파이썬 스타일 코드1 - 연습해보기

일반문제

1. Css Selector 수정

Css Selector는 웹 페이지에서 특정 요소를 선택하기 위해 해당 요소까지 찾아갈 수 있도록 해주는 주소와 같은 것이다. 대부분의 웹브라우저에서 는 해당 요소에 대한 css selector 값을 쉽게 얻어올 수 있다 (F12 > Select an Element). 다음과 같은 selector가 있다고 한다.

today_main_news > div.hdline_news > ul > li:nth-child(1)

이러한 selector를 웹크롤링에서 사용하기 위해서는 :nth-child라는 부분을 제거하는 작업이 필요한데, 이를 자동화 해보자.

Q: 해당 Selector를 문자열로 표시하고, split과 join 함수를 활용하여 다음 예시와 같은 selector를 출력하시오.

출력결과 예시

selector = "#today_main_news > div.hdline_news > ul > li:nth-child(1)"

CODE

'#today_main_news > div.hdline_news > ul > li'

HINT

1.특정 구분자(seperator)를 통해 구분된 리스트를 만든다.

2.구분된 리스트에서 해당 부분을 선택하고 1과는 다른 특정 구분자로 나눠준다.

3.2의 리스트에서 필요한 부분만 선택하여 기존 리스트에 할당한다.

4.구분자를 기준으로 리스트를 문자열로 합쳐준다.

In [20]:
```
selector = "#today_main_news > div.hdline_news > ul > li:nth-child(1)"

selector_list = selector.split(">") #.split을 사용하여 ">"를 기준으로 문자열을 나눈다
selector_list[-1] = selector_list[-1].split(":")[0]#필요한 문자열을 골라 split을 사용


" > ".join(selector_list) #나머지 필요한 부분을 조합하여 join을 사용해 문자열을 합쳐준
```

Out[20]: '#today_main_news > div.hdline_news > ul > li'

2. list comprehension으로 만드는 구구단

PR5 문제 3번에서 만들었던 구구단 계산기를 list comprehension으로 구현해보고자 한다.

Q: list comprehension을 사용하여 구구단을 연산하는 함수 gugu_com을 작성하고 구구단 7단을 출력하시오.

출력 결과 예시

gugu_com(x=2)

2 x 1 = 2

2 x 2 = 4

2 x 3 = 6

2 x 4 = 8

2 x 5 = 10

2 x 6 = 12

2 x 7 = 14

2 x 8 = 16

2 x 9 = 18

In [22]:
```
def gugu_com(x=2):
    [print(f"{x} x {i} = {x*i}") for i in range(1, 10)]

gugu_com(2) # def를 사용해 gugu_com함수를 만든다. range를 이용하여 범위를 지정해주어 1
```

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
```

두 주사위의 곱

두 주사위의 곱은 다음과 같은 결과를 가진다.

Q: list comprehension을 사용하여, 힌트를 제외하고는 한줄의 코드로 해당 결과를 가지는 이차원 리스트를 만드시오.

출력결과 예시

CODE

[[1, 2, 3, 4, 5, 6],

[2, 4, 6, 8, 10, 12],

[3, 6, 9, 12, 15, 18],

[4, 8, 12, 16, 20, 24],

[5, 10, 15, 20, 25, 30],

[6, 12, 18, 24, 30, 36]]

HINT

한개의 주사위는 다음과 같이 표현할 수 있습니다.

die = [i for i in range(1,7)]

In [25]:
```
die = [i for i in range(1,7)] #range로 1부터7까지 지정하면 1부터6까지인 주사위 수를 표

[[j*i for i in die] for j in die] #지정한i와 die가 곱해지며 값이 출력됨
```

Out[25]:
```
[[1, 2, 3, 4, 5, 6],
 [2, 4, 6, 8, 10, 12],
 [3, 6, 9, 12, 15, 18],
 [4, 8, 12, 16, 20, 24],
 [5, 10, 15, 20, 25, 30],
 [6, 12, 18, 24, 30, 36]]
```

4. 두 주사위의 합

간단한 테이블 형태의 데이터를 2차원 리스트로 표현해보자. 2개의 주사위를 굴리면 다음 표와 같이 36가지의 결과가 나온다.

Q: 이것을 6 x 6 크기의 2차원 리스트로 생성하고, 인덱싱을 통해 2 + 6의 값을 2가지 방법으로 나타내시오. (2차원 리스트 생성시 방법의 제한은 없습니다.)

In [27]:
```
dice_sum = [[2, 3, 4, 5, 6, 7],
            [4, 5, 6, 7, 8, 9],
```

```
            [5, 6, 7, 8, 9, 10],
            [6, 7, 8, 9, 10, 11],
            [7, 8, 9, 10, 11, 12]]

print(dice_sum[1][5]) #리스트의 인덱싱을 지정하여 1번째,5번째 수인 2와6의 합을 구함
print(dice_sum[5][1])
```

```
8
8
```

In [10]:

```
die = [i for i in range(1,7)]

dice_sum = [[j+i for i in die] for j in die]

print(dice_sum[1][5])
print(dice_sum[5][1])
```

```
8
8
```

도전문제

표절 검사 프로그램

강의노트 07 자료구조 collections 설명 참고

아주대학교 글로벌 경영학과의 한 교수님은 과제의 표절 검사를 쉽게 하기 위해 Python을 통한 간단한 표절 검사 프로그램을 작성해보고자 한다.

현재 구상 중인 프로그램은 복잡한 알고리즘을 필요로하지 않고, 간단하게 단어 빈도를 기반으로 하여, 그 유사도를 측정하고자한다.

In [11]:

```
from collections import defaultdict, Counter

text = """Python is a very simple programming language so even if you are new to progr

text2 = """C is a very difficult programming language so even if you are good at progr

text3 = """R Programming is good at statistical analysis. you can learn easily"""
```

문제1

Q: defaultdict를 활용하여 text를 입력받으면 단어별 빈도를 측정하여 반환하는 함수 word_counter를 만드시오.

HINT

1.collections 모듈의 defaultdict는 단순한 dict와 다르게, 인덱싱에서 key 값이 없으면 오류가 아닌 0을 기본 값으로 가지게 한다.

word_dict = dict() word_dict["key"]

KeyError

t["key"]

0

word_dict["key"] += 1 word_dict["key"]

1

1.유사도 측정을 위해 문장을 단어별로 분할해야하며, 편의를 위해 모두 소문자로 바꿔준다.

- split
- lower

In [30]:
```
def word_counter(text): #글자 빈도 추출하기
    word_count = defaultdict(lambda: 0)
    for word in text.lower().split(): # lower를 이용해 소문자로,split를 이용해 문자를
        word_count[word] += 1

    return word_count
```

In [14]:
```
word_counter(text)
```

Out[14]:
```
defaultdict(<function __main__.word_counter.<locals>.<lambda>()>,
            {'python': 2,
             'is': 1,
             'a': 1,
             'very': 1,
             'simple': 1,
             'programming': 1,
             'language': 1,
             'so': 1,
             'even': 1,
             'if': 1,
             'you': 2,
             'are': 1,
             'new': 1,
             'to': 1,
             'programming,': 1,
             'can': 1,
             'learn': 1,
             'without': 1,
             'facing': 1,
             'any': 1,
             'issues.': 1})
```

문제2

Q: 도전문제 1의 word_counter 활용하여 text와 text2의 유사도와 text와 text3의 유사도를 구하시오.

HINT

1.collections 모듈의 Counter는 dict의 형태이지만 Counter들 간의 덧셈, 뺄셈 연산이 가능하며 defaultdict를 Counter로 변환할 수 있다.

Counter({"a": 1, "b": 2, "c": 3}) - Counter({"a": 1, "b": 1, "c": 1})

Counter({'b': 1, 'c': 2})

1.dictionary 형태의 모든 자료구조는 .values() 를 통해 value 값만 추출할 수 있다.

In [31]:
```
sum(Counter({"a": 1, "b": 2, "c": 3}).values()) # 전체 단어수 합

## 6
```

Out[31]: 6

1.Counter(A)가 Counter(B)와 얼마나 유사한지는 다음과 같은 공식을 따른다고 한다.(시그마는 해당 Counter dict 안의 value 값을 모두 합하라는 의미)

In [17]:
```
def text_similarity(text_count_1, text_count_2): #text1과text2의 유사도 구하기
    text1_count = Counter(text_count_1)
    text2_count = Counter(text_count_2)

    word_total = sum(text1_count.values())
    word_diff = sum((text1_count - text2_count).values())

    return (1 - word_diff / word_total) * 100
```

In [33]:
```
text_similarity(word_counter(text), word_counter(text2)) #text와 text2유사도 계산
```

Out[33]: 73.91304347826086

In [34]:
```
text_similarity(word_counter(text), word_counter(text3))# text와text3유사도 계산
```

Out[34]: 21.739130434782606