# 18차 연습문항\_답안

## **파이썬 심화문항: 정규표현**

|  |
| --- |
| 수강생 공지 사항   * 제출은 프로그래머스를 통해 해주시기 바랍니다([링크](https://campus.programmers.co.kr/app/courses/24550/curriculum)). * 파일명은 아래와 같은 형식으로 제출해주세요   + 교육생번호\_이름\_교과목\_문항\_N차시.pdf   ex) DR-11111\_홍길동\_파이썬\_연습문항\_1차시.pdf  ex) DR-11111\_홍길동\_파이썬\_추가문항\_1차시.pdf   * 답은 "write your answer"에 적어주세요. 다만 코딩 문제의 경우 output까지 답에 포함시켜 주세요.   ex)     * 답은 캡쳐를 하셔도 되고 텍스트로 넣으셔도 됩니다. * 마감 기한은 문제가 나간 주 **일요일 23:59까지**입니다. |

### 정규표현식에 대한 설명 중 틀린 것은?

A. 정규표현식을 사용하면 특정 패턴을 가진 문자열을 손쉽게 찾을 수 있습니다.

B. 파이썬에서만 지원되는 강력한 문자열 관리 기능입니다.

C. 반복적인 텍스트 처리 작업을 자동화할 수 있습니다.

D. 단순한 문자열 검색 기능으로는 구현하기 어려운 복잡한 검색 조건을 정규표현식을 통해 구현할 수 있습니다.

|  |
| --- |
| B |

### 다음 코드가 어떤 작업을 하는지 생각해 보고 결과를 확인 하시오.

|  |
| --- |
| 'Python' found in: "I love programming in Python."  'Python' not found in: "JavaScript is a versatile language."  'Python' found in: "The Python snake is a non-venomous species." |

### 아래의 보기 중 결과가 참이 아닌 것을 고르시오.

A. re.match('[0-9]+', '2bab6d')

B. re.match('a+b', 'a')

C. re.search('ou$', 'Good to see you')

D. re.match('ab[0-9]+?c', 'ab56c')

|  |
| --- |
| B |

### 아래 문자열 안에 이메일 주소들이 들어 있습니다. 정규표현식(Regex)을 이용해 이메일 주소를 찾는 코드를 작성하시오.

text = "alice@example.com 이 첫 번째 이메일이고, 다음은 bob.smith@mail.co.uk 입니다."

|  |
| --- |
| import re    text = "alice@example.com 이 첫 번째 이메일이고, 다음은 bob.smith@mail.co.uk 입니다."    # 일반 문자열로 이메일 패턴을 정의  pattern = '[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}'    # match 메소드를 이용해 문자열의 시작에서 패턴에 매칭되는지 확인  match = re.match(pattern, text)    if match:      print(match.group())  else:      print("문자열의 시작 부분에 이메일이 없습니다.") |

### 아래 문자열이 특정 단어('Hello')로 시작하는지 확인하는 코드를 작성하시오.

text = "Hello, how are you today? Hello, I'm fine."

|  |
| --- |
| import re  text = "Hello, how are you today? Hello, I'm fine."  # 일반 문자열로 'Hello'로 시작하는지 확인하는 패턴 정의  pattern = '^Hello'  # match 메소드를 이용해 문자열의 시작에서 패턴에 매칭되는지 확인  match = re.match(pattern, text)  if match:      print("문자열은 'Hello'로 시작합니다.")  else:      print("문자열은 'Hello'로 시작하지 않습니다.") |

### 아래 문자열에 특정 단어(‘sample')가 있는지 확인하는 코드를 작성하시오.

text = "This is a sample text with the word 'sample' in it."

|  |
| --- |
| import re  text = "This is a sample text with the word 'sample' in it."  # 일반 문자열로 'sample'이라는 단어를 찾기 위한 패턴 정의  pattern = '\\bsample\\b'  # search 메소드를 이용해 첫 번째 패턴 매칭을 찾음  match = re.search(pattern, text)  if match:      print("문자열 내에 'sample' 단어가 존재합니다.")  else:      print("문자열 내에 'sample' 단어가 존재하지 않습니다.") |

### 아래 문자열에서 날짜 형식을 가진 문자열이 있는지 확인하는 코드를 작성하시오.

text = "오늘의 날짜는 2023-06-28이고, 내일은 2023-06-29입니다."

|  |
| --- |
| Example Code |
| import re  text = "오늘의 날짜는 2023-06-28이고, 내일은 2023-06-29입니다."  # 일반 문자열로 날짜 형식 패턴 정의  pattern = r'\d{4}-\d{2}-\d{2}'  # findall 메소드를 이용해 모든 패턴 매칭을 찾음  matches = re.findall(pattern, text)  if matches:      print("찾은 날짜들:", matches)  else:      print("날짜 형식을 찾을 수 없습니다.") |

|  |
| --- |
| write your answer |

### 아래의 문자열에 있는 모든 이메일 주소를 raw 문자열 패턴을 사용하여 찾는 코드를 작성해 보시오.

text = "다음 사람들에게 이메일을 보내세요: alice@example.com, bob.smith@mail.co.uk, [charlie123@domain.org](mailto:charlie123@domain.org)."

이메일 형식은 아래와 같다.

* 이메일은 영문자, 숫자, @, 마침표로만 구성되어 있다.
* 이메일은 @와 마침표를 구분자로 사용하여 크게 3부분으로 구성되어 있다.

|  |
| --- |
| import re    text = "다음 사람들에게 이메일을 보내세요: alice@example.com, bob.smith@mail.co.uk, charlie123@domain.org."    pattern = r'[a-zA-Z0-9.]+@[a-zA-Z0-9.]+\.[a-zA-Z]{2,}'  emails = re.findall(pattern, text)    print(emails) |

### 아래의 문자열에 있는 모든 소수점을 포함한 숫자를 찾는 코드를 작성하시오.

text = "상품의 가격은 19.99달러와 100.0달러, 0.99달러 입니다."

|  |
| --- |
| import re    text = "상품의 가격은 19.99달러와 100.0달러, 0.99달러 입니다."    pattern = r'\d+\.\d+'  decimals = re.findall(pattern, text)    print(decimals) |

### 아래의 문자열(HTML)에서 모든 태그를 찾는 코드를 작성하시오.

text = "<html><head><title>Test</title></head><body><h1>Header</h1><p>Paragraph</p></body></html>"

|  |
| --- |
| import re    text = "<html><head><title>Test</title></head><body><h1>Header</h1><p>Paragraph</p></body></html>"    pattern = r'<[^>]+>'  tags = re.findall(pattern, text)    print(tags) |

### 패스워드 규칙을 검사하는 정규표현식을 사용한 코드를 작성해 보시오. - 패스워드 규칙은 보안을 강화하기 위해 다양한 조건을 만족해야 할 수 있습니다. 최소길이, 대소문자 포함 여부, 숫자 포함 여부, 특수문자 포함 여부 등이 있습니다.

- 패스워드 규칙을 검사하는 함수(함수명: check\_password\_strength)를 만들고, 패스워드를 입력하면 패스워드의 통과여부를 알려준다.

|  |
| --- |
| import re  def check\_password\_strength(password):      # 최소 8자 이상, 숫자와 특수문자를 포함하는지 확인하는 정규표현식      pattern = r'^(?=.\*[0-9])(?=.\*[!@#$%^&\*])(?=.\*[a-zA-Z]).{8,}$'      if re.match(pattern, password):          return True      else:          return False  # 테스트  password1 = "StrongPassword123!"  print(f"Password '{password1}' is strong: {check\_password\_strength(password1)}")  newPswd = input('\n패스워드를 입력하세요: ')  if check\_password\_strength(newPswd):      print('안전한 패스워드입니다.')  else:      print('패스워드의 기준에 맞지 않습니다') |

### 주어진 텍스트 안의 모든 특수문자를 제거하는 정규표현식을 만들어 보세요.

text = "Hello! @world! This is 2024. @#$%^&\*"

|  |
| --- |
| import re  def remove\_special\_characters(text):      # 특수문자 제외하고 알파벳 문자와 숫자만 남기는 정규표현식 패턴      pattern = r'[^a-zA-Z0-9가-힣\s]' # 한글도 포함하여 처리할 경우 추가      # 정규표현식을 사용하여 패턴과 일치하지 않는 문자열을 제거하고 반환      clean\_text = re.sub(pattern, '', text)      return clean\_text  # 테스트  text = "Hello! @world! This is 2024. @#$%^&\*"  clean\_text = remove\_special\_characters(text)  print(clean\_text) # 출력: "Hello world This is 2023" |