파이썬 프로그래밍

리스트의 활용

한국기술교육대학교
온라인평생교육원

▣ 리스트 내포

1. 일반적인 리스트 생성법

```
L = []
for k in range(10):
    L.append(k*k)

print L
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

- range(10) = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
- 2*2, 3*3, 4*4 …… 9*9

• 위 코딩은 리스트 내포 리터럴 방식을 활용해서 아래와 같이 변경할 수 있다.

```
L = [k * k for k in range(10)]
print L
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

- 리스트 내포 = 리스트 안에 실제 포함되어야 할 원소가 식으로 들어감
- for 의 변수가 식으로 들어간 변수와 동일해야함
- 0, 1, 2, 3…이 k에 할당될 때마다 식을 진행하여 리스트의 원소로 할당

▣ 리스트 내포

2. 리스트 내포 리터럴

```
[expression for expr1 in sequence1
            for expr2 in sequence2
            ...
            for exprN in sequenceN
            if condition]
```

•expression의 평가 결과 반드시 한 개의 원소가 나와야 한다.
  - 틀린 예: [x, y for x in seq1 for u in seq2]

•만약 두 개의 이상의 평가 결과가 나오면 튜플 등으로 감싸 주어야 한다.
  - 올바른 예: [(x, y) for x in seq1 for u in seq2]

•위 리터럴은 다음의 일반적인 for 문의 리털러과 동일

```
l = []
for expr1 in sequence1:
  for expr2 in sequence2:
    ...
      for exprtN in sequenceN:
        if condition:
          l.append(expression)
```

- k*k 또는 K % 10 등등 → expression → 식
- expr1, expr2의 값들이 expression 안에 쓰임
- if condition → if 뒤에 있는 조건 충족 시 expression 안 변수 적용
- 리스트 내포 안 각각의 for 문을 중첩된 포문 형태로 작성
- for 문 앞에 나오는 평가 결과는 반드시 한 개 원소
- 튜플로 묶어주면 하나의 원소로 취급 가능

▣ 리스트 내포

2. 리스트 내포 리터럴

```
L = [k * k for k in range(10) if k % 2]   # 홀수의 제곱만 리스트로 형성
print L
```

```
[1, 9, 25, 49, 81]
```

- k=0 → 0 % 2 = 0 → if 0 → 결과가 false로 expression 수행 X
- k=1 → 1%2 = 1 → if 결과가 true로 expression 수행
- k=2 → 2 % 2 = 0 → if 0 → 결과가 false로 expression 수행 X
- k=3 → 3%2 = 1 → if 결과가 true로 expression 수행

• 위의 리스트 내포 코드는 아래와 동일

```
L = []
for k in range(10):
    if k%2:
        L.append(k*k)
print L
```

```
[1, 9, 25, 49, 81]
```

▣ 리스트 내포

2. 리스트 내포 리터럴

• 20보다 작은 2의 배수와 3의 배수에 대해 그 두 수의 합이 7의 배수인 것들에 대해 그 두 수의 곱을 출력하는 코드

```
L = [(i, j, i*j) for i in range(2, 20, 2) for j in range(3, 20, 3) if (i + j) % 7 == 0]
print L
```

```
[(2, 12, 24), (4, 3, 12), (6, 15, 90), (8, 6, 48), (10, 18, 180), (12, 9, 108),
(16, 12, 192), (18, 3, 54)]
```

- range(2, 20, 2) = [2, 4, 6, 8, 10, 12, 14, 16, 18]
- range(3, 20, 3) = [3, 6, 9, 12, 15, 18]
- i ← 20보다 작은 2의 배수, j ← 20보다 작은 3의 배수
- 2 + 12 = 14 ← 7의 배수
- 4 + 3 = 7 ← 7의 배수
- 18 + 3 = 21 ← 7의 배수

• 두 개의 시퀀스 자료형에 대해 각각의 원소에 대한 쌍을 튜플 형태로 만들면서 리스트에 저장하는 코드

```
seq1 = 'abc'
seq2 = (1, 2, 3)
print [(x, y) for x in seq1 for y in seq2]
```

```
[('a', 1), ('a', 2), ('a', 3), ('b', 1), ('b', 2), ('b', 3), ('c', 1), ('c', 2), ('c', 3)]
```

- seq1 → 문자열, seq2 → 튜플
- 첫 번째 for문 = seq1에서 x, 두 번째 for 문 = seq2에서 y
- x가 a인 경우, b인 경우, c인 경우 y는 1,2, 3 반복 → 9쌍

▣ 리스트 내포

2. 리스트 내포 리터럴

```
words = 'The quick brown fox jumps over the lazy dog'.split()
stuff = [[w.upper(), w.lower(), len(w)] for w in words]
for i in stuff:
    print i
```

```
['THE', 'the', 3]
['QUICK', 'quick', 5]
['BROWN', 'brown', 5]
['FOX', 'fox', 3]
['JUMPS', 'jumps', 5]
['OVER', 'over', 4]
['THE', 'the', 3]
['LAZY', 'lazy', 4]
['DOG', 'dog', 3]
```

- split ()→ 공백을 기준으로 문자열 잘라 리스트 만들기
- stuff는 리스트 내포 문법 형태
- 리스트 내포 안의 원소는 3개의 원소를 가진 리스트 형태로 하나
- upper : 대문자로 변환, lower: 소문자로 변환, len: 단어의 길이
- i 자체가 리스트