중요사항 정리

리스트

리스트 이름 = [인덱스0, 인덱스1, 인덱스2, ...]
인덱스 값을 활용하여 리스트의 항목에 접근할 수 있다.
빈 리스트도 만들 수 있다.
빈 리스트에 새롭게 추가할 때 append(항목) 메소드를 사용한다. 리스트 이름.append('추가할 값')
또는 리스트 이름 = 리스트 이름 + ['추가할 값'] 이런식으로도 추가 가능
리스트는 연산도 가능하다. +연산은 합치고 * 연산은 반복되어 저장된다.
in 연산자는 리스트에 어떤 값이 포함되는 지를 확인할 수 있다.
리스트 안에 리스트를 인덱스 값으로 가질 수 있다.
인덱스를 음수로 줄 수 있다. 맨끝에서 부터 -1로 시작해서 1씩 작아진다.
슬라이싱 -> 리스트 이름[시작값 : 끝값 : step값]
리스트를 복제한 새로운 리스트를 만들 때는 blist = list(alist) or blist = alist[:]
리스트 함축 기능은 번거롭게 반복문을 여러줄로 사용하는 것을 한줄로 줄일 수 있다.

리스트에 사용 가능한 함수

len(): 길이를 반환
max(): 리스트 항목 중 최댓값
min(): 리스트 항목 중 최솟값
list(range()): range()함수가 생성한 값을 리스트에 넣는다.
any(): 리스트에 0이 아닌 원소가 하나라도 있으면 True를 반환

리스트의 메소드

index(x): x를 이용하여 위치를 찾는 기능을 한다.
append(x): x를 리스트의 끝에 추가.
count(x): 리스트내에서 원소 x의 개수.
extend([x1, x2]): [x1, x2]리스트를 기존 리스트에 삽입.
insert(index, x): 원하는 index 위치에 x를 추가.
remove(x): x를 리스트에서 제거.
pop(index): index 위치의 원소를 삭제 후 반환.
sort(): 오름차순으로 값을 정렬.
reverse(): 리스트의 원소들을 역순으로 만든다.

튜플

튜플 생성은 ()안에 항목들을 넣으면 된다.
튜플은 한번 지정되면 변결될 수 없는 불변속성 자료형이다.

자기점검

튜플의 개념을 새롭게 알 수 있었다. 그리고 리스트의 메소드들을 사용하는데 약간의 어색함을 느꼈다.

심화문제 풀기

7-1

In [18]:
```
num_list = [100, 200, 300, 400, 500, 600, 700, 800]
high = 6
low = 3
num_list[high]
```

Out[18]: 700

In [3]:
```
num_list[high - 2]
```

Out[3]: 500

In [4]:
```
num_list[high - low]
```

Out[4]: 400

In [5]:
```
num_list[low - high]
```

Out[5]: 600

In [6]:
```
num_list[-1]
```

Out[6]: 800

In [7]:
```
num_list[-low]
```

Out[7]: 600

In [8]:
```
num_list[2 * 3]
```

Out[8]: 700

In [9]:
```
num_list[2] * 3
```

Out[9]: 900

In [10]:
```
num_list[5 % 4]
```

Out[10]: 200

In [11]:
```
len(num_list)
```

Out[11]:
```
8
```

In [12]:
```
min(num_list)
```

Out[12]:
```
100
```

In [13]:
```
max(num_list)
```

Out[13]:
```
800
```

In [14]:
```
num_list[:3]
```

Out[14]:
```
[100, 200, 300]
```

In [15]:
```
num_list[1:5]
```

Out[15]:
```
[200, 300, 400, 500]
```

In [16]:
```
num_list[-1:-5:-1]
```

Out[16]:
```
[800, 700, 600, 500]
```

In [17]:
```
num_list[-5:-1:1]
```

Out[17]:
```
[400, 500, 600, 700]
```

7-2

In [22]:
```
list1 = [3, 5, 7]
list2 = [2, 3, 4, 5, 6]
for i in list1:
    for x in list2:
        print('{} * {} = {}'.format(i, x, i*x))
```

```
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
```

7-3

In [24]:
```
list1 = ['I like ', 'I love ']
```

```
list2 = ['pancake.', 'kiwi juice.', 'espresso.']
for i in list1:
    for x in list2:
        print(i + x)
```

```
I like pancake.
I like kiwi juice.
I like espresso.
I love pancake.
I love kiwi juice.
I love espresso.
```

7-4

In [25]:
```
t = (10, 20, 30, 40)
t.append(50)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~WAppDataWLocalWTemp/ipykernel_1768/1722827794.py in <module>
      1 t = (10, 20, 30, 40)
----> 2 t.append(50)

AttributeError: 'tuple' object has no attribute 'append'
```

t는 튜플이므로 불가변속성이어서 append 메소드를 사용할 수 없다. 리스트에만 사용 가능하다.

In [26]:
```
t = (10, 20, 30, 40)
t.remove(40)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~WAppDataWLocalWTemp/ipykernel_1768/4054911080.py in <module>
      1 t = (10, 20, 30, 40)
----> 2 t.remove(40)

AttributeError: 'tuple' object has no attribute 'remove'
```

t는 튜플이므로 불가변속성이어서 remove 메소드를 사용할 수 없다. 리스트에만 사용 가능하다.

In [28]:
```
t = (10, 20, 30, 40)
t[0] = 0
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~WAppDataWLocalWTemp/ipykernel_1768/848212476.py in <module>
      1 t = (10, 20, 30, 40)
----> 2 t[0] = 0

TypeError: 'tuple' object does not support item assignment
```

t는 튜플이므로 불가변속성이이서 다른 인덱스 값으로 바꿀수 없다.

7-5

In [45]:
```
seats = []
alp = ['A', 'B', 'C']
for n in range(1,3):
    for i in alp:
        seats.append('{}{}'.format(i,n))
```

```
    n+=1
seats
```

Out[45]:
```
['A1', 'B1', 'C1', 'A2', 'B2', 'C2']
```

In [46]:
```
seats = []
alp = ['A', 'B', 'C']
num = ['1', '2']
for n in num:
    for i in alp:
        seats.append('{}{}'.format(i,n))
seats
```

Out[46]:
```
['A1', 'B1', 'C1', 'A2', 'B2', 'C2']
```

7-6

In [8]:
```
str = input('문자열을 입력하세요: ')
for i in range(len(str)+1):
    print(str[:i])

for i in range(len(str)+1):
    print(str[:len(str)-i-1])
```

```
p
py
pyt
pyth
pytho
python
pytho
pyth
pyt
py
p

pytho
```

7-7

In [9]:
```
fruit_list = ['banana', 'orange', 'kiwi', 'apple', 'melon']
maximum = len(max(fruit_list))
print('가장 길이가 긴 문자열 : ', end = " ")
for i in ['banana', 'orange', 'kiwi', 'apple', 'melon'] :
    if len(i) == maximum :
        print(i, end = ' ')
        fruit_list.remove(i)
print('')
print('fruit_list =', fruit_list)
```

```
가장 길이가 긴 문자열 :  banana orange
fruit_list = ['kiwi', 'apple', 'melon']
```

In [11]:
```
fruit_list = ['banana', 'orange', 'kiwi', 'apple', 'melon']

for i in range (0, len(fruit_list)):
    print('{} : 문자열의 길이 {}'.format(fruit_list[i], len(fruit_list[i])))
```

```
banana : 문자열의 길이 6
orange : 문자열의 길이 6
kiwi : 문자열의 길이 4
apple : 문자열의 길이 5
melon : 문자열의 길이 5
```

7-8

In [12]:

```
A = int(input("n을 입력하시오 : "))
D = [x for x in range(1, A * A + 1)]
S = ""

for i in range(0, A):
    print("")
    if i % 2 == 0:
        S = D[(i*A) : (i*A) + A]
        for t in range (0, len(S)):
            print('{0:3d}'.format(S[t]), end = "")
    else:
        S = D[(i*A) + (A-1) : (i * A) - 1 : -1]
        for t in range (0, len(S)):
            print('{0:3d}'.format(S[t]), end = "")
```

```
  1  2  3  4  5
 10  9  8  7  6
 11 12 13 14 15
 20 19 18 17 16
 21 22 23 24 25
```

7-9

In [17]:

```
A = input('A = ')
B = input('B = ')

def overlap(A, B):

    for i in range(0, len(A)):
        last = A[i : len(A)]
        first = B[: len(A) - i]
        if last == first:
            C = A[0 : len(A)] + B[len(A) - i : len(B)]
            return C

        elif i == len(A)-1:
            C = A + B
            return C

print('C =', overlap(A, B))
```

```
C = commummy
```