줄 바꾸기 : \n

multiline = '''
Life is
too short
''' → print(multiline) → Life is
too short 출력

문자열 순라이싱 : a[0:4] → 0≤a<4 : a[0] + ··· + a[3]

a[19:] → 19≤a≤끝

a[:17] → 처음≤a<17

a[19:-7] → a[19] + ··· + a[-8]

a[2:8:3] → 2부터 3씩 증가 7까지

a[::-1] → 거꾸로 출력됨

소수점 표현하기 : %0.4f → 4번째 자리까지

%10.4f → 공백 4개 + 3.423

"%-10s" % 'hi' → hi + 공백 8개

"%10s" % 'hi' → 공백 8개 + hi

format 함수 : "I ~ {0} apples. ~ {1}".format(number, 3)

"I ~ {0} apples. ~ {day}".format(10, day=3)

소수점 표현 : y = 3.42134234

"{0:0.4f}".format(y) → '3.4213'

"{0:10.4f}".format(y) → '____3.4213'

위치 알려주기 : a.find('k')

a.index('k')

문자열 삽입 : ",".join('abcd') → a,b,c,d

소.대문자 : a.upper(), a.lower()

왼쪽 공백지우기 : a = " hi ", a.lstrip() → 'hi '

오른쪽 " rstrip → ' hi'

양쪽 " strip → 'hi'

replace
문자열 바꾸기 : a= "Life is too short"
a.replace("Life", "Your leg") → Your leg is too short

split
문자열 나누기 : 1) a= "Life is too short"
a.split( ) ← 공백 기준 문자열 나눔.
a => ['Life', 'is', 'too', 'short']

2) b= "a:b:c:d"
b.split(':')
b=> ['a', 'b', 'c', 'd']

리스트 반복하기 (*) a=[1,2]
a*3 => [1,2,1,2,1,2]

리스트 삭제: del a[1] → a[1] 삭제
del a[2:] → a[2] ··· 끝 삭제

리스트 요소 추가 : a=[1.2.3]
(append)
a.append(4) → [1.2.3.4]
a.append([5,6]) → [1,2,3,4,[5,6]]

리스트 정렬 : a= [1.4.3.2]
(sort)
a.sort() → a =>[1.2.3.4]
a={'a','b','c'}
a.sort() → a=> ['a','b','c']

뒤집기
reverse
a= ['a','c','b']
a.reverse() → ['b','c','a']

리스트에 요소 삽입 : a=[1.2.3]

(insert) a.insert(0.4) ⇒ [4.1.2.3]

a=[1.2.3.1.2, 3] ↑첫번째 나오는 값 삭제

리스트 요소 제거 : a.remove(3) ⇒ [1.2. 1.2,3]

(remove)

리스트 요소 끄집어내기 : pop()은 리스트의 요소를 돌려주고 그 요소 삭제

(pop)

a=[1.2.3] ⋮ a=[1.2.3]

a.pop() ⇒ 3 ⋮ a.pop(1) ⇒ 2

a ⇒ [1.2] ⋮ a ⇒ [1.3]

리스트 요소 개수 세기 a=[1.3.2.1]

(count) a.count(1) ⇒ 2

리스트 확장 a=[1.2.3]

(extend) a.extend([4.5]) ⇒ [1.2.3. 4, 5]

↳ extend 함수는 이루어진 리스트만 올

리스트 정렬 (오름차순) N = sorted(N) or N.sort()

(sort)

리스트를 key로 만들순없음

딕셔너리 : {Key1: Value1, key2: Value2, Key3: Value3...}
ex) dic= {'name':'pey', 'phone':'01023', 1:'hi', 'a':[1,2,3]}

딕셔너리쌍추가
a= {1:'a'}
a[2] = 'b' ← {2:'b'} 추가 ⇒ a={1:'a', 2:'b'}
a['name'] = 'pey' ← {'name':'pey'} 추가 ⇒ a={1:'a', 2:'b', 'name':'pey'}

딕셔너리 쌍 삭제
del a[1] ← key가 1인 쌍 삭제
a⇒ {2:'b', 'name':'pey'}

key → Value
grade= {'a':1, 'b':2}
grade['a'] → 1, grade['b'] → 2

key 리스트 만들기 (Keys 함수)
a= {'a':1, 'b':2, 'c':3}
print(a.keys()) ⇒ dict_keys(['name', 'phone', 'birth'])
keys를 리스트로 받기 → list(a.keys())

Value리스트 : a.values()

key: Values 쌍 얻기 (items)
a.items()
→ dict_items([(,), (,), (,)])

Key: Value쌍 모두 지우기 a.clear() → print(a) ⇒ { }

key로 Value 얻기 (get)
a.get('name') → 'pey'

집합 : set → 중복 X, 순서 X → a=[1,1,2,2]
(중복 요소 제거하기 좋음)
aSet = set(a)
b = list(aSet)

리스트 저장 공간 · 변수

1)
```
a=[1.2.3]
b=a                  =>     a           b
a.append(4)              [1.2.3.4]  [1.2.3.4]
print(a)
print(b)
```

2)
```
a=[1.2.3]                   a         b
b=a[:]          =>    [1.2.3.4]   [1.2.3]
a.append(4)
```

3)
```
from copy import copy
b= copy(a)
```

List Comprehension

리스트 내포

1)
```
a=[1,2,3,4]
result=[num*3 for num in a]
 ↳ [3,6,9,12]
```

[표현식 for 항목 in 반복 가능 객체 if 조건]

(if 활용)

2)
```
result=[num*3 for num in a if num%2==0]
 ↳ [6,12]
```

for문 2개이상 ↓

```
[표현식 for 항목1 in 〃 〃객체1 if조건1
        for 항목2 〃 〃 〃객체2 if조건2
                  ⋮
                                    ]
```

enumerate

리스트 등의 요소를 인덱스와 함께 얻기 위해 사용!

↳ 1)
```
for i, name in enumerate(['A','B','C']):
    print(i, name)
→ 0 A
  1 B
  2 C
```

2)
```
for i, name in enumerate(['A','B','C'], 42)
    print(i, name)
→ 42 A
  43 B
  44 C
```

↑ 시작값

여러개의 값을 입력받는 함수 :

```
def add_many(*args):
    result = 0
    for i in args:
        result = result + i
    return result
```

함수 빠져나가기 : return

while문 빠져나가기: break