정규표현식

- 파이썬기초 -

문자열 연습 1

index : 문자열에 있는 문자 가져오기

temp_string = "파이썬 문자열 연습"

temp_string[0]

'파'

temp_string[9]

'습'

temp_string[-1]

'습'

slicing : 문자열에서 범위로 문자열 가져오기

시작 index에서 끝 index 전까지 문자열 리턴

temp_string = "파이썬 문자열 연습"

temp_string[2:6]

'썬 문자'

소스코드

99-1 정규표현식.ipynb

문자열 연습 2

split : 문자열 분리하기

```
coffee = "에스프레소,아메리카노,카페라테,카푸치노"
coffee.split(',')
```

```
"에스프레소 아메리카노 카페라테 카푸치노".split(' ')
```

```
"  에스프레소  \n\n  아메리카노  \n  카페라테   카푸치노
\n\n".split()
```

```
"에스프레소 아메리카노 카페라테 카푸치노".split(maxsplit=2)
```

```
# 국가 번호가 포함된 전화번호
phone_number = "+82-01-2345-6789"

# 국가 번호와 나머지 번호 분리
split_num = phone_number.split("-", 1)
```

소스코드
99-1 정규표현식.ipynb

문자열 연습 3

strip : 필요 없는 문자열 삭제하기

```
"aaaaPythonaaa".strip('a')

test_str_multi = "##***!!!##....  Python is powerful.!...  %%!#..   "
test_str_multi.strip('*.#! %')

"\n  Python  \n\n".strip()
```

```
str_lr = "000Python is easy to learn.000"
print(str_lr.strip('0'))
print(str_lr.lstrip('0'))
print(str_lr.rstrip('0'))
```

연습 :

coffee_menu = " 에스프레소, 아메리카노, 카페라테 , 카푸치노 "

=> `['에스프레소', '아메리카노', '카페라테', '카푸치노']`

소스코드

99-1 정규표현식.ipynb

문자열 연습 4

join : 문자열 연결하기

```
address_list = ["서울시","서초구","방배동","432-7"]
a = " "
a.join(address_list)
"*^-^*".join(address_list)
```

find : 문자열 찾기

매칭된 문자열이 시작되는 인덱스 반환

```
str_f = "Regular expression."
str_f.find("Regular")
str_f.find("expression")
str_f.find("x")
str_f.find("easy")
```

소스코드
99-1 정규표현식.ipynb

문자열 연습 5

find : 문자열 찾기

```
sentence = "Life is venture or nothing"

print(sentence.find("Life", 10, 30))
print(sentence.find("nothing", 15))
```

count : 일치하는 문자열 횟수

```
string = "Python is awesome, isn't it?"
substring = "is"

count = string.count(substring)
```

소스코드
99-1 정규표현식.ipynb

문자열 연습 6

startswith : 시작하는 문자열

```
text = "Python is easy to learn."

text.startswith('is easy')
text.startswith('Python is ')
text.startswith('Python is easy to learn.')
```

endswith : 끝나는 문자열

```
text = "Python is easy to learn."

text.endswith('to learn')
text.endswith('to learn.')
text.endswith('Python is easy to learn.')
```

소스코드
99-1 정규표현식.ipynb

문자열 연습 7

replace : 문자열 치환

```
song = 'cold, cold heart'
song.replace('cold', 'hurt')

song = 'Let it be, let it be, let it be, let it be'
song.replace('let', "don't let", 2)
```

isdigit()

isupper()

islower() …

but

8. Life is ventu
or nothing 인생은
모험이거나 아무것도
아니거나.

소스코드

99-1 정규표현식.ipynb

정규표현식

일정한 규칙(패턴)을 가진 문자열을 추출, 변경시 사용

html 소스에서 링크주소만 가져오는 패턴예제

https://regexr.com/4c44n

이미지출처

https://xkcd.com/208/

정규표현식 함수들

import re

match() - 문자열 처음부터 매치여부 조사. 객체리턴
search() - 문자열 전체를 조사. 처음 검색된 최초 문자열 객체리턴
findall() - 매치되는 모든 문자열 리스트로 리턴
finditer() - 매치되는 모든 문자열의 반복가능한 객체로 리턴

소스코드
99-1 정규표현식.ipynb

정규표현식 리턴 객체의 메서드

group() - 매치된 문자열의 리턴

group(0) 매치된 전체 문자열
group(1) 첫 번째 그룹에 해당되는 문자열
group(2) 두 번째 그룹에 해당되는 문자열
group(n) n 번째 그룹에 해당되는 문자열

start() - 매치된 문자열의 시작위치 리턴

end() - 매치된 문자열의 끝 위치 리턴

span() - 매치된 문자열의 (시작, 끝) 에 해당하는 튜플 리턴

소스코드
99-1 정규표현식.ipynb

re.match()

문자열 처음부터 매치여부 조사. 객체리턴

```
>>> import re

>>> text = “I like orange! I love orange!”
>>> result = re.match(“orange”, text)
>>> print(result)
None
```

```
>>> import re

>>> text = “orange! I love orange!”
>>> result = re.match(“orange”, text)
>>> print(result)
<re.Match object; span=(0, 6), match=’orange’>
>>> print(result.group())
orange
>>> print(result.start())
0
>>> print(result.end())
6
>>> print(result.span())
(0, 6)
```

소스코드
99-1 정규표현식.ipynb

re.search()

문자열 전체를 조사. 처음 검색된 최초 문자열 객체리턴

```
>>> import re

>>> text = “I like orange! I love orange!”
>>> result = re.search(“orange”, text)
>>> print(result)
<re.Match object; span=(7, 13), match='orange'>
>>> print(result.group())
orange
>>> print(result.start())
7
>>> print(result.end())
13
>>> print(result.span())
(7, 13)
```

소스코드
99-1 정규표현식.ipynb

re.findall()

매치되는 모든 문자열 리스트로 리턴

```
>>> import re

>>> text = “I like orange! I love orange!”
>>> result = re.findall(“orange”, text)
>>> print(result)
[‘orange’,’orange’]
```

소스코드
99-1 정규표현식.ipynb

re.finditer()

매치되는 모든 문자열 리스트로 리턴

```
>>> import re

>>> text = “I like orange! I love orange!”
>>> result = re.finditer(“orange”, text)
>>> for each in result:
>>>     print(each)
<re.Match object; span=(7, 13), match='orange'>
<re.Match object; span=(22, 28), match='orange'>
```

소스코드
99-1 정규표현식.ipynb

실습

해당 기사에서 네이버가 총 몇번 나오는지
정규표현식을 이용 파이썬에서 확인

http://regexr.com/4gntf

소스코드
99-1 정규표현식.ipynb

정규표현식 (문자)

문자나 문자열 검색가능

표현식 : orange

I like orange! I like orange!

https://regexr.com/4cfc5

표현식 : like orange

I like orange! I like orange!

https://regexr.com/4cfce

정규표현식 (^, $)

^ - 문자열의 시작

$ - 문자열의 끝

표현식 : ^I like

I like orange! I like orange!

https://regexr.com/4cfct

표현식 : orange!$

I like orange! I like orange!

https://regexr.com/4cfd3

정규표현식 (특수문자 사용시 ₩)

₩ ^ $ * + ? . [ ] ( ) | : , - 등등

표현식 : ₩$

I like orange! 200$ I like orange! 200$

https://regexr.com/4cfdu

정규표현식 (.)

모든문자 (문자, 숫자, 공백포함)

표현식 : .

I like orange! 200$ I like orange! 200$

문자 하나씩 추출

https://regexr.com/4cfea

표현식 : ....

I like orange! 200$ I like orange! 200$

4개씩 잘라서 추출

https://regexr.com/4cfe7

정규표현식 ([])

범위 판단

표현식 : [orn]

I like orange! 200$ I like orange! 200$

문자 하나씩 추출

https://regexr.com/4cfih

표현식 : [orn][orn].

I like orange! 200$ I like orange! 200$

https://regexr.com/4cfik

정규표현식 ([])

범위 판단

표현식 : [0-9]

I like orange! 200$ I like orange! 200$

문자 하나씩 추출

https://regexr.com/4cfiq

표현식 : [A-Za-z]

I like orange! 200$ I like orange! 200$

영문만

https://regexr.com/4cfik

정규표현식 ([])

범위 판단

표현식 : [가-힣]

I like orange! 200$ 오렌지! 200$

한글만

https://regexr.com/4cfj6

표현식 : [^A-Za-z]

I like orange! 200$ I like orange! 200$

영문만제외

https://regexr.com/4cfjf

정규표현식 (())

그룹

표현식 : (orange)

I like orange! 200$ I like orange! 200$

https://regexr.com/4cfkv

표현식 : (orange|like)

I like orange! 200$ I like orange! 200$

https://regexr.com/4cfl2

정규표현식 (?*+)

? - 문자가 0개 또는 1개
* - 문자가 0개이상
+ - 문자가 1개이상

정규표현식 (?*+)

표현식 : a.c

I like orange! abc I like orange! ac

https://regexr.com/4cfl8

표현식 : a.?c

I like orange! abc I like orange! ac

https://regexr.com/4cflk

정규표현식 (?*+)

표현식 : ab*c

I like orange! abc I like orange! ac abbbc

https://regexr.com/4cflq

표현식 : ab+c

I like orange! abc I like orange! ac abbbc

https://regexr.com/4cflt

정규표현식 ([]*, []+)

표현식 : [^]+

I like orange! abc I like orange! ac abbbc

https://regexr.com/4cfmf

표현식 : a[bd]*c

I like orange! abdc I like orange! ac abdbdc

https://regexr.com/4cfmi

정규표현식 ({})

개수

표현식 : .{5}

I like orange! 200$ I like orange! 200$

5개씩 끊어서

https://regexr.com/4cfkv

표현식 : [abc]{3}

abcc abc ab bab abccabcab

https://regexr.com/4cfmr

정규표현식 (₩d, ₩D, ₩w, ₩W, ₩s, ₩S)

₩d - [0-9]
₩D - [^0-9]
₩w - [a-zA-Z0-9_]
₩W - [^a-zA-Z0-9_]
₩s - [₩t₩n₩r₩f₩v] 공백, 탭, 라인피드, 캐리지리턴, 폼피드, 수직탭
₩S - [^ ₩t₩n₩r₩f₩v]

정규표현식 (₩d, ₩D, ₩w, ₩W, ₩s, ₩S)

표현식 : ₩w

I like orange! 200$ I like orange! 200$

https://regexr.com/4cfng

표현식 : ₩d

I like orange! 200$ I like orange! 200$

https://regexr.com/4cfnp

정규표현식 (?=, ?<=)

긍정형 전방탐색, 긍정형 후방탐색

표현식 : oran(?=ge!)

I like orange! 200$ I like orange! 200$

https://regexr.com/4chn2

표현식 : (?<=ora)nge!

I like orange! 200$ I like orange! 200$

https://regexr.com/4chnn

정규표현식 (?!, ?<!)

부정형 전방탐색, 부정형 후방탐색

표현식 : oran(?!ge!)

I like orange 200$ I like orange! 200$

https://regexr.com/4chn2

표현식 : (?<!ora)nge!

I like range! 200$ I like orange! 200$

https://regexr.com/4chnn

파이썬에서 전화번호찾기

https://regexr.com/4chor

```
>>> import re

>>> numbers = """
010-2334-3234
02-302-3033
010-1321-4043
02-01-32
33-3303-3033
016-444-3042
"""
```

```
>>> results = re.findall("[0-9]{3}-[0-9]{3,4}-[0-9]{4}", numbers)

>>> for result in results:
>>>     print(result)
```

```
>>> results = re.finditer("[0-9]{3}-[0-9]{3,4}-[0-9]{4}", numbers)

>>> for result in results:
>>>     print(result.group())
```

소스코드
99-1 정규표현식.ipynb

파이썬에서 필요없는 부분제거

https://regexr.com/4chq5

re.sub(정규식, 치환할문자, 대상문자)

```
>>> results = re.sub("[\[.+\]]","", text)
```

소스코드
99-1 정규표현식.ipynb

re.split()

입력된 정규 표현식을 기준으로 문자열들을 분리하여 리스트로 리턴

```
>>> import re

>>> text = “apple, orange! banana pineapple”
>>> result = re.split(“[,! ]”, text)
>>> print(result):
['apple', '', 'orange', '', 'banana', 'pineapple']
```

탐욕적 혹은 게으른

탐욕적 수량자	게으른 수량자
*	*?
+	+?
{n,}	{n,}?

```
>>> import re

>>> text = “<html><head><title>Title</title>”
>>> print(print(re.match('<.*>', text).group()):
<html><head><title>Title</title>

>>> print(re.match('<.*?>', s).group())
<html>
```

실습01

https://regexr.com/4chri

정상적인 이메일만 추출해주세요

결과 :

jkilee@gmail.com

kttredef@naver.com

adekik@best.kr

adefgree@korea.co.kr

소스코드

99-1 실습01.ipynb

실습02

https://regexr.com/4rdvb

텍스트중에 <내용> 괄호로 묶여진 텍스트를 괄호 포함 모두 제거해주세요

결과 :

안녕하세요 저는 홍길동입니다. 나이는 24살 세계 최고의 데이터 분석가가 되고싶습니다.

소스코드

99-1 실습02.ipynb

실습03

https://regexr.com/4rdve

1. 정규표현식을 이용 내용 을 각각 추출
2. 추출된 항목에서 과 태그를 모두제거
3. 각각 총 3개의 항목을 리스트에 넣기

결과

["네이버가 뉴스 서비스에 인공지능(AI)을 도입해 페이지 뷰(PV)를 늘리고 이용자를 끌어 모으고 있다. ",
"네이버는 5일 오전 서울 강남구 그랜드 인터컨티넨털 호텔에서 AI 콜로키움 2019를 열고 이 같은 AI 성과와 전략을 소개했다.",
"이날 기조연설에서 김광현 네이버 서치앤클로바 리더는 "AI 뉴스 추천 시스템인 에어스(AiRS)를 도입하면서 뉴스 소비량이 확대되고 있다" 고 말했다."]

심화 : 위의 1, 2 과정을 하나의 정규식으로 해결해보세요

소스코드
99-1 실습03.ipynb