

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 파이썬 표준 데이터 자료형

- 파이썬은 다섯 표준 데이터 유형이 있다:
  - ▣ Numbers
  - ▣ String
  - ▣ List
  - ▣ Tuple
  - ▣ Dictionary

- 변수에 어떤 종류의 자료도 저장할 수 있다

```
x = 10  
print("x =", x)  
x = 3.14  
print("x =", x)  
x = "Hello World!"  
print("x =", x)
```

```
x = 10  
x = 3.14  
x = Hello World!
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

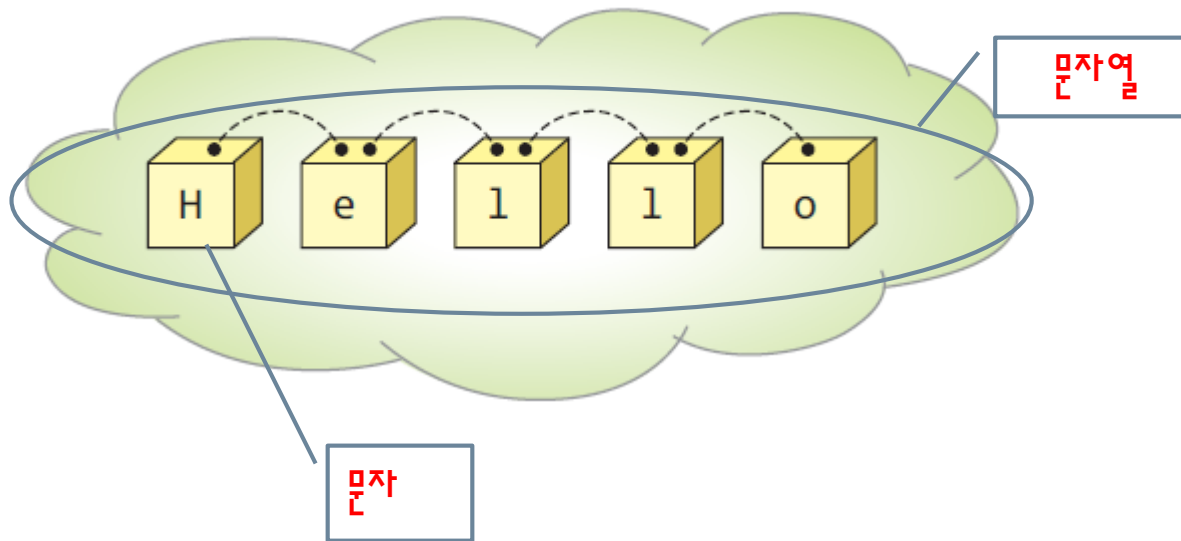
- 문자열 인덱스

- ▣ 문자의 나열 (String)
  - "Hello Python Programming...!"
  - "안녕하세요"
  
- ▣ 큰따옴표/작은따옴표 사용
  - "<글자>" / '<글자>'

```
print("안녕하세요")  
print('안녕하세요')
```

- 안녕하세요 출력

- 문자열(string)은 문자들의 나열(sequence of characters)이다.



# 문자열 - 인덱싱(Indexing)

7/36

- 문자열에서 원하는 위치에 있는 문자를 마음대로 꺼낼 수 있는데 이를 인덱싱(Indexing)이라 함
  - ▣ 문자열의 인덱싱은 0부터 시작함
  - ▣ 문자는 ' '로 표현

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
p	y	t	h	o	n		i	s		v	e	r	y		p

```
var = "python is very powerful"
print(var[4])
print(var[11])
print(var[0])
print(var[1])
```



o  
e  
p  
y

문자열은 인덱싱을 통한 요소의 수정은 할 수 없다!!



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 문자열 슬라이싱



# 개별 문자 추출(슬라이싱)

9/36

- 문자열에서 개별 문자들을 추출하려면 -> 인덱스라는 번호를 사용한다.

형식 : 변수명[시작인덱스:종료인덱스:증가폭]  
(종료 인덱스는 추출되지 않음)

						[6:10]					
0	1	2	3	4	5	6	7	8	9	10	11
M	o	n	t	y		P	y	t	h	o	n
-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
[-12:-7]											

```
s = "Monty Python"  
print(s[6:10])
```

Pyth

# 개별 문자 추출(슬라이싱)

10/36

						[6:10]						
0	1	2	3	4	5	6	7	8	9	10	11	
M	o	n	t	y		P	y	t	h	o	n	
-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	
[-12:-7]												

시작인덱스를 생략한 경우

```
s = "Monty Python"  
print(s[:4])
```

Mont

종료인덱스를 생략한 경우

```
s = "Monty Python"  
print(s[6:])
```

Python

## □ 증가폭

```
s = "123456789"  
print(s[::2])
```

13579

- 증감폭을 음수로 지정하면 역순으로 데이터를 가져옵니다.

- print (nums[::-1])

```
s = "123456789"  
print(s[::-1])
```

987654321



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 문자열 슬라이싱 실습문제

- 다음과 같은 문자열이 있을 경우 슬라이싱을 이용 다음과 같은 값을 출력하세요
  - ▣ `today = "20011031Rainy"`
  - ▣ 출력값
    - 4자리 연도
    - 2자리 월
    - 2자리 일
    - 날씨

<b>today</b>	<b>2</b>	<b>0</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>0</b>	<b>3</b>	<b>1</b>	<b>R</b>	<b>a</b>	<b>i</b>	<b>n</b>	<b>y</b>
<b>인덱스</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>

## □ 문자열을 반복하려면 -> \* 연산자

```
message = " Congratulations!"  
print(message*3)
```

--- 실행결과 ----

Congratulations!Congratulations!Congratulations!

```
print("="*50)
```

--- 실행결과 ----

=====

- 문자열 내의 특정한 값을 바꿔야 할 경우가 있을 때 이것을 가능하게 해주는 것이 바로 문자열 포매팅 기법임

코드	설명
%s	문자열 (String)
%c	문자 1개(character)
%d	정수 (Integer)
%f	부동소수 (floating-point)

## □ 숫자 대입

- ▣ 형식 : 문자열 %숫자 or 변수
- ▣ 변수로 대체 가능

```
num=3  
str ="I eat %d apples." %num  
print(str)
```

I eat 3 apples.

## □ 문자열 대입

- ▣ 형식 : 문자열 %문자열 or 변수
- ▣ 변수로 대체 가능

```
five="5"  
str ="I eat %s apples." %five  
print(str)
```

I eat 5 apples.



## □ 한개이상의 변수 포맷팅


```
three=3  
four=4  
str ="I eat %d apples and %d oranges" %(three,four)  
print(str)
```

I eat 3 apples and 4 oranges

# Lab: 연, 월, 일을 합하여 출력하기

18/36

- 문자열을 저장하는 변수를 사용하여 사용자가 입력하는 오늘의 연도, 월, 일을 모두 합하여 화면에 출력하는 프로그램을 작성해 보자.(포매팅 이용)



```
오늘의 연도를 입력하시오: 2016
오늘의 월을 입력하시오: 12
오늘의 일을 입력하시오: 25
오늘은 2016년 12월 25일입니다.
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

3장 자료형

□ 리스트란?

- 리스트(list): 여러 개의 자료들을 모아서 하나의 묶음으로 저장하는 것
- 모든 자료형 가능
- 형식 :   리스트명 = [요소1, 요소2, 요소3, ...]

```
a = []  
b = [1, 2, 3]  
c = ['Life', 'is', 'too', 'short']  
d = [1, 2, 'Life', 'is']
```



한림대학교 SW중심대학

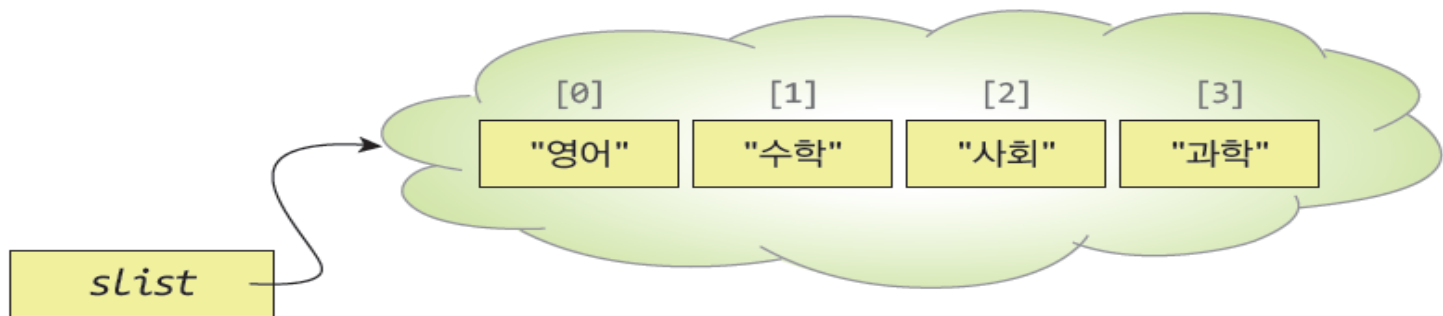
# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 형식과 인덱스

## □ 인덱싱

```
slist = ['영어', '수학', '사회', '과학']  
print(slist)
```



```
['영어', '수학', '사회', '과학']
```

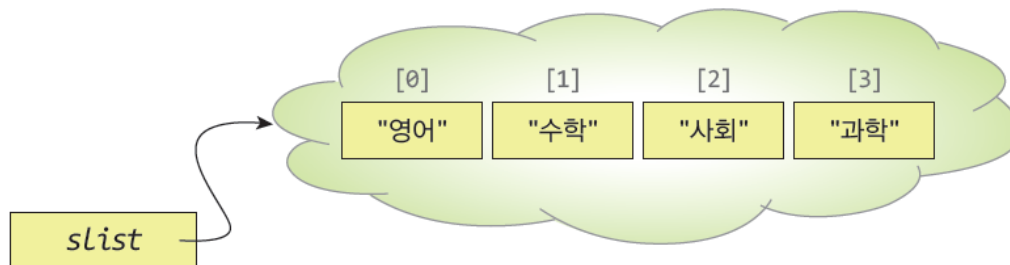
## □ 요소에 접근하기 - 인덱스를 이용해서 접근

### ▣ 형식

변수명[인덱스]

```
slist = ['영어', '수학', '사회', '과학']  
print(slist[0])
```

영어





한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 값 수정하기



## □ 인덱스를 이용 요소값 수정

- ▣ 리스트는 문자열과는 다르게 인덱싱을 통하여 리스트의 요소를 변경가능
- ▣ 하나의 요소 값 수정

```
a = [1, 2, 3]  
a[2] = 4  
print(a)
```



```
[1, 2, 4]
```

- ▣ 연속된 요소의 값 수정

```
a = [1, 2, 4]  
lVal=a[1:2]  
print(lVal)  
a[1:2] = ['a', 'b', 'c']  
print(a)
```



```
[2]  
[1, 'a', 'b', 'c', 4]
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 값 삭제하기

- 리스트 요소 삭제
  - ▣ [] 사용해 리스트 요소 삭제하기

```
a = [1, 'a', 'b', 'c', 4]  
a[1:3] = []  
print(a)
```



```
[1, 'c', 4]
```

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 관련 연산자

## □ 연산자

### ▣ + : 리스트 연결

```
a = [1, 2, 3]  
b = [4, 5, 6]  
print(a + b)
```



```
[1, 2, 3, 4, 5, 6]
```

### ▣ \* : 리스트반복

```
a = [1, 2, 3]  
print(a * 3)
```



```
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

### ▣ 리스트 길이 구하기



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - append 함수

## □ append()

- 리스트에 값을 하나씩 추가하는 것
- 뒤쪽으로 추가됨
- 사용법 – **append(값 or 변수)**의 매개변수로 추가값을 넣어줌
  - 변수, 상수 모두 가능

```
a = [1, 2, 3]
a.append(4)
print(a)
```

```
[1, 2, 3, 4]
```

```
a = [] ← 리스트형 변수로 선언
one=1
a.append(one)
a.append(2)
a.append(3)
a.append(4)
print(a)
```

```
[1, 2, 3, 4]
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - expend 함수



## □ **extend()**

- ▣ 여러개의 값을 추가 가능- `append()` 는 하나씩만 추가 가능
- ▣ 뒤쪽으로 추가됨
- ▣ 사용법
  - `extend(리스트타입의 값)`

```
a = [1,2,3]
a.extend([4,5])
print(a)
b = [6, 7]
a.extend(b)
print(a)
```

```
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5, 6, 7]
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - insert 함수

## □ insert()

- ▣ 지정한 위치에 값을 추가시 사용
- ▣ 사용법
  - insert(위치, 값)

```
a = [1, 2, 3]  
a.insert(0, 4)  
print(a)
```

```
[4, 1, 2, 3]
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - remove 함수

## □ remove()

- ▣ 리스트의 요소를 제거
- ▣ 제거하려는 값이 리스트에 여러개가 존재하면 가장 앞에있는 요소부터 제거

## ▣ 사용법

### ■ remove(값)

```
a = [1, 2, 3, 5, 4, 3]  
a.remove(3)  
print(a)
```

```
[1, 2, 5, 4, 3]
```

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - index 함수

## □ index()

□ 리스트에서 찾으려는 값의 위치를 반환한다.

□ 사용법

- Index(인수로는 찾으려는 요소의 값이 들어가며, 두번째 인수에는 탐색이 시작하는 위치, 세번째 인수에는 탐색이 종결)
  - 두번째 세번째 인자는 생략가능
- 리턴값 : 찾으려는 요소값의 위치

```
a = [1,2,3]
loc=a.index(3)
print(loc)
print(a.index(1))
```

```
2
0
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - count 함수



- count()
  - ▣ count 함수를 통해서 요소 개수를 확인
  - ▣ 사용법
    - count(확인할 요소값)
    - 리턴값 : 개수

```
a = [1,2,3,1]  
tno =a.count(1)  
print(tno)
```

2



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - sort 함수

## □ sort()

- 리스트의 요소들을 정렬

- 사용법

  - sort(reverse=True)

    - 정순으로 정렬할 경우 reverse=True 생략가능

```
a = [1, 4, 3, 2]
a.sort()
print(a)
```

```
[1, 2, 3, 4]
```

```
a = ['a', 'c', 'b']
a.sort(reverse=True)
print(a)
```

```
['c', 'b', 'a']
```



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - len 함수

## □ len()

- ▣ 리스트의 길이를 구하는 함수
- ▣ 문자열에서도 사용 가능
- ▣ 사용법
  - len(리스트변수명)
  - 리턴값 : 리스트 요소의 갯수

```
a = [1, 2, 3]  
a_len=len(a)  
print(a_len)
```



3



한림대학교 SW중심대학

# 생초보를 위한 파이썬 프로그래밍

## 3장 자료형

- 리스트 - sum, min, max 함수

## □ sum()

- 요소의 총합의 값을 구하는 함수

```
a = [1, 2, 3]  
total=sum(a)  
print(total)
```

→ 6

## □ max()

- 리스트 내의 최대값을 구하는 함수

```
a = [1, 2, 3]  
mx=max(a)  
print(mx)
```

→ 3

## □ min()

- 리스트 내의 최소값을 구하는 함수

```
a = [1, 2, 3]  
mn=min(a)  
print(mn)
```

→ 1

# Lab: 친구들의 리스트 생성하기



48/36

- 제일 친한 친구 5명의 이름을 리스트에 저장했다가 출력하는 프로그램을 작성하자.

```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
친구의 이름을 입력하시오: 홍길동
친구의 이름을 입력하시오: 강감찬
친구의 이름을 입력하시오: 이순신
친구의 이름을 입력하시오: 권율
친구의 이름을 입력하시오: 정약용
['홍길동', '강감찬', '이순신', '권율', '정약용']
>>> |
```

Ln: 23 Col: 4



# Solution

49/3

```
friend_list = [ ]  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
print(friend_list)
```

# Solution

50/3

```
friend_list = [ ]  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
friend = input("친구의 이름을 입력하시오: ")  
friend_list.append(friend)  
  
print(friend_list)
```