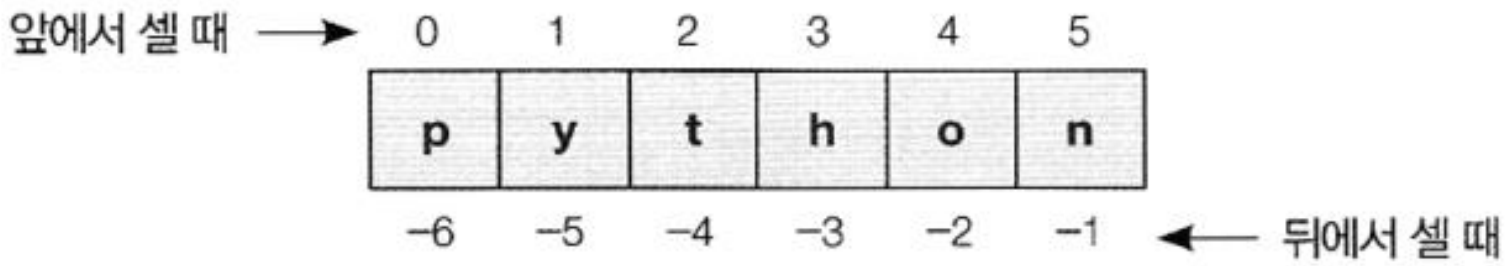
문자열 관리

문자열 분리

❖ 첨자

- ㅇ 문자열[정수] 0부터 인덱싱
- ㅇ 문자열[-정수] 끝에서부터 인덱싱

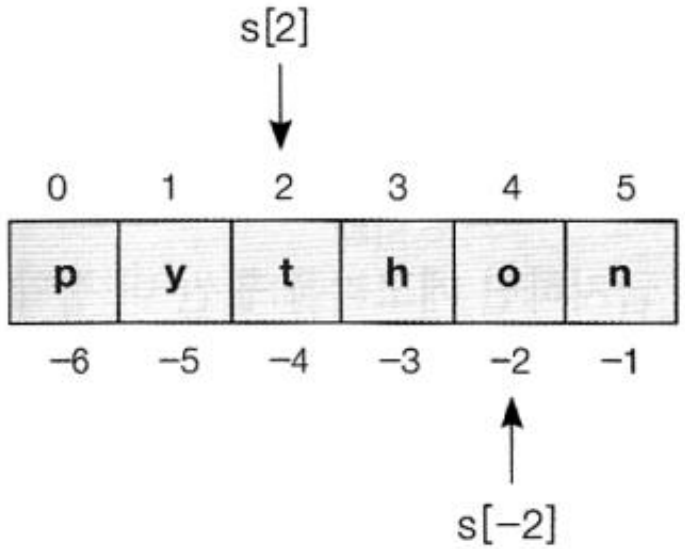


문자열 분리

❖ 첨자

```
s = "python"
print(s[2])
print(s[-2])
```

```
t
o
```



문자열 분리

❖ 첨자

```
s = "python"
for c in s:
    print(c, end = ",")
```

```
p,y,t,h,o,n,
```

```
s = "python"
for i in range(len(s)):
    print(s[i], end = ",")
```

```
p,y,t,h,o,n,
```

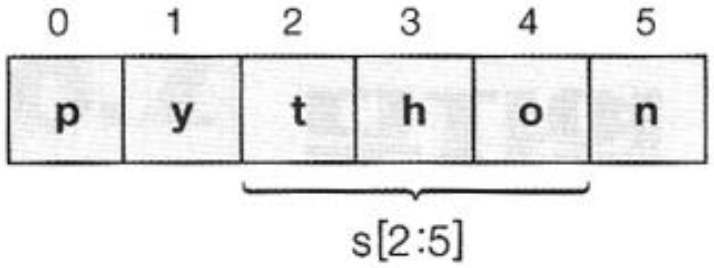
문자열 분리

❖ 슬라이싱

- 문자열[begin:end:step]
  - step: 음수이면 뒤에서부터 진행

```
s = "0123456789"
print(s[2:5])
print(s[3:])
print(s[:4])
```

```
234
3456789
0123
```

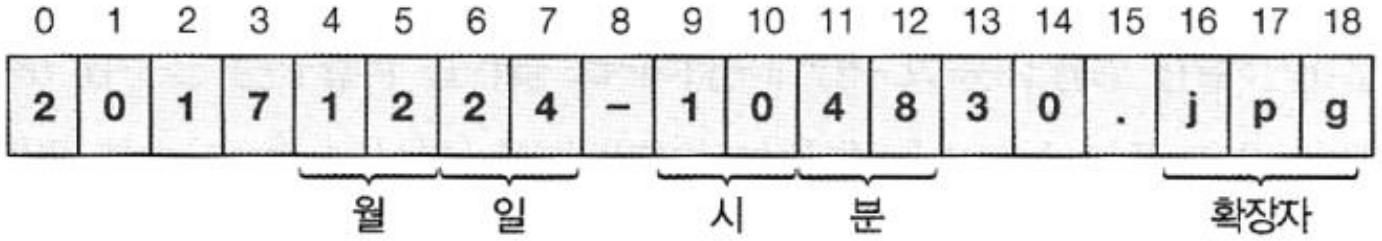


문자열 분리

❖ 슬라이싱

```
file = "20200101-104830.jpg"
print("촬영 날짜" + file[4:6] + "월" + file[6:8] + "일")
print("촬영 시간" + file[9:11] + "월" + file[11:13] + "일")
print("확장자" + file[-3:])
```

```
촬영 날짜01월01일
촬영 시간10월48일
확장자jpg
```

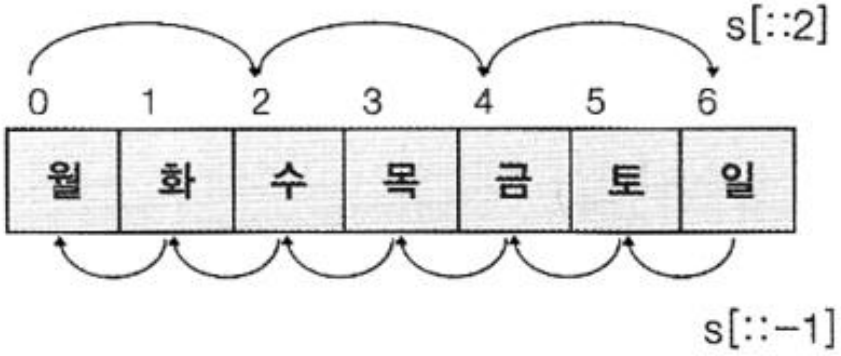


문자열 분리

❖ 슬라이싱

```
dates = "월화수목금토일"
print(dates[::2])
print(dates[::-1])
```

월수금일
일토금목수화월



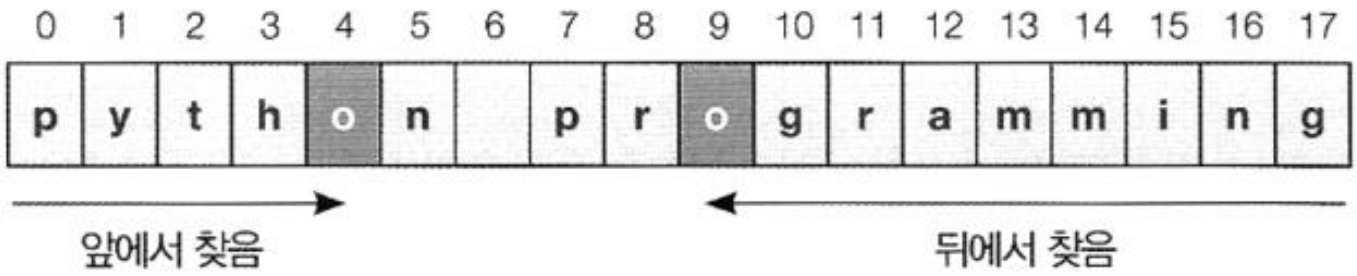
문자열 메서드

❖ 검색

- .find(str): str 문자열을 찾아 인덱스 반환, 없으면 -1 반환
- .rfind(str): 뒤에서 str 문자열을 찾아 인덱스 반환, 없으면 -1 반환
- .index(str): find()와 동일, 없으면 예외 발생
- .count(str): str 문자열이 몇번 등장하는지 리턴

```
s = "python programming"
print(len(s))
print(s.find('o'))
print(s.rfind('o'))
print(s.index('r'))
print(s.count('n'))
```

```
18
4
9
8
2
```



문자열 메서드

❖ 조사

- 단어 in 문자열 -> bool
- 단어 not in 문자열 -> bool
- .startswith(str) -> bool
- .endswith(str) -> bool

```
s = "python programming"

print('a' in s)
print('z' in s)
print('pro' in s)
print('x' not in s)
```

```
True
False
True
True
```

문자열 메서드

❖ 조사

```
name = "홍길동"

if name.startswith("홍"):
    print("홍씨입니다.")

if name.startswith("김"):
    print("김씨입니다.")

file = "figure.jpg"
if file.endswith(".jpg"):
    print("JPG 그림 파일입니다.")
```

```
홍씨입니다.
JPG 그림 파일입니다.
```

문자열 메서드

❖ 기타 메서드

- isalpha
- islower
- isupper
- isspace
- isalnum
- isdecimal
- isdigit
- isnumeric
- isidentifier
- isprintable

```
height = input("키 : ")
if height.isnumeric():
    print("키 = ", height)
else:
    print("숫자만 입력하세요")
```

```
키 : abc
숫자만 입력하세요
```

문자열 메서드

❖ 변경

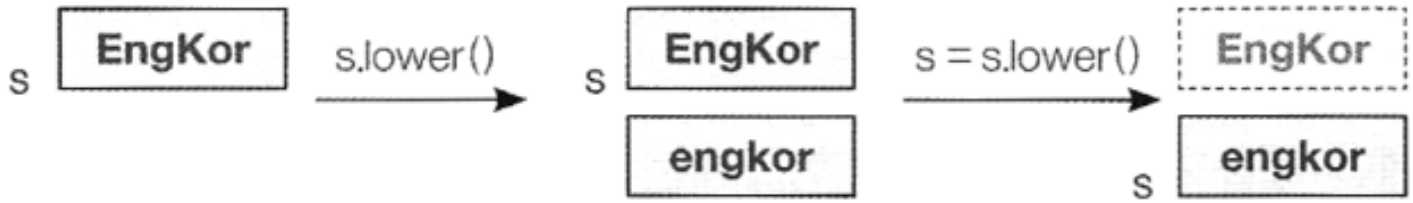
- .lower()
- .upper()
- .swapcase() : 대문자는 소문자로, 소문자는 대문자로 변환
- .capitalize() : 첫글자는 대문자 나머지는 모두 소문자로 변환
- .title(): 모든 단어의 첫 글자를 대문자로 나머지는 소문자로 변환
- .strip() : 좌우에 있는 공백을 제거
- .lstrip(): 왼쪽에 있는 공백을 제거
- .rstrip(): 오른쪽에 있는 공백을 제거

문자열 메서드

❖ 변경

```
s = "Good morning. my love KIM."

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

```
good morning. my love kim.
GOOD MORNING. MY LOVE KIM.
gOOD MORNING. MY LOVE kim.
Good morning. my love kim.
Good Morning. My Love Kim.
```

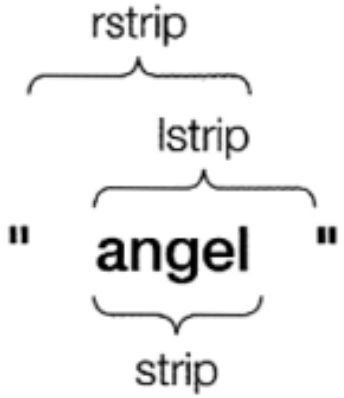


문자열 메서드

❖ 변경

```
s = "    angel    "
print(s + "님")
print(s.strip() + "님")
print(s.lstrip() + "님")
print(s.rstrip() + "님")
```

```
    angel    님
angel님
angel    님
    angel님
```



문자열 메서드

❖ 분할

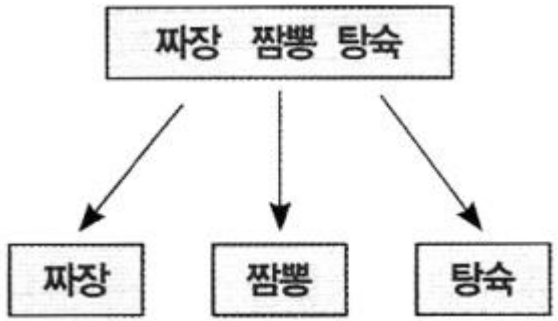
- .slpit(구분자)
  - 구분자를 기준으로 단어를 분리하여 리스트로 리턴, 디폴트는 공백
- .splitlines()
  - 개행 문자를 기준으로 분리. 개행문자만 있는 경우 비어있는 문자열로 처리
- 결합문자열.join(문자열)
  - 글자들을 결합문자열로 연결하여 하나의 문자열로 리턴

문자열 메서드

❖ 분할

```
s = "짜장 짬뽕 탕수육"
print(s.split())

s2 = "서울->대전->대구->부산"
cities = s2.split("->")
print(cities)

for city in cities:
    print(city)
```

```
['짜장', '짬뽕', '탕수육']
['서울', '대전', '대구', '부산']
서울
대전
대구
부산
```



문자열 메서드

❖ 분할

```
trabler ="""
강나루 거너서
밀밭 길을

구름에 달 가듯이
가는 나그네
"""

poet = trabler.splitlines()
for line in poet:
    print(line)
```

```
강나루 거너서
밀밭 길을

구름에 달 가듯이
가는 나그네
```

문자열 메서드

❖ 분할

```
s ="._."
print(s.join("대한민국"))
```

대._.한._.민._.국

문자열 메서드

❖ 대체

- .replace(기존문자열, 대체문자열)
  - 기존 문자열을 대체 문자열로 대체
- .center(폭숫자)
  - 좌우에 공백을 채워 폭숫자만큼 문자열 길이를 맞춤
- .ljust(폭숫자)
  - 왼쪽에 공백을 채워 폭숫자만큼 문자열 길이를 맞춤
- .rjust(폭숫자)
  - 오른쪽에 공백을 채워 폭숫자만큼 문자열 길이를 맞춤

문자열 메서드

❖ 대체

```
s = "독도는 일본땅. 대마도도 일본땅"
print(s)
print(s.replace("일본", "한국"))
```

```
독도는 일본땅. 대마도도 일본땅
독도는 한국땅. 대마도도 한국땅
```

```
message = "안녕하세요"
print(message.center(30))
print(message.ljust(30))
print(message.rjust(30))
```


```
            안녕하세요
안녕하세요
                        안녕하세요
```

문자열 메서드

❖ 대체

```
trabler ="""
강나루 거너서
밀밭 길을

구름에 달 가듯이
가는 나그네
"""

poet = trabler.splitlines()
for line in poet:
    print(line.center(30))
```

```
          강나루 거너서
           밀밭 길을

         구름에 달 가듯이
           가는 나그네
```

포맷팅

❖ 포맷팅

- %d 정수
- %f 실수
- %s 문자열
- %c 문자 하나
- %h 16진수
- %o 8진수
- %% % 문자
- %[-]폭[.유효자리수]서식, 폭에는 소수점에 포함, 반올림 발생

포맷팅

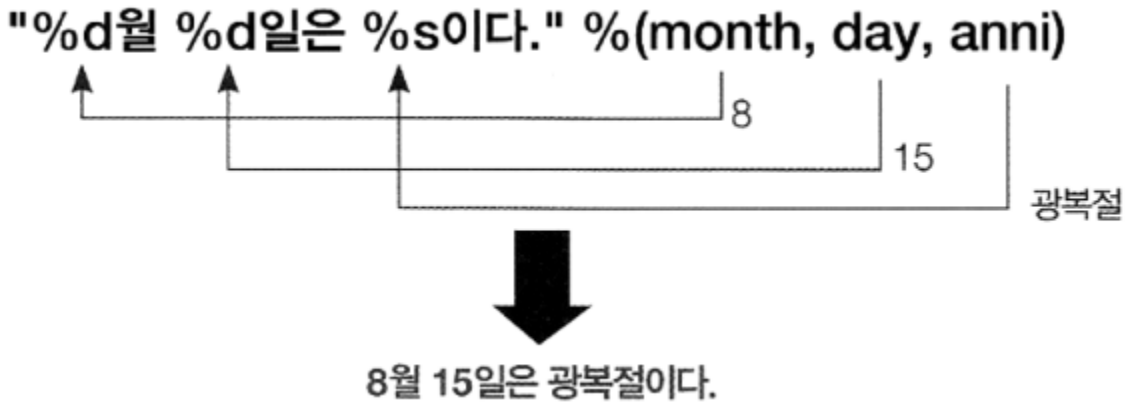
❖ 포맷팅

```
price = 500
print("궁금하면 " + str(price) + "원!")
```

궁금하면 500원!

```
mont = 8
day = 15
anni = "광복절"
print("%d월 %d일은 %s이다." % (mont, day, anni))
```

8월 15일은 광복절이다.



포맷팅

❖ 포맷팅

```
value = 123
print("###%d###"%value)
print("###%5d###"%value)
print("###%10d###"%value)
print("###%-10d###"%value)
print("###%1d###"%value)
```

```
###123###
###  123###
###       123###
###123       ###
###123###
```

포맷팅

❖ 포맷팅

```
price = [30, 13500, 2000]
for p in price :
    print("가격 : %d원"%p)
print()
for p in price :
    print("가격 : %7d원"%p)
print()
for p in price :
    print("가격 : %-7d원"%p)
```

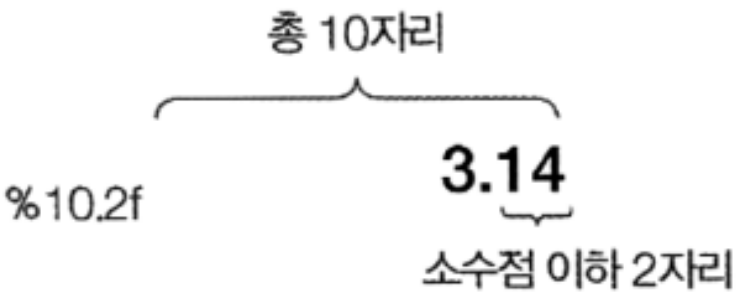
```
가격 : 30원
가격 : 13500원
가격 : 2000원

가격 :      30원
가격 :   13500원
가격 :    2000원

가격 : 30     원
가격 : 13500  원
가격 : 2000   원
```

포맷팅

❖ 포맷팅

```
f = 123.1234567
print("%10f"%f)
print("%10.8f"%f)
print("%10.5f"%f)
print("%10.2f"%f)
print("%.2f"%123.126)
```

```
123.123457
123.12345670
 123.12346
    123.12
123.13
```



포맷팅

❖ 선형 포맷팅

- "{[:포맷문자열]} ".format(값...)
- "{인덱스[:포맷문자열]} ".format(값...)
- "{변수명[:포맷문자열]} ".format(값...)

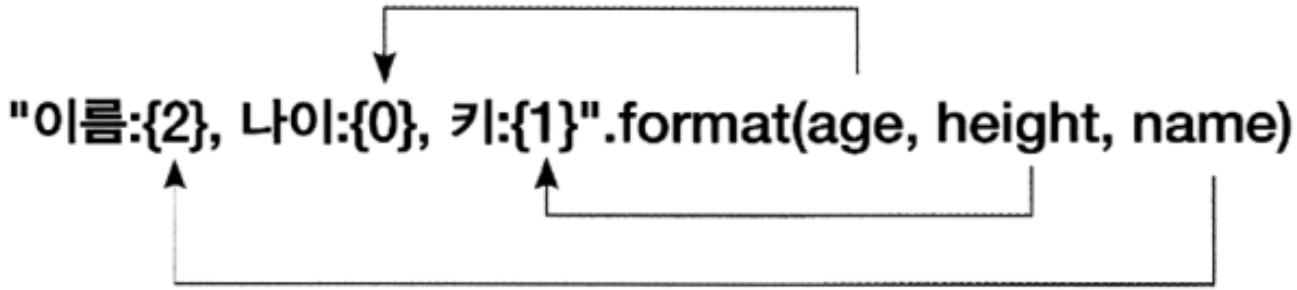
```
name = "한결"
age = 16
height = 162.5
print("이름:{}, 나이: {}, 키: {}".format(name, age, height))
print("이름:{:s}, 나이: {:d}, 키: {:f}".format(name, age, height))
print("이름:{:4s}, 나이: {:3d}, 키: {:.2f}".format(name, age, height))
```

```
이름:한결, 나이: 16, 키: 162.5
이름:한결, 나이: 16, 키: 162.500000
이름:한결  , 나이:  16, 키: 162.50
```

포맷팅

❖ 선형 포맷팅

- ㅇ 위치 지정



포맷팅

❖ 선형 포맷팅

```
name = "한결"
age = 16
height = 162.5
print("이름:{0}, 나이: {1}, 키: {2}".format(name, age, height))
print("이름:{2}, 나이: {1}, 키: {0}".format(height, age, name))
print("이름:{name}, 나이: {age}, 키: {height}".format(
      age=20, height=160.9, name="길동"))
```

```
이름:한결, 나이: 16, 키: 162.5
이름:한결, 나이: 16, 키: 162.5
이름:길동, 나이: 20, 키: 160.9
```

```
boy = {"name": "길동", "age":20, "height":160.9}
print("이름:{0[name]}, 나이:{0[age]}, 키:{0[height]}".format(boy))
```

```
이름:길동, 나이:20, 키:160.9
```

포맷팅

❖ 자리 채움방식

- 채움문자$: 좌우 채움
- 채움문자> : 왼쪽에 채움
- 채움문자< : 오른쪽에 채움

```
name = "길동"
age = 16
height = 162.5
print("이름:{0:$<10s}, 나이:{1:>05d}, 키:{2:!<8.2f}".format(name, age, height))
```

```
이름:길동$$$$$$$$, 나이:00016, 키:162.50!!
```

포맷팅

❖ f-string

- Python 3.6 이상 버전부터 지원
- 문자열 앞에 접두사 f를 붙이고, 중괄호 ({}) 안에 변수를 지정

```
f'{값:자리수/정렬 표현}'
```

```
s = '일'
print(f'나는 하고 싶은 {s}을 하면서 살고싶다.')
```

```
나는 하고 싶은 일을 하면서 살고싶다.
```

```
hour = 7
print(f'나는 {hour}시에 밥 먹을거야')
```

```
나는 7시에 밥 먹을거야
```

포맷팅

❖ f-string

```
# f-string 왼쪽 정렬
s1 = 'left'
result1 = f'|{s1:<10}|'
print(result1)

# f-string 가운데 정렬
s2 = 'mid'
result2 = f'|{s2:^10}|'
print(result2)

# f-string 오른쪽 정렬
s3 = 'right'
result3 = f'|{s3:>10}|'
print(result3)
```

```
|left      |
|   mid    |
|     right|
```

포맷팅

❖ f-string

```
f1 = 12.3456
result1 = f'{f1:.2f}'     # 소수점 셋째자리에서 반올림
print(result1)
```

```
12.35
```