Chapter_14~16 python 문제 풀기 (답안) G (1)

1. 주어진 정수 x와 자연수 n을 이용해, x부터 시작해 x씩 증가하는 숫자를 n개 지니는 리스트를 출력해주세요

x = 2, n =5

🔑 정답 :

리스트 컴프리헨션을 활용한 답안

```
[i * x for i in range(1, n + 1)]
```

리스트 컴프리헨션을 활용하지 않은 답안

```
for i in range(1, n + 1):
    result.append(i * x)
```

`result`를 생성한 후, for문을 이용하여 1부터 n까지의 숫자를 x와 곱한 값을 리스트에 추가합니다. 이렇게 하면 x부터 시작해 x씩 증가하는 숫자를 n개 지니는 리스트를 생성할 수 있습니

---

2. 전화번호가 문자열 phone_number로 주어졌을 때, 전화번호의 뒷 4자리를 *으로 가린 문자열이 출력되도록 코드를 작성해주세요

phone_number = "01012347890"
출력 예시 : "0101234****"

🔑 정답 :

문자열 슬라이싱과 문자열 연결 방식을 이용한 답안

```
new_number = phone_number[:-4] + "****"
```

전화번호의 뒷 4자리를 제외한 부분을 슬라이싱하여 new_number에 할당하고, 그 뒤에 "****" 문자열을 추가하여 전화번호의 뒷 4자리를 가립니다. 그리고 마지막으로 가려진 전화번호를 출력합니다.

문자열의 길이를 반환하는 `len()` 함수와 문자열을 특정 문자로 채우는 `str.ljust()` 함수를 이용한 답안

```
new _number = phone_number[:-4].ljust(len(phone_number), '*')
```

전화번호의 뒷 4자리를 제외한 부분을 슬라이싱하여 new_number에 할당합니다. 그리고 `str.ljust()` 함수를 이용하여 new_number의 길이를 전화번호의 길이와 같게 만들면서 빈 공간을 *''로 채웁니다. 결국 전화번호의 뒷 4자리가* ''로 가려져서 출력됩니다.

---

3. 숫자를 건넬 때 일부 자릿수를 영단어로 바꾼 카드를 건네주면 프로도는 원래 숫자를 찾는 게임입니다.

다음은 숫자의 일부 자릿수를 영단어로 바꾸는 예시입니다.

1478 → "one4seveneight"

234567 → "23four5six7"

10203 → "1zerotwozero3"

이렇게 숫자의 일부 자릿수가 영단어로 바뀌어졌거나,

혹은 바뀌지 않고 그대로인 문자열 s가 주어졌을때 s가 의미하는 원래 숫자를 출력하도록 코드를 작성해주세요

s = "77three4one

참고로 각 숫자에 대응되는 영단어는 다음 표와 같습니다.

숫자 영단어

0 zero

1 one

2 two

3 three

4 four

5 five

6 six

7 seven

8 eight

9 nine

바꾸려는 것 : onetwothree46

차례대로 하나씩 읽어오면 되겠네?

one = 1

two = 2

check = ''

0. 읽어오는 값을 check 변수에 저장.
   왜? 저장을 해야, 컴퓨터가 인식해서 활용할 수 있기 때문에!
1. **check 변수에 바꿀 수 있는 문자열이 있을 때, 숫자로 변환.**
2. 문자열이 변환되면 (one → 1) check 변수에 있던 문자열은 비워진다.
3. 하나씩 읽어오는 행위가 반복 → 반복문 사용

🔑 정답 :

딕셔너리 자료형과 replace 내장 함수를 이용한 답안

```
num_dict = {
    "zero": "0",
    "one": "1",
    "two": "2",
    "three": "3",
    "four": "4",
    "five": "5",
    "six": "6",
    "seven": "7",
    "eight": "8",
    "nine": "9"
}
s = "77three4one"
for key in num_dict.keys():
    s = s.replace(key, num_dict[key])

print(s)
```

숫자와 해당 영단어를 매핑한 딕셔너리 `num_dict`를 생성합니다. 그리고 `num_dict`의 모든 키에 대해, 주어진 문자열 `s`에서 해당 키를 찾아서 그에 해당하는 딕셔너리 값으로 바꾸는 과정을 반복합니다. 이렇게 하면 문자열 `s`에서 숫자를 나타내는 모든 영단어가 실제 숫자로 바뀝니다.

이 코드를 실행하면 "77three4one"이 "77341"로 출력됩니다. 이는 "77three4one"이 의미하는 원래 숫자입니다.

---

4. 행렬의 덧셈은 행과 열의 크기가 같은 두 행렬의 같은 행, 같은 열의 값을 서로 더한 결과가 됩니다. 2개의 행렬 arr1과 arr2가 주어졌을 때, 행렬 덧셈의 결과를 출력하는 코드를 작성해주세요(result와 같은 결과가 출력되도록 작성)

arr1 = [[1,2],[2,3]]

arr2 = [[3,4],[5,6]]

result = [[4,6],[7,9]]

🔑 정답 :

컴프리헨션을 사용하지 않고 행렬을 더하는 방법

```
arr1 = [[1,2],[2,3]]
arr2 = [[3,4],[5,6]]

result = []
for i in range(len(arr1)):
    row = []
    for j in range(len(arr1[0])):
        row.append(arr1[i][j] + arr2[i][j])
    result.append(row)

print(result)
```

두 행렬이 담겨있는 변수를 이용해 각 행렬의 각 행과 열에 대해 루프를 돌면서 각 요소를 더합니다. 더한 결과는 `row`라는 임시 리스트에 추가되고, 각 행의 계산이 끝나면 이 `row` 리스트가 결과 리스트 `result`에 추가됩니다. 이렇게 해서 두 행렬의 덧셈 결과를 계산하고 반환합니다.

이 코드를 실행하면 `[[4,6],[7,9]]`가 출력됩니다. 이는 주어진 두 행렬의 덧셈 결과입니다.

컴프리헨션을 사용해 행렬을 더하는 방법

```
arr1 = [[1,2],[2,3]]
arr2 = [[3,4],[5,6]]

result = [[c1 + c2 for c1, c2 in zip(r1, r2)] for r1, r2 in zip(arr1, arr2)]

print(result)
```

두 행렬이 담겨있는 변수를 이용해 각 행렬의 각 행을 zip 함수로 묶은 뒤, 각 행의 각 요소를 더하는 리스트 컴프리헨션을 실행합니다. 이렇게 해서 두 행렬의 덧셈 결과를 계산하고 반환합니다.

이 코드를 실행하면 `[[4,6],[7,9]]`가 출력됩니다. 이는 주어진 두 행렬의 덧셈 결과입니다.

---

5. 양의 정수 x가 하샤드 수이려면 x의 자릿수의 합으로 x가 나누어져야 합니다.

예를 들어 18의 자릿수 합은 1+8=9이고, 18은 9로 나누어 떨어지므로 18은 하샤드 수입니다.

주어진 x를 이용해 x가 하샤드 수인지 아닌지 검사하는 코드를 작성해주세요

x = 17

입출력 예시 x 결과값

10 true

12 true

11 false

13 false

🔑 정답 :

```
x = 17
num_list = list(map(int, str(x)))
result = x % sum(num_list) == 0

print(result)
```

주어진 정수 `x`를 문자열로 변환한 뒤, 각 자릿수를 정수로 변환하여 리스트로 만듭니다. 그리고 `sum(num_list)`로 모든 자릿수의 합을 구하고, `x`를 그 값으로 나누어 나머지가 0인지 검사합니다. 나머지가 0이면 `x`는 하샤드 수이므로 `True`를 반환하고, 그렇지 않으면 `False`를 반환합니다.

여기서 `x = 17`로 설정했으므로, 이 코드를 실행하면 `False`가 출력됩니다. 왜냐하면 17의 자릿수의 합은 1+7=8이고, 17은 8로 나누어 떨어지지 않기 때문입니다.

---

6. String형 배열 seoul의 element중 "Kim"의 인덱스 x를 찾아, "김서방은 x에 있다"는 String을 출력하는 코드를 작성해주세요.

seoul에 "Kim"은 오직 한 번만 나타나며 잘못된 값이 입력되는 경우는 없습니다.

seoul = ["Jane", "Kim"]

결과 "김서방은 1에 있다"

🔑 정답 :

```
seoul = ["Jane", "Kim"]

print(f"김서방은 {seoul.index('Kim')}에 있다")
```

"Kim"의 위치를 찾아 "김서방은 {index}에 있다" 형태의 문자열을 반환합니다. f-string을 사용하면 format 함수를 사용하는 것보다 코드가 간결해지고 가독성이 좋아집니다.

---

7. 0부터 9까지의 숫자 중 일부가 들어있는 정수 배열 numbers가 주어집니다.

numbers에서 찾을 수 없는 0부터 9까지의 숫자를 모두 찾아 더한 수를 출력해주세요

numbers = [5,8,1,0,6,9]

🔑 정답 :

```
numbers = [5,8,1,0,6,9]

full_set = set(range(10))  # 0부터 9까지의 숫자를 포함하는 집합을 생성
numbers_set = set(numbers)  # 주어진 배열을 집합으로 변환
missing_numbers = full_set - numbers_set  # 두 집합의 차집합을 구함
result = sum(missing_numbers)  # 차집합의 모든 원소를 더함

print(result)
```

`set` 자료형의 차집합 연산을 이용하여 `full_set`에는 있지만 `numbers_set`에는 없는 숫자를 찾아내고, 이를 모두 더하는 방식으로 작동합니다. `set` 자료형은 중복을 허용하지 않고 순서가 없는 자료형으로, 집합 연산에 유용하게 사용할 수 있습니다.