AI Programming

Lecture 9

Assignment 7 Solution

```
tic, bar = [], []

n_items = int(input("항목의 수를 입력하시오: "))

for i in range(n_items):
    tic.append( input(f"{i}번째 항목의 이름을 입력하시오: ") )
    bar.append( int(input(f"{i}번째 항목의 막대 길이를 입력하시오: ")) )

for i in range(n_items):
    print(f"{tic[i]}: ", end="")

    for j in range(bar[i]):
        print("*", end="")

    print(f" {bar[i]:2d}개")
```

Preview

- **Ch. 7 리스트, 튜플, 딕셔너리**
  - 7.2 리스트의 기본
    - 7.2.7 리스트 조작 함수
  - 7.3 2차원 리스트
  - 7.4 튜플

7.2 리스트의 기본

7.2.7 리스트 조작 함수

List Methods

- **List methods**

함수	설명	사용법
append()	리스트 맨 뒤에 항목을 추가한다.	리스트명.append(값)
pop()	리스트 맨 뒤의 항목을 빼낸다(리스트에서 해당 항목이 삭제된다).	리스트명.pop()
sort()	리스트의 항목을 정렬한다.	리스트명.sort()
reverse()	리스트 항목의 순서를 역순으로 만든다.	리스트명.reverse()
index()	지정한 값을 찾아 해당 위치를 반환한다.	리스트명.index(찾을값)
insert()	지정된 위치에 값을 삽입한다.	리스트명.insert(위치, 값)
remove()	리스트에서 지정한 값을 삭제한다. 단 지정한 값이 여러 개면 첫 번째 값만 지운다.	리스트명.remove(지울값)

List Methods

- **List methods**

extend( )	리스트 뒤에 리스트를 추가한다. 리스트의 더하기(+) 연산과 기능이 동일하다.	리스트명.extend(추가할리스트)
count( )	리스트에서 해당 값의 개수를 센다.	리스트명.count(찾을값)
clear( )	리스트의 내용을 모두 지운다.	리스트명.clear( )
del( )	리스트에서 해당 위치의 항목을 삭제한다.	del(리스트명[위치])
len( )	리스트에 포함된 전체 항목의 개수를 센다.	len(리스트명)
copy( )	리스트의 내용을 새로운 리스트에 복사한다.	새리스트=리스트명.copy( )
sorted( )	리스트의 항목을 정렬해서 새로운 리스트에 대입한다.	새리스트=sorted(리스트)

List Methods

- **Exercise**

```
1  myList = [30, 10, 20]
2  print("현재 리스트 : %s" % myList)
3
4  myList.append(40)
5  print("append(40) 후의 리스트 : %s" % myList)
6
7  print("pop()으로 추출한 값 : %s" % myList.pop())
8  print("pop() 후의 리스트 : %s" % myList)
9
10 myList.sort()
11 print("sort() 후의 리스트 : %s" % myList)
12
13 myList.reverse()
14 print("reverse() 후의 리스트 : %s" % myList)
15
16 print("20값의 위치 : %d" % myList.index(20))
```

```
18 myList.insert(2, 222)
19 print("insert(2, 222) 후의 리스트 : %s" % myList)
20
```

```
21 myList.remove(222)
22 print("remove(222) 후의 리스트 : %s" % myList)
23
24 myList.extend([77, 88, 77])
25 print("extend([77, 88, 77]) 후의 리스트 : %s" % myList)
26
27 print("77값의 개수 : %d" % myList.count(77))
```

List Methods

- **Exercise**

출력 결과

```
현재 리스트 : [30, 10, 20]
append(40) 후의 리스트 : [30, 10, 20, 40]
pop()으로 추출한 값 : 40
pop() 후의 리스트 : [30, 10, 20]
sort() 후의 리스트 : [10, 20, 30]
reverse() 후의 리스트 : [30, 20, 10]
20값의 위치 : 1
insert(2, 222) 후의 리스트 : [30, 20, 222, 10]
remove(222) 후의 리스트 : [30, 20, 10]
extend([77, 88, 77]) 후의 리스트 : [30, 20, 10, 77, 88, 77]
77값의 개수 : 2
```

List Methods

- myList.sort() vs. sorted(myList)

```
myList = [30, 10, 20]
newList = sorted(myList)
print("sorted() 후의 myList : %s" % myList)
print("sorted() 후의 newList : %s" % newList)
```

출력 결과

```
sorted() 후의 myList : [30, 10, 20]
sorted() 후의 newList : [10, 20, 30]
```

List Methods

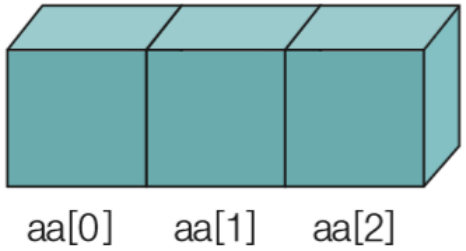
- **Deep copy vs. Shallow copy**
  - Immutable vs. mutable
    - Immutable: int, float, bool, string, tuple
    - Mutable: list, dictionary
  - Shallow copy
    - `b = a`
  - Deep copy
    - `b = a.copy()`
    - `import copy; b = copy.copy(a)`

7.3 2차원 리스트

2D List

- **2D list**
  - List of lists

```
aa = [10, 20, 30]
```

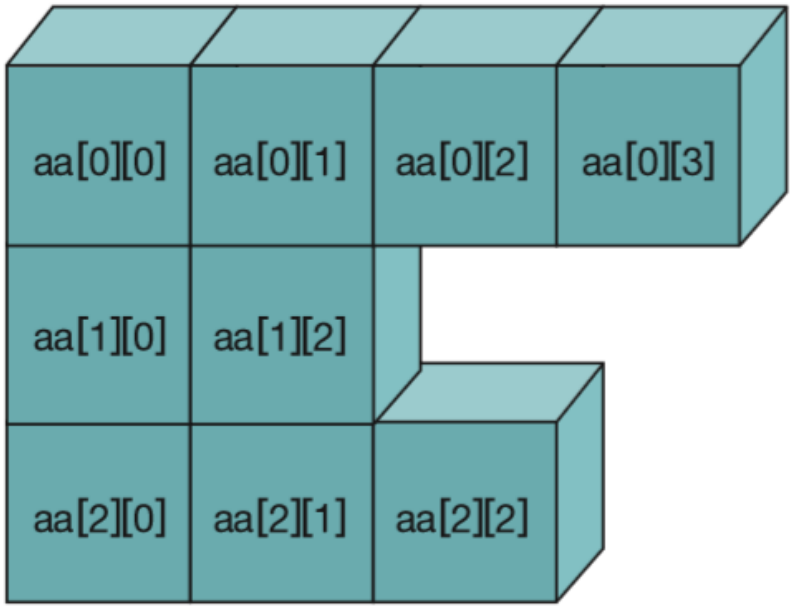



```
aa = [[1, 2, 3, 4],
      [5, 6, 7, 8],
      [9, 10, 11, 12]]
```

2D List

- **Asymmetric 2D list**

```
aa = [[1, 2, 3, 4],
      [5, 6],
      [7, 8, 9]]
```



2D List

- **Examples**

```
>>> aa = [[1, 2], [3, 4]]
>>> aa
[[1, 2], [3, 4]]
>>> aa1 = [1, 2]
>>> aa2 = [3, 4]
>>> aa = [aa1, aa2]
>>> aa
[[1, 2], [3, 4]]
```

```
>>> aa[0][0]
1
>>> aa[0][1]
2
>>> aa[1][0]
3
>>> aa[1][1]
4
```

2D List

- **Exercise**

```
1  list1 = []
2  list2 = []
3  value = 1
4  for i in range(0, 3) :
5      for k in range(0, 4) :
6          list1.append(value)
7          value += 1
8      list2.append(list1)
9      list1 = []
10
11 for i in range(0, 3) :
12     for k in range(0, 4) :
13         print("%3d" % list2[i][k], end = " ")
14     print("")
```

출력 결과

```
 1  2  3  4
 5  6  7  8
 9 10 11 12
```

7.4 튜플

Tuple

- **Basics**
  - Read-only list

```
print("%d / %d = %5.1f" % (100, 200, 0.5))
```

```
tt1 = (10, 20, 30); tt1
tt2 = 10, 20, 30; tt2
```

출력 결과

```
(10, 20, 30)
(10, 20, 30)
```

Tuple

- **Basics**
  - Single-element tuple

```
tt3 = (10); tt3
tt4 = 10; tt4
tt5 = (10,); tt5
tt6 = 10,; tt6
```

출력 결과

```
10
10
(10,)
(10,)
```

Tuple

- **Indexing**
  - `myTuple[index], myTuple[start:end+1]`

```
tt1 = (10, 20, 30, 40)
tt1[0]
tt1[0] + tt1[1] + tt1[2]
```

출력 결과

```
10
60
```

```
tt1[1:3]
tt1[1:]
tt1[:3]
```

출력 결과

```
(20, 30)
(20, 30, 40)
(10, 20, 30)
```

Tuple

- **Operations**
  - +: concatenation
  - *: repetition

```
tt2 = ('A', 'B')
tt1 + tt2
tt2 * 3
```

출력 결과

```
(10, 20, 30, 40, 'A', 'B')
('A', 'B', 'A', 'B', 'A', 'B')
```

Tuple

- **Read-only memory**
  - Indexing (O), change (X), delete of an element (X)

```
tt1.append(40)
tt1[0] = 40
del(tt1[0])
```

```
>>> tt1 = (10, 20, 30)
>>> tt1.append(40)
Traceback (most recent call last):
  File "<pyshell#35>", line 1, in <module>
    tt1.append(40)
AttributeError: 'tuple' object has no attribute 'append'
>>> tt1[0] = 40
Traceback (most recent call last):
  File "<pyshell#36>", line 1, in <module>
    tt1[0] = 40
TypeError: 'tuple' object does not support item assignment
>>> del(tt1[0])
Traceback (most recent call last):
  File "<pyshell#37>", line 1, in <module>
    del(tt1[0])
TypeError: 'tuple' object doesn't support item deletion
```

Tuple

- **Delete**

```
>>> tt1 = (10, 20, 30)
>>> del(tt1[0])
Traceback (most recent call last):
  File "<pyshell#43>", line 1, in <module>
    del(tt1[0])
TypeError: 'tuple' object doesn't support item deletion
>>> del(tt1)
>>> tt1
Traceback (most recent call last):
  File "<pyshell#45>", line 1, in <module>
    tt1
NameError: name 'tt1' is not defined
```

Tuple

- **Type casting**
  - `list(myTuple)`: tuple → list
  - `tuple(myList)`: list → tuple

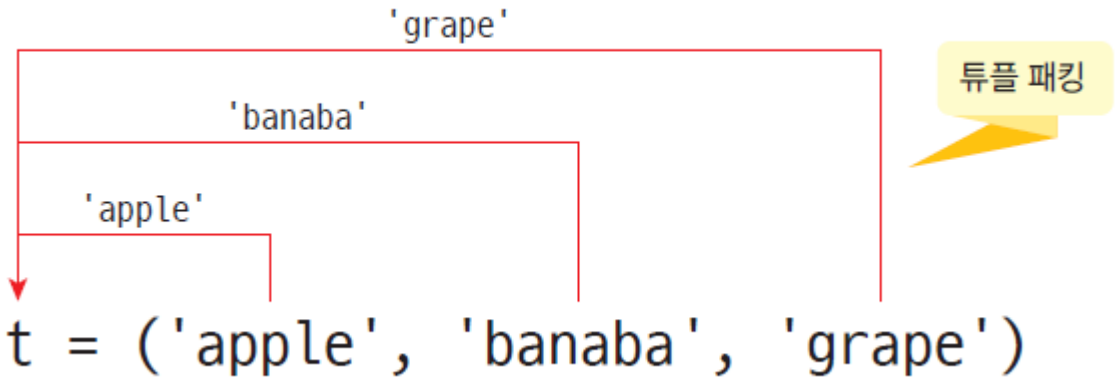
```
myTuple = (10, 20, 30)
myList = list(myTuple)
myList.append(40)
myTuple = tuple(myList)
myTuple
```

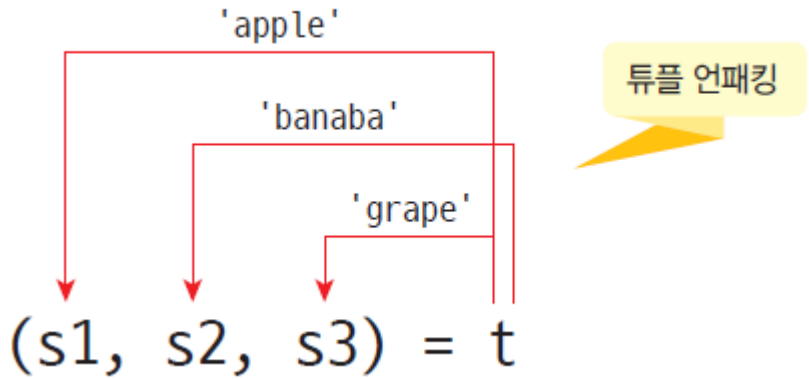
출력 결과

```
(10, 20, 30, 40)
```

Tuple

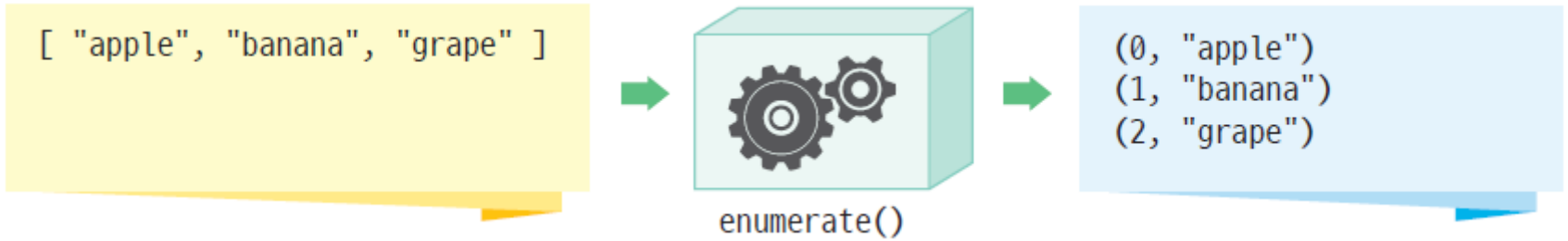
- **Unpacking**
  - Tuple에 저장된 데이터를 풀어서 개별 변수에 저장





Tuple

- enumerate()




```
fruits =["apple","banana","grape"]

for index, value in enumerate(fruits):
    print(index, value)
```

Tuple

- **Example**

```
for i in range(n_items):
    print(f"{tic[i]}: ", end="")

    for j in range(bar[i]):
        print("*", end="")

    print(f" {bar[i]:2d}개")
```

```
for i, k in enumerate(tic):
    print(f"{k}: ", end="")

    for j in range(bar[i]):
        print("*", end="")

    print(f" {bar[i]:2d}개")
```

Assignment 8

- **Transpose (전치행렬)**
  - *Step 1)* 열과 행의 개수를 입력 받음
  - *Step 2)* `random.randint(1, 10)`을 이용해 임의의 2D list 생성
  - *Step 3)* 생성된 2D list의 열과 행 인덱스를 뒤집어 전치 행렬 계산
  - *Step 4)* 원본 행렬과 전치 행렬 출력

Assignment 8

- **Transpose (전치행렬)**
  - Hints
    - `import random, random.randint(1, 10)`

```
행 개수를 입력하시오 : 2

열 개수를 입력하시오 : 2

원본 행렬
  5  3
  9 10

전치 행렬
  5  9
  3 10

행 개수를 입력하시오 : 3

열 개수를 입력하시오 : 2

원본 행렬
 10  8
  2  2
  2  7

전치 행렬
 10  2  2
  8  2  7
```