Lab. Using search()

1. 사용 tool
    -Jupyter Notebook
    -Microsoft Visual Studio Code

2. Code

```
    #re.search()
    #문자열의 일부분이 정규 표현식과 matching되는지 확인하는 method
    #첫번째로 pattern을 찾으면 match 객체 반환
    #못찾으면 None 반환
    #matching되는 문자열의 앞부분에 있지 않다면 match() 대신에 search()를 사용하는 것이 좋다.

    import re

    result = re.search(r'abc', 'abcdef')
    print(type(result))   #<class 're.Match'>

    print(result.start())   #0
    print(result.end())     #3
    print(result.group())  #abc

    result = re.search(r'abc', '123abcdef')
    print(result.start())   #3
    print(result.end())     #6
    print(result.group())  #abc

    result = re.search(r'abc', '123abdef')
    print(result)   #None


    result = re.search(r'\d\d', '123abcdef321')
    print(result)   #<re.Match object; span=(0, 2), match='12'>

    result = re.search(r'\d\d\d\d', '123abcdef321')
    print(result)   #None

    result = re.search(r'\d\d\d\w', '123abcdef321')
    print(result)   #<re.Match object; span=(0, 4), match='123a'>

    result = re.search(r'..\w\w', '@#$%ABCDabcd')
    print(result)    #<re.Match object; span=(2, 6), match='$%AB'>


    #Metacharacters [] 다루기
    result = re.search(r'[cbm]at', 'cat')
    print(result)    #<re.Match object; span=(0, 3), match='cat'>

    result = re.search(r'[cbm]at', 'bat')
    print(result)   #<re.Match object; span=(0, 3), match='bat'>

    result = re.search(r'[0-9]hello', '4hello')
    print(result)   #<re.Match object; span=(0, 6), match='4hello'>

    result = re.search(r'[0-7]hello', '9hello')
    print(result)    #None

    result = re.search(r'[abc.^]amera', 'camera')
    print(result)    #<re.Match object; span=(0, 6), match='camera'>

    result = re.search(r'[abc.^]amera', '.amera')
    print(result)    #<re.Match object; span=(0, 6), match='.amera'>

    result = re.search(r'[abc.^]amera', 'damera')
    print(result)    #None

    result = re.search(r'[^abc]amera', 'camera')
    print(result)    #None

    result = re.search(r'[^abc]amera', 'damera')
    print(result)    #<re.Match object; span=(0, 6), match='damera'>


    #Special Character Classes \ 다루기
    result = re.search(r'\sand ', 'Apple and Banana')
    print(result)    #<re.Match object; span=(5, 10), match=' and '>

    result = re.search(r'\Sand ', 'Apple and Banana')
    print(result)    #None

    result = re.search(r'\Sand ', 'Apple sand Banana')
    print(result)    #<re.Match object; span=(5, 11), match='sand '>

```

```

# .(모든문자) 다루기
result = re.search(r'.and', 'land')
print(result)    #<re.Match object; span=(0, 4), match='land'>

result = re.search(r'\.and', 'land')
print(result)    #None

result = re.search(r'd.g', 'dog')
print(result)    #<re.Match object; span=(0, 3), match='dog'>


#Repetition Cases(반복패턴) 다루기
result = re.search(r'a[bcd]*b', 'abcbdccb')
print(result)    #<re.Match object; span=(0, 8), match='abcbdccb'>

result = re.search(r'b\w+a', 'banana')
print(result)     #<re.Match object; span=(0, 6), match='banana'>

result = re.search(r'i+', 'piigiii')
print(result)     #<re.Match object; span=(1, 3), match='ii'>

result = re.search(r'pi+g', 'piig')
print(result)     #<re.Match object; span=(0, 4), match='piig'>

result = re.search(r'pi+g', 'pg')
print(result)     #None

result = re.search(r'pi*g', 'pg')
print(result)     #<re.Match object; span=(0, 2), match='pg'>

result = re.search(r'https?', 'https://www.google.com')
print(result)     #<re.Match object; span=(0, 5), match='https'>

result = re.search(r'https?', 'httpk://www.google.com')
print(result)     #<re.Match object; span=(0, 4), match='http'>

result = re.search(r'n\w+a', 'carnival')
print(result)     #<re.Match object; span=(3, 7), match='niva'>


#^, $ 다루기
result = re.search(r'^n\w+a', 'carnival')
print(result)     #None

result = re.search(r'^c\w+a', 'carnival')
print(result)     #<re.Match object; span=(0, 7), match='carniva'>

result = re.search(r'c\w+al$', 'carnival')
print(result)     #<re.Match object; span=(0, 8), match='carnival'>

result = re.search(r'c\w+a$', 'carnival')
print(result)     #None


#grouping () 다루기
result = re.search(r'\w+@.+', 'javaexpert@nate.com')
print(result)     #<re.Match object; span=(0, 19), match='javaexpert@nate.com'>
print(result.group())  #javaexpert@nate.com

result = re.search(r'(\w+)@(.+)', 'javaexpert@nate.com')
print(result.group(1))  #javaexpert
print(result.group(2))  #nate.com
print(result.group(0))  #javaexpert@nate.com


#{} 다루기
result = re.search(r'car*al', 'carrrrral')
print(result)     #<re.Match object; span=(0, 9), match='carrrrral'>

result = re.search(r'car{3}al', 'carrrrral')
print(result)     #None

result = re.search(r'car{3}al', 'carrral')
print(result)     #<re.Match object; span=(0, 7), match='carrral'>

result = re.search(r'car{3,5}al', 'carrrrral')
print(result)     #<re.Match object; span=(0, 9), match='carrrrral'>


#Minimum matching
result = re.search(r'<.+>', '<body>hello</body>')
print(result)     #<re.Match object; span=(0, 18), match='<body>hello</body>'>

```

```
result = re.search(r'<.+?>', '<body>hello</body>')
print(result)     #<re.Match object; span=(0, 6), match='<body>'>

result = re.search(r'a{3,5}', 'aaaaa')
print(result)     #<re.Match object; span=(0, 5), match='aaaaa'>

result = re.search(r'a{3,5}?', 'aaaaa')
print(result)     #<re.Match object; span=(0, 3), match='aaa'>


#flag 다루기
result = re.search(r'[a-z]+', '0010010 Has at least one 010 letter 0010010', re.I)
print(result)  #<re.Match object; span=(8, 11), match='Has'>

result = re.search(r'[a-z]+', '0010010 Has at least one 010 letter 0010010')
print(result)  #<re.Match object; span=(9, 11), match='as'>

line = "Cats are smarter than dogs";
searchObj = re.search( r'(.*) are (.*?) .*', line, re.M|re.I)

if searchObj:
    print("searchObj.group() : ", searchObj.group())
    print("searchObj.group(1) : ", searchObj.group(1))
    print("searchObj.group(2) : ", searchObj.group(2))
else:
    print("Nothing found!!")
```