Python – 2

2.1 태성이는 자연수로 이루어진 숫자들을 정렬합니다. 하지만 태성이는 숫자를 뒤집어 읽습니다. 예를 들어 123이라면 321로 읽습니다. 태성이와 같은 방식으로 자연수 리스트를 오름차순으로 정렬하여 반환하는 함수를 작성하세요.

(단 역순값이 동일하면, 길이는 상관하지 않습니다.)

```
1    def solution2_1(numbers):
2
3        return numbers
```

입력 예시	출력 예시
[1, 12, 1, 32, 51]	[1, 1, 51, 12, 32]
[5, 61, 27, 132, 119, 2]	[2, 5, 61, 27, 132, 119]
range(50)	[0, 1, 10, 2, 20, 3, 30, 4, 40, 5, 6, 7, 8, 9, 11, 21, 31, 41, 12, 22, 32, 42, 13, 23, 33, 43, 14, 24, 34, 44, 15, 25, 35, 45, 16, 26, 36, 46, 17, 27, 37, 47, 18, 28, 38, 48, 19, 29, 39, 49]

2.2 아래와 같은 소스코드 존재합니다.

```
3
4   def solution2_2(the_list=[]):
5       for value in range(3):
6               the_list.append(value)
7       return the_list
8
9   if __name__ == '__main__':
10        print(solution2_2())
11        print(solution2_2([1, 4]))
12        print(solution2_2())
13        print(solution2_2([0, 0, 0]))
```

해당 함수는 인자를 주지 않으면 [0, 1, 2]와 같은 값의 리스트를 반환하고, 리스트를 인자로 넘기면 [0, 1, 2]를 append하여 반환하는 것을 목적으로 합니다.

출력을 예측해보고 왜 이유를 작성하세요. 실제로 출력을 해보고 예측한 결과와 동일한지 확인해보세요.

예측이 틀렸다면, 해당 함수가 목적에 맞게 작동하도록 수정하세요. 이 때, 함수 내부외에는 수정할 수 없습니다.

답변 목록	답변
예상 출력	. . .
실제 출력	. . .
수정 코드	. . .

2.3 [0, 100) 범위의 요소로 이루어진 리스트 numbers 입력으로 들어옵니다. 해당 리스트의 요소 중 [0, N) 범위의 요소를 key로하고 요소의 개수를 value로 하는 dictionary를 반환하시오. (단, 0 < N < 100)

```
4    def solution2_3(numbers, N):
5
6        return numbers
```

입력 예시	출력 예시
[1, 2, 1, 2, 3], 5	{0: 0, 1: 2, 2: 2, 3: 1, 4: 0} # 출력예시 수정
list(range(10))+list(range(20))+list(range(30)), 17	{0: 3, 1: 3, 2: 3, 3: 3, 4: 3, 5: 3, 6: 3, 7: 3, 8: 3, 9: 3, 10: 2, 11: 2, 12: 2, 13: 2, 14: 2, 15: 2, 16: 2}

2.4 두 개의 문자열이 입력으로 들어옵니다. 하나의 문자열을 A, 또 하나의 문자열을 B라고 할 때 A에는 존재하지만 B에는 존재하지 않는 요소의 종류를 리스트로 묶어 반환하는 함수를 작성하시오.

입력 예시	출력 예시
Abbbcdegh, 가나cbeqjjl;	['h', 'A', 'd', 'g']
가각궤난다람박, 불궤난다라마바사	['람', '박', '각', '가']
!&@()@&aoweij, aoweimmmmmmn	['(', '@', ')', 'j', '&', '!']

2.5 두 개의 리스트가 입력으로 주어진다. 리스트 A의 요소는 불변 객체로 구성되어있으며 리스트 B는 임의의 객체들로 구성되어있다.

동일한 인덱스의 리스트 A, B의 요소를 각각 key, value로 하는 딕셔너리를 반환하는 함수를 작성하시오. 이 때, 딕셔너리의 길이는 두 리스트 중 짧은 리스트와 동일합니다.

입력 예시	출력 예시
[1, 3, 2, 'abcd', 38, (1, 3, 5)], [[1, 2, 3], 3, (1,), 'AB', 'BC', 'DEFG', 14, 123]	{1: [1, 2, 3], 3: 3, 2: (1,), 'abcd': 'AB', 38: 'BC', (1, 3, 5): 'DEFG'}
[1, 2, 3, 4, 5], [1, 3]	{1: 1, 2: 3}