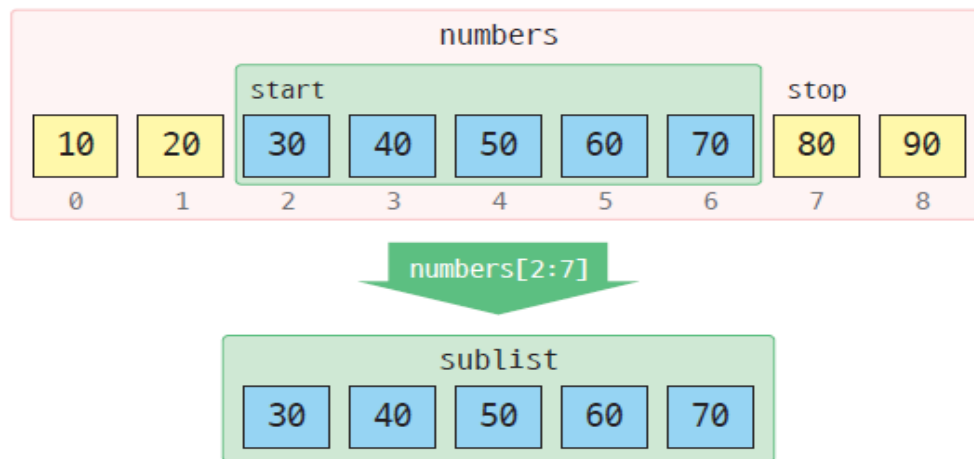
▶ 슬라이싱(slicing)

- 슬라이싱(slicing)은 리스트에서 한 번에 여러 개의 항목을 추출하는 기법

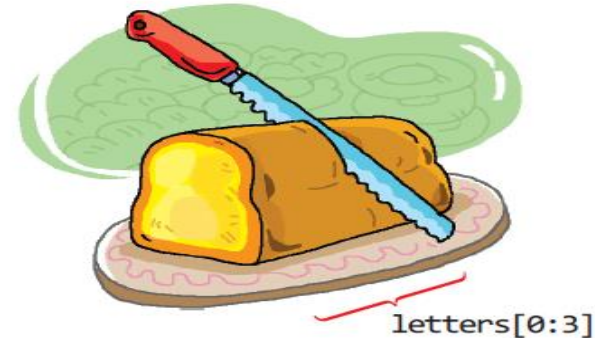
Syntax: 슬라이싱 #1

형식 리스트[start : stop]

예 numbers = [10, 20, 30, 40, 50, 60, 70, 80, 90]
sublist = numbers[2:7]



음수는 맨 뒷자리부터



▶ 슬라이싱(slicing)

```
>>> letters = ['A', 'B', 'C', 'D', 'E', 'F']
```

```
>>> print(letters[:3])  
['A', 'B', 'C']
```

```
>>> print(letters[3:])  
['D', 'E', 'F']
```

```
>>> print(letters[:])  
['A', 'B', 'C', 'D', 'E', 'F']
```

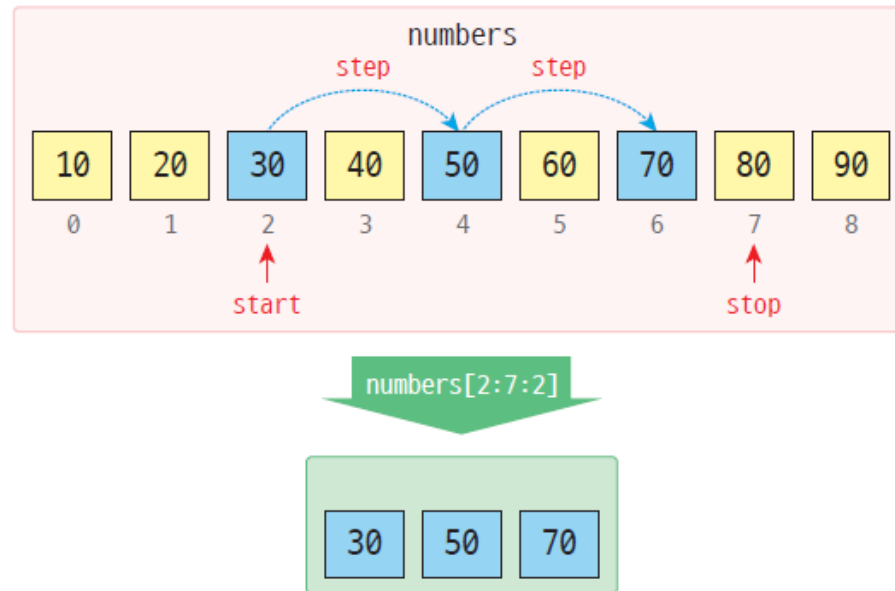
```
>>> print(letters[-3:-1])  
['D', 'E']
```

▶ 고급 슬라이싱

Syntax: 슬라이싱 #2

형식 리스트[start : stop : step]

예 numbers = [10, 20, 30, 40, 50, 60, 70, 80, 90]
sublist = numbers[2:7:2]

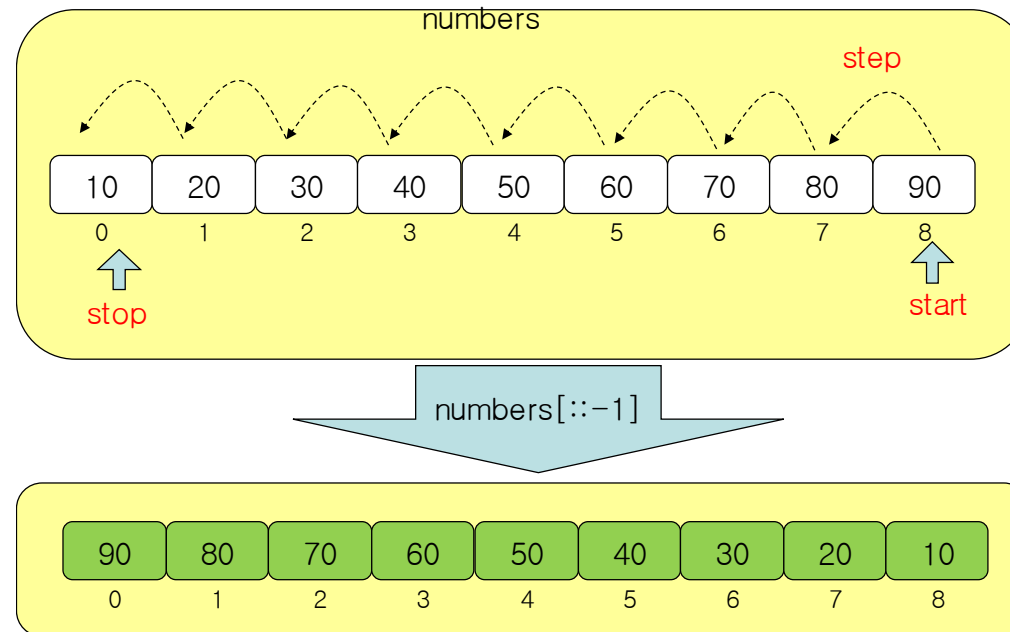


▶ 리스트를 역순으로 만드는 방법

```
>>> numbers = [ 10, 20, 30, 40, 50, 60, 70, 80, 90 ]
```

```
>>> numbers[::-1]
```

```
[90, 80, 70, 60, 50, 40, 30, 20, 10]
```



▶ 슬라이싱을 이용한 리스트 자료변경

```
>>> print(heroes)
['아이언맨', '닥터 스트레인지', '헐크', '스칼렛 위치']

>>> heroes[2:4] = ["슈퍼맨 ", " 원더우먼 "]

>>> print(heroes)
['아이언맨', '닥터 스트레인지', ' 슈퍼맨 ', ' 원더우먼 ']
```


▶ 슬라이싱을 이용한 리스트 자료변경

```
>>> lst = [1, 2, 3, 4, 5, 6, 7, 8]
>>> lst[0:3] = ['white', 'blue', 'red']
>>> lst
['white', 'blue', 'red', 4, 5, 6, 7, 8]
```

리스트 일부 변경

```
>>> lst = [1, 2, 3, 4, 5, 6, 7, 8]
>>> lst[::2] = [99, 99, 99, 99]
>>> lst
[99, 2, 99, 4, 99, 6, 99, 8]
```

99를 중간에 추가한다.

```
>>> lst = [1, 2, 3, 4, 5, 6, 7, 8]
>>> lst[:] = []
>>> lst
[]
```

리스트의 모든 요소를 삭제한다.

리스트 자료삭제



▶ 리스트 자료삭제

- 기본사용방법 : 리스트이름.함수명()

함수	설명	사용법
append()	리스트 제일 뒤에 항목을 추가한다.	리스트이름.append(값)
→ pop()	리스트 제일 뒤의 항목을 빼내고, 빼낸 항목은 삭제한다.	리스트이름.pop()
sort()	리스트의 항목을 정렬한다.	리스트이름.sort()
reverse()	리스트 항목의 순서를 역순으로 만든다.	리스트이름.reverse()
index()	지정한 값을 찾아서 그 위치를 반환한다.	리스트이름.index(찾을 값)
insert()	지정된 위치에 값을 삽입한다.	리스트이름.insert(위치, 값)
→ remove()	리스트에서 지정한 값을 제거한다. 단 지정한 값이 여러 개일 경우 첫 번째 값만 지운다.	리스트이름.remove(지울 값)
extend()	리스트 뒤에 리스트를 추가한다. 리스트의 더하기(+) 연산과 동일한 기능을 한다.	리스트이름.extend(리스트)
count()	리스트에서 찾을 값의 개수를 센다.	리스트이름.count(찾을 값)
→ del()	리스트에서 해당 위치의 항목을 삭제한다.	del(리스트이름[위치])
len()	리스트에 포함된 전체 항목의 개수를 센다.	len(리스트이름)



▶ 자료삭제(1) : remove()

- 리스트에서 지정한 값을 삭제
- 지정한 값이 여러 개일 경우 첫번째 값만 지운다

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
heroes.remove("스칼렛 위치")  
print(heroes)
```

```
['아이언맨', '토르', '헐크']
```

만약 삭제하고자 하는 항목이 없다면
오류(예외) 발생.



```
if "토르" in heroes:  
    heroes.remove("토르")
```

▶ 자료삭제(2) : pop()

- pop은 리스트에서 **마지막 항목을 꺼내고** 삭제
- 위치지정삭제 가능

```
>>> lst = [1, 2, 3, 4, 5]
>>> lst.pop()
5
>>> lst
[1, 2, 3, 4]
>>> lst.pop(2)
3
>>> lst
[1, 2, 4]
```

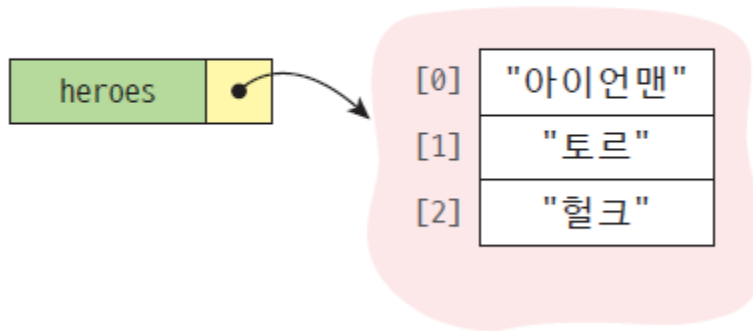
```
heroes = [ "아이언맨", "토르", "헐크"]
last_hero = heroes.pop()
print(last_hero)
```

헐크

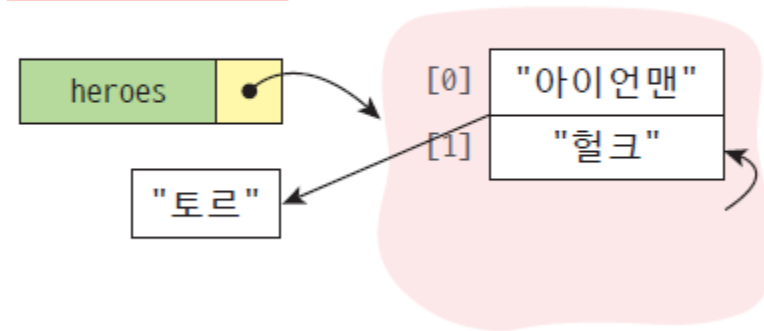
▶ pop() 과 remove()

- 항목이 저장된 위치를 알고 있다면 pop(i)을 사용한다.
- 항목의 값만 알고 있다면 remove(value)를 사용한다.

① heroes=("아이언맨", "토르", "헐크")



② `heroes.pop(1)`



▶ 자료삭제(3) : del()

- del은 인덱스를 사용하여 항목을 삭제한다

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
del heroes[0] 또는 heroes.pop(0)  
print(heroes)
```

```
['토르', '헐크', '스칼렛 위치']
```

▶ 자료삭제(4) : del

- del 리스트명 : **del heroes**
- del 리스트명[인덱스]

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
del heroes[0] 또는 del(heroes[0])  
print(heroes)
```

```
['토르', '헐크', '스칼렛 위치']
```


▶ 슬라이싱을 이용한 삭제

- del 리스트명[슬라이싱]
- 리스트명[시작:끝+1]=[]

```
aa = [10,20,30,40,50]  
aa[1:4]=[ ] #또는 del aa[1:4]  
print(aa)
```

```
[10, 50]
```

리스트 함수



▶ 리스트 자료정렬

```
>>> numbers = [9, 6, 7, 8, 3, 2, 5]  
>>> numbers.sort()  
>>> print(numbers)  
[2, 3, 5, 6, 7, 8, 9]
```

```
>>> numbers.reverse()  
>>> print(numbers)  
[5, 2, 3, 8, 7, 6, 9]
```

```
>>> numbers = [9, 6, 7, 8, 3, 2, 5]  
>>> numbers.sort(reverse=True)  
>>> print(numbers)  
[9, 8, 7, 6, 5, 3, 2]
```



▶ 리스트 자료정렬(파이썬 내장함수 sorted())

```
>>> lst = [3, 5, 1, 9, 6]
>>> print(sorted(lst))
[1, 3, 5, 6, 9]
>>> print(lst)    # 원본 list는 불변
[3, 5, 1, 9, 6]
>>> print(sorted(lst, reverse = True)) # 내림차순
[9, 6, 5, 3, 1]
>>> print(lst)
[3, 5, 1, 9, 6]    # 원본 list는 불변
```

Built-in Functions

A

abs()
aiter()
all()
any()
anext()
ascii()

B

bin()
bool()
breakpoint()
bytearray()
bytes()

C

callable()
chr()
classmethod()
compile()
complex()

D

delattr()
dict()
dir()
divmod()

E

enumerate()
eval()
exec()

F

filter()
float()
format()
frozenset()

G

getattr()
globals()

H

hasattr()
hash()
help()
hex()

I

id()
input()
int()
isinstance()
issubclass()
iter()

L

len()
list()
locals()

M

map()
max()
memoryview()
min()

N

next()

O

object()
oct()
open()
ord()

P

pow()
print()
property()

R

range()
repr()
reversed()
round()

S

set()
setattr()
slice()
sorted()
staticmethod()
str()
sum()
super()

T

tuple()
type()

V

vars()

Z

zip()

__import__()

▶ 리스트 연산자

- +연산자 : 리스트 결합

```
>>> a = [1, 2, 3, 4]
>>> b = [5, 6, 7]
>>> a + b
[1, 2, 3, 4, 5, 6, 7]
>>> c = a + b
>>> c
[1, 2, 3, 4, 5, 6, 7]
```

- *연산자 : 반복 (*n)

```
>>> a = [1, 2, 3, 4]
>>> b = a*3
>>> b
[1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4]
```

▶ 리스트 활용

- `x in S` : T or F
- `x not in S` : T or F

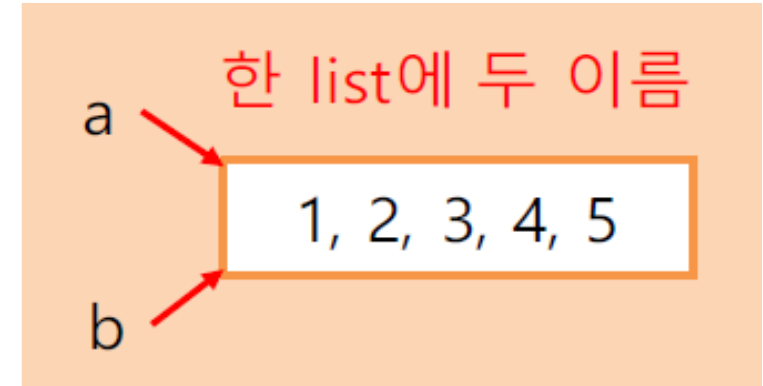
```
>>> if 4 in a:  
    a.remove(4)
```

```
>>> if "슈퍼맨" in heroes:  
    heroes.remove("슈퍼맨")
```

```
>>> if 4 not in a:  
    a.append(4)
```

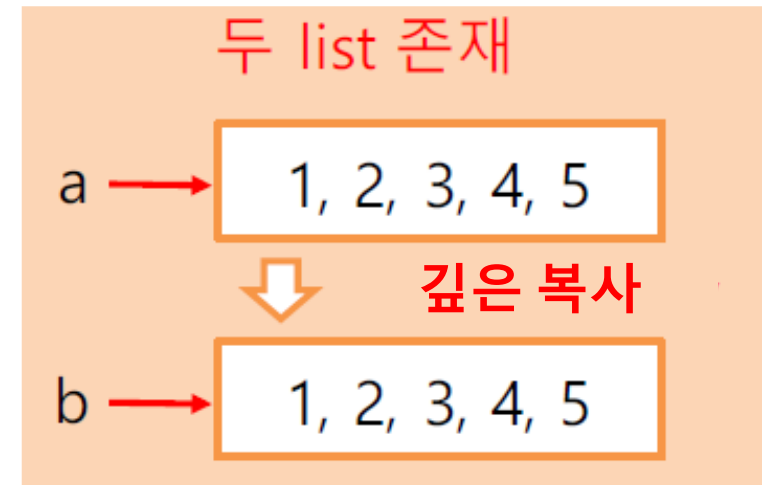
▶ 리스트 복사(copy)

```
>>> a = [1, 2, 3, 4, 5]
>>> b = a # 얇은 복사
>>> a.append(6)
>>> a
[1, 2, 3, 4, 5, 6]
>>> b
[1, 2, 3, 4, 5, 6]
```



- copy()

```
>>> a = [1, 2, 3, 4, 5]
>>> b = a.copy() # 깊은 복사
>>> a.append(6)
>>> a
[1, 2, 3, 4, 5, 6]
>>> b
[1, 2, 3, 4, 5]
```



▶ 리스트 함수

- len(list) : 리스트 항목의 개수
- min(list) : 리스트 항목 중 최소값 반환
- max(list) : 리스트 항목 중 최대값 반환
- count(값) : 해당값이 리스트에 포함된 개수
- index(값) : 해당값의 첫 인덱스 반환

```
>>> lst = [1, 2, 3, 2, 2, 4, 5, 6, 7]

>>> len(lst)
9
>>> min(lst)
1
>>> max(lst)
7
>>> lst.count(2)
3
>>> lst.count(7)
1
>>> lst.index(2)
1
>>> lst.index(7)
8
```


▶ 리스트 함수

- index(값, 시작위치) : 시작위치에서부터 찾기 시작

```
>>> lst = [1, 2, 3, 2, 2, 4, 5, 6, 7]
```

```
>>> lst.index(2,3)
```

```
4
```

- clear(리스트명) : 리스트의 모든 항목을 삭제
- sum(리스트명) : 리스트 요소들의 합

▶ 리스트 함수(3)

- 리스트에서 랜덤값 선택하기 : `random.choice(리스트명)`

```
import random  
numberList = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
print("랜덤하게 선택한 항목=", random.choice(numberList))
```

랜덤하게 선택한 항목= 6

```
import random  
movie_list = [ " 내부자들 ", " 공공의 적 ", " 서울의 봄 ", " 베테랑", "범죄도시"]  
item = random.choice(movie_list)  
print ("랜덤하게 선택한 항목=", item)
```

랜덤하게 선택한 항목= 서울의 봄

▶ 리스트 함수(4)

- sum(), min(), max()

```
numbers =[10,20,30,40,50]
```

```
print("합=",sum(numbers))      # 항목의 합계를 계산한다.  
print("최대값=",max(numbers))  # 가장 큰 항목을 반환한다.  
print("최소값=",min(numbers))  # 가장 작은 항목을 반환한다
```

```
합= 150  
최대값= 50  
최소값= 10
```

리스트 활용

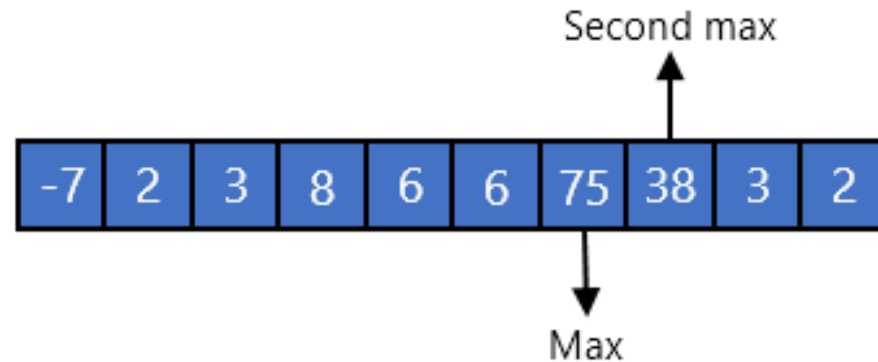


▶ 실습 1/3

- 정수들이 저장된 리스트에서 두 번째로 큰 수를 찾아보자. (sort, max 사용가능)

결과화면

두 번째로 큰 수 = 38



1

```
list1 = [-7, 3, 8, 6, 6, 75, 38, 3, 2]
```

```
# 리스트를 정렬한다.  
list1.sort()
```

```
# 뒤에서 두 번째 요소를 출력한다.  
print("두 번째로 큰 수=", list1[-2])
```

2

```
list1 = [-7, 3, 8, 6, 6, 75, 38, 3, 2]
```

```
list1.remove(max(list1))
```

```
print("두 번째로 큰 수=", max(list1))
```

▶ 실습2/3

- 심판들의 점수가 리스트에 저장되어 있다
- 최소값과 최대값을 리스트에서 제거하는 프로그램을 작성해보자.

결과화면

제거전 [10.0, 9.0, 8.3, 7.1, 3.0, 9.0]

제거후 [9.0, 8.3, 7.1, 9.0]



```
scores = [10.0, 9.0, 8.3, 7.1, 3.0, 9.0]
print("제거전", scores)
scores.remove(max(scores))
scores.remove(min(scores))
print("제거후", scores)
```


▶ 실습3/3

- 헌혈자에게 혈액형을 입력받아 리스트에 보관하고, 종류별로 개수를 보고하는 프로그램

결과화면

헌혈해주셔서 감사합니다.

혈액형을 선택하세요(A,B,AB,O) : O

A 형 개수 : 0

B 형 개수 : 0

AB 형 개수 : 0

O 형 개수 : 1

입력을 계속할까요?(yes/no) : yes

헌혈해주셔서 감사합니다.

혈액형을 선택하세요(A,B,AB,O) : AB

A 형 개수 : 0

B 형 개수 : 0

AB 형 개수 : 1

O 형 개수 : 1

입력을 계속할까요?(yes/no) : no

```
bloods=[]
while True:
    print('헌혈해주셔서 감사합니다.')
    type=input('혈액형을 선택하세요 (A,B,AB,O) :' )
    bloods.append(type)
    print("A형 :", bloods.count('A'))
    print("B형 :", bloods.count('B'))
    print("AB형 :", bloods.count('AB'))
    print("O형 :", bloods.count('O'))
    answer= input('입력을 계속할까요?(Yes/No) : ')
    if answer=="no":
        break
```

리스트를 이용하여 최대/최소값 구하기



▶ (6주차)세 수의 최대값

```
a=int(input("첫번째 수 :"))
b=int(input("두번째 수 :"))
c=int(input("세번째 수 :"))
max=a
if max<b:
    max=b
if max<c:
    max=c
print("%d,%d,%d 중 최댓값은 %d입니다"%(a,b,c,max))
```

▶ 10개의 값 중에서 최대값 구하기

- 값이 여러 개일 경우 리스트를 이용하여 값을 저장하고
- 인덱스를 이용하여 각각의 값을 비교하는 방식으로 최대값을 구할 수 있다

random 수를 리스트의 item으로 채워보자

```
score=[90,43,76,71,89,54,65,29,97,65]
```

```
max=0
```

```
for i in range(len(score)):
```

```
    if score[i]>max:
```

```
        max=score[i]
```

```
print("최고 높은 점수는 %d입니다"%max)
```

▶ 10개의 값 중에서 최대값 구하기

```
import random
score=[ ]
for i in range(10)
    score.append(random.randint(1,99))
print(score)

max=0

for i in range(len(score)):
    if score[i]>max:
        max=score[i]

print("최고 높은 점수는 %d입니다"%max)
```

리스트를 이용하여 등수 구하기



▶ (6주차) 등수구하기

```
a=int(input("input number 1 :"))
b=int(input("input number 2 :"))
c=int(input("input number 3 :"))
score=int(input(" %d,%d,%d 중 등수를 알고싶은 점수는? : " %(a,b,c)))
rank=1
if a>score:
    rank=rank+1
if b>score:
    rank=rank+1
if c>score:
    rank=rank+1
print("%d는 %d등입니다 " %(score,rank))
```


▶ 10개의 값 중에서 등수 구하기

- 10개의 점수리스트 생성
- 사용자로부터 등수를 알고 싶은 점수를 입력받는다
- 해당점수가 리스트에 없을 경우 메시지 출력
- 해당점수가 리스트에 있을 경우 10명 중 몇 등인지 알려준다

▶ 10개의 값 중에서 등수 구하기

random 수를 리스트의 item으로 채워보자

```
score=[45,65,37,98,87,34,99,34,77]
ask=int(input("등수를 알고싶은 점수를 입력하시오"))
rank=1
if ask not in score:
    print("그런 점수는 없습니다")
else:
    for i in range(len(score)):
        if ask<score[i]:
            rank=rank+1
    print('%d점은 우리 반에서 %d등입니다'%(ask,rank))
```

▶ 10개의 값 중에서 등수 구하기

```
import random
score=[ ]
for i in range(10)
    score.append(random.randint(1,99))
print(score)
ask=int(input("등수를 알고싶은 점수를 입력하시오"))
rank=1
if ask not in score:
    print("그런 점수는 없습니다")
else:
    for i in range(len(score)):
        if ask<score[i]:
            rank=rank+1
    print('%d점은 우리 반에서 %d등입니다'%(ask,rank))
```

감사합니다 :)



인하대학교
INHA UNIVERSITY

