



## 📖 점프 투 파이썬

(/book/1)

### 00장 들어가기 전에

00-1 머리말

00-2 저자소개

00-3 주요변경이력

00-4 책 구입 안내

### 01장 파이썬이란 무엇인가?

01-1 파이썬이란?

01-2 파이썬의 특징

01-3 파이썬으로 무엇을 할 수 있을까?

01-4 파이썬 설치하기

01-5 파이썬 둘러보기

01-6 파이썬과 에디터

### 02장 파이썬 프로그래밍의 기초, 자료형

02-1 숫자형

02-2 문자열 자료형

02-3 리스트 자료형

02-4 튜플 자료형

02-5 딕셔너리 자료형

02-6 집합 자료형

02-7 불 자료형

02-8 자료형의 값을 저장하는 공간, 변수
02장 연습문제
03장 프로그램의 구조를 쌓는다! 제어문
03-1 if문
03-2 while문
03-3 for문
03장 연습문제
04장 프로그램의 입력과 출력은 어떻게 해야 할까?
04-1 함수
04-2 사용자 입력과 출력
04-3 파일 읽고 쓰기
04장 연습문제
05장 파이썬 날개달기
05-1 클래스
05-2 모듈
05-3 패키지
05-4 예외 처리
05-5 내장 함수
05-6 라이브러리
05장 연습문제
06장 파이썬 프로그래밍, 어떻게 시작해야 할까?
06-1 내가 프로그램을 만들 수 있을까?

06-2 3과 5의 배수 합하기
06-3 게시판 페이지징하기
06-4 간단한 메모장 만들기
06-5 탭을 4개의 공백으로 바꾸기
06-6 하위 디렉터리 검색하기
06-7 파이보
06-8 코딩도장
07장 정규표현식
07-1 정규 표현식 살펴보기
07-2 정규 표현식 시작하기
07-3 강력한 정규 표현식의 세계로
08장 종합문제
09장 풀이
10장 마치며

Published with WikiDocs (/)

[■ 점프 투 파이썬 \(/book/1\)](#) / [07장 정규표현식 \(/1669\)](#) / [07-1 정규 표현식 살펴보기 \(/1642\)](#)[🏠 WikiDocs \(/\)](#)

## 07-1 정규 표현식 살펴보기

정규 표현식(Regular Expressions)은 복잡한 문자열을 처리할 때 사용하는 기법으로, 파이썬만의 고유 문법이 아니라 문자열을 처리하는 모든 곳에서 사용한다. 정규 표현식을 배우는 것은 파이썬을 배우는 것과는 또 다른 영역의 과제이다.

※ 정규 표현식은 줄여서 간단히 "정규식"이라고도 말한다.

## 정규 표현식은 왜 필요한가?

다음과 같은 문제가 주어졌다고 가정해 보자.

주민등록번호를 포함하고 있는 텍스트가 있다. 이 텍스트에 포함된 모든 주민등록번호의 뒷자리를 \* 문자로 변경해 보자.

우선 정규식을 전혀 모르면 다음과 같은 순서로 프로그램을 작성해야 할 것이다.

1. 전체 텍스트를 공백 문자로 나눈다(split).
2. 나뉜 단어가 주민등록번호 형식인지 조사한다.
3. 단어가 주민등록번호 형식이라면 뒷자리를 \* 로 변환한다.
4. 나뉜 단어를 다시 조립한다.

이를 구현한 코드는 아마도 다음과 같을 것이다.

```
data = """
park 800905-1049118
kim 700905-1059119
"""

result = []
for line in data.split("\n"):
    word_result = []
    for word in line.split(" "):
        if len(word) == 14 and word[:6].isdigit() and word[7:].isdigit():
            word = word[:6] + "-" + "*****"
        word_result.append(word)
    result.append(" ".join(word_result))
print("\n".join(result))
```

결과값:

```
park 800905-*****
kim 700905-*****
```

반면에 정규식을 사용하면 다음처럼 훨씬 간편하고 직관적인 코드를 작성할 수 있다. 아직 정규식 사용 방법을 배우지 않았으니 눈으로만 살펴보자.

```
import re

data = """
park 800905-1049118
kim 700905-1059119
"""

pat = re.compile("(\\d{6})[-]\\d{7}")
print(pat.sub("\\g<1>-*****", data))
```

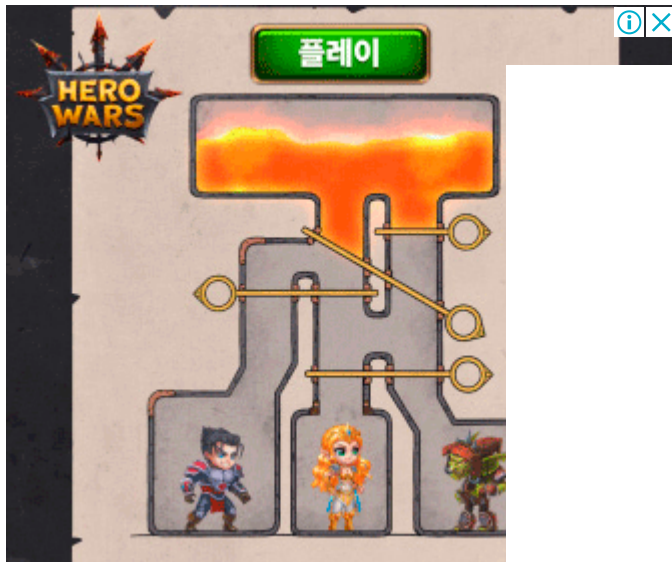
결과값:

```
park 800905-*****
kim 700905-*****
```

정규 표현식을 사용하면 이렇게 간단한 예제에서도 코드가 상당히 간결해진다. 만약 찾으려는 문자열 또는 바꾸어야 할 문자열의 규칙이 매우 복잡하다면 정규식의 효용은 더 커지게 된다.

이제부터 정규 표현식의 기초부터 심화 부분까지 차근차근 알아보자.

마지막 편집일시 : 2020년 3월 13일 5:28 오후





(<https://wikidocs.net/105844>)

댓글 18 피드백

- 이전글 : 07장 정규표현식
- 다음글 : 07-2 정규 표현식 시작하기

↑ TOP