제8장 문자열

- 학습목표
  - 문자열을 익힌다
  - 문자열 조작함수를 활용할 수 있다.
  - 문자열 활용법을 익힌다

문자열 기본

- 문자열
  - “ ” 혹은 ‘ ’ 로 묶어 표현
  - 리스트와 비슷한 부분이 많음
    - 리스트명[인덱스], 리스트명[시작:끝값+1], 리스트명[시작:]
  - 예: “안녕하세요“ , ‘파이썬’
  - 예

```
s = “파이썬최고”
s[0]
s[1:3]
s[3:]
```

⇨

```
‘파’
‘이썬’
‘최고’
```

[0]	[1]	[2]	[3]	[4]
파	이	썬	최	고

문자열 기본

- 문자열 연산 : +(연결), *(반복)
  - 예

```
ss = ‘파이썬’ + ‘최고‘
ss = ‘파이썬’ * 3
```

⇨

```
‘파이썬최고’
‘파이썬파이썬파이썬‘
```

- 문자열 길이 : len(문자열)
  - 예

```
s = '파이썬짱!'
slen = len(s)
print("문자열 : %s" % s)
print("문자열길이: %d " % slen)

for i in range(0, slen) :
    print(s[i] + '$', end = '')
```

⇨

```
문자열 : 파이썬짱!
문자열길이: 5
파$이$썬$짱$!$
```


문자열 기본

- 예: 입력한 문자열 거꾸로 출력하기

```
## 변수 선언 부분 ##
inStr, outStr = "", ""
count, i = 0, 0

## 메인 코드 부분 ##
inStr = input("문자열? ")
count = len(inStr)
print("문자길이 : %d" % count)

for i in range(0, count) :
    outStr += inStr[count - (i + 1)]

print("내용을 거꾸로 출력 --> %s" % outStr)
```

[0]	[1]	[2]
파	이	썬

```
문자열? 파이썬
문자길이 : 3
내용을 거꾸로 출력 --> 썬이파
```

문자열 함수

- 대소문자 변환
  - upper() : 소문자 → 대문자
  - lower() : 대문자 → 소문자
  - swapcase() : 대소문자 상호 변환
  - title() : 첫글자만 대문자로 변환
  - 예

```
s = 'Python is Easy'

s.upper()
s.lower()
s.swapcase()
s.title()
```

```
PYTHON IS EASY
python is easy
pYTHON IS eASY
Python Is Easy
```


문자열 함수

- 문자열 찾기

함수	설명
count("지정문자열")	지정문자열이 몇 개 있는지 개수 세기
find("지정문자열")	지정문자열이 왼쪽끝(0번 위치)에서부터 몇번째에 위치하는지 찾기
rfind("지정문자열")	지정문자열이 오른쪽끝에서부터 몇번째에 위치하는지 찾기. 지정문자열없으면 -1 반환
index("지정문자열")	지정문자열이 없으면 오류 발생
rindex("지정문자열")	지정문자열이 마지막 나타나는 위치 반환
startswith("지정문자열")	지정문자열로 시작하면 true
endswith("지정문자열"	지정문자열로 끝나면 true

문자열 함수

- 예제

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
파	이	썬		기	초		난	생	처	음		파	이	썬		혼	공	^	^

```
s.count('파이썬') : 2
s.find('파이썬'): 0
s.rfind('파이썬'): 12
s.find('파이썬',5): 12
s.find('가을'): -1
```

```
s.index('파이썬'): 0
s.rindex('파이썬'): 12
s.index('파이썬',5): 12
s.startswith('^^'): 0
s.startswith('기초',4): 1
s.endswith('^^'): 1
```


문자열 함수

- 예 : 문자열이 괄호로 감싸있지않으면 괄호로 감싸주는 프로그램

```
s = input("입력 문자열 : ")
print("출력 문자열 : ", end = '')

if s.startswith('(') == False :  # ( 로 시작되지 않으면
    print("(", end = '')

print(s, end = '')

if s.endswith(')') == False :  # ) 로 끝나지 않으면
    print(")", end = '')
```

```
입력 문자열 : 파이썬
출력 문자열 : (파이썬)
```

문자열 함수

- 문자열 공백 삭제
  - 문자열 중간의 공백은 삭제되지 않음
  - 문자열.strip() : 문자열의 앞뒤 공백 삭제
  - 문자열.rstrip() : 문자열의 오른쪽 공백 삭제
  - 문자열.lstrip() : 문자열의 왼쪽 공백 삭제
  - 예

```
s1 = '  파 이 썬  '
print(s1.strip());
print(s1.rstrip());
print(s1.lstrip());
```

⇨

```
파 이 썬
  파 이 썬
파 이 썬
```


문자열 함수

- 예: 문자열 중간의 공백 삭제

```
inStr = "  한글 Python 프로그래밍  "
outStr = ""

for i in range(0, len(inStr)) :
    if inStr[i] != ' ' :
        outStr += inStr[i]

print("원래 문자열 ==> " + '[' + inStr + ']')
print("공백 삭제 문자열 ==> " + '[' + outStr + ']')
```

```
원래 문자열 ==> [  한글 Python 프로그래밍  ]
공백 삭제 문자열 ==> [한글Python프로그래밍]
```


문자열 함수

- 문자열 변경 : 문자열.replace('기존문자열', '새문자열')
  - 예

```
s = '열심히 파이썬 공부 중~~'
print(s.replace('파이썬', 'Python'));
```

⇨

```
열심히 Python 공부 중~~
```

  - 예 : 사용자가 입력한 문자열 중에서 o를 $로 변경

```
s = input("입력 문자열 : ")
print("출력 문자열 : ", end = '')
print(s.replace('o', '$'))
```

⇨

```
입력 문자열 : Cook
출력 문자열 : C$$k
```

```
s = input("입력 문자열 : ")
print("출력 문자열 : ", end = '')
for i in range(0, len(s)) :
    if s[i] != 'o' :  print(s[i], end = '')
    else :            print('$', end = '')
```


문자열 함수

- 문자열 분리
  - 문자열을 공백이나 다른 문자나 ₩n로 분리해서 리스트로 반환
  - 문자열.split(), 문자열.split(문자), 문자열.splitlines()
  - 예

```
s1 = '봄 여름 가을 겨울'
print("1) s1 : %s" % s1)

list_s1 = s1.split()
print("--> split() 후 : ", end = '')
print(list_s1)

print("리스트 요소 출력 : ", end = '')
for i in range(0, len(list_s1)) :
    print(list_s1[i], end = ' ')
```

[0]	[1]	[2]	[3]
봄	여름	가을	겨울

```
1) s1 : 봄 여름 가을 겨울
--> split() 후 : ['봄', '여름', '가을', '겨울']
리스트 요소 출력 : 봄 여름 가을 겨울
```

문자열 함수

- 문자열 분리
  - 예 : ':'을 기준으로 문자열 분리

```
s1 = '봄:여름:가을:겨울'
print("1) s1 : %s" % s1)

list_s1 = s1.split(':')
print("--> split() 후 : ", end = '')
print(list_s1)

print("리스트 요소 출력 : ", end = '')
for i in range(0, len(list_s1)) :
    print(list_s1[i], end = ' ')
```

[0]	[1]	[2]	[3]
봄	여름	가을	겨울

```
1) s1 : 봄:여름:가을:겨울
--> split() 후 : ['봄', '여름', '가을', '겨울']
리스트 요소 출력 : 봄 여름 가을 겨울
```


문자열 함수

- 문자열 분리
  - 문자열.splitlines() : ₩n을 기준으로 문자열 분리
    - 예

```
s1 = '봄\n여름\n가을\n겨울'
print("1) s1 : %s" % s1)

list_s1 = s1.splitlines()
print("--> split() 후 : ", end = '')
print(list_s1)

print("리스트 요소 출력 : ", end =
for i in range(0, len(list_s1)) :
    print(list_s1[i], end = ' ')
```

[0]	[1]	[2]	[3]
봄	여름	가을	겨울

```
1) s1 : 봄
여름
가을
겨울
--> split() 후 : ['봄', '여름', '가을', '겨울']
리스트 요소 출력 : 봄 여름 가을 겨울
```


문자열 함수

- 문자열 결합
  - '구분자'.join(문자열): 리스트의 값과 값 사이에 구분자를 넣어서 하나의 문자열로 합쳐줌
  - 예

```
a = ['봄', '여름', '가을', '겨울']
print(a)

result1 = '_'.join(a)
print(result1)

result2 = ''.join(a)
print(result2)
```

```
['봄', '여름', '가을', '겨울']
봄_여름_가을_겨울
봄여름가을겨울
```

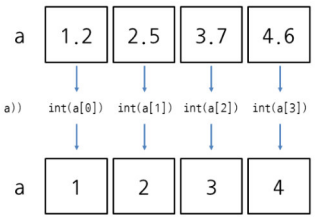

문자열 함수

- map(함수명, 리스트명)
  - 리스트의 요소를 지정된 함수로 처리해주는 함수
  - map은 원본 리스트를 변경하지 않고 새 리스트를 생성
  - 예: 실수로 구성된 리스트의 모든 요소를 정수로 변환

```
a = [1.2, 2.5, 3.7, 4.6]

print(a)

for i in range(0, len(a)):
    a[i] = int(a[i])

print(a)
```

```
a = [1.2, 2.5, 3.7, 4.6]
print(a)

a = list(map(int, a))
print(a)
```

```
[1.2, 2.5, 3.7, 4.6]
[1, 2, 3, 4]
```



문자열 함수

- 문자열 정렬하기, 채우기
  - 문자열.center(숫자) : 전체 자릿수를 잡은 후 문자열을 가운데에 배치
  - 문자열.ljust(숫자) : 왼쪽에 붙여서 문자열 출력
  - 문자열.rjust(숫자) : 오른쪽에 붙여서 문자열 출력
  - 문자열.zfill(숫자) : 오른쪽으로 붙여쓰고 왼쪽 빈공간은 0으로 채움
  - 예

```
s = '파이썬'
print(s.center(10));
print(s.center(10, '-'));
print(s.ljust(10));
print(s.rjust(10));
print(s.zfill(10));
```

```
 파이썬
---파이썬----
파이썬
      파이썬
0000000파이썬
```


문자열 함수

- 문자열 구성 파악 : isdigit(), isalpha(), isalnum(), islower(), isupper(), isspace() → 결과는 True 혹은 False
  - 예

```
'1234'.isdigit()
'abcd'.isalpha()
'abc123'.isalnum()
'abcd'.islower()
'ABCD'.isupper()
'   '.isspace()
```

```
True
True
True
True
True
True
```