문자열 💯

알고리즘 스터디 1주차 강수지

문자열 유형 (Palindrome)

회문 : 앞뒤가 똑같은 단어나 문장

대소문자를 구분하지 않으며 글자와 숫자만을 고려하면 된다 !

앞뒤가 같은 단어를 찾아라 회문인 것을 찾아라 ..
“Palindrome” 라는 단어가 문제에 명시 되기도 한다 !

문자열 유형 (Palindrome)

- 대소문자나 특수기호 포함시 전처리 과정 후 판별진행

- 보통은 문자열의 중점 기준으로 처음과 끝을 비교해가며 회문판별

```
def isPalindrome(str):
    for i in range(len(str)//2):
        if str[i] == str[-i-1]:
            continue
        else: print("회문 X ")

    print("회문 O ")

data = "abcba"
isPalindrome(data)
```

문자열 유형 (뒤집기) 투 포인터를 이용하는 방법

```
def reverseString(str):
    left_idx, right_idx = 0, len(str)-1
    while left_idx < right_idx:
        str[left_idx], str[right_idx] = str[right_idx], str[left_idx]
        left_idx += 1
        right_idx -= 1
    print(str)


a = ['a', 'b', 'c', 'd', 'e']
reverseString(a)
```

리스트에서 제공하는 **reverse()** 함수나 슬라이싱 **a[::-1]** 과 동일한 결과

문자열 유형 (재정렬)

Sort 이용하는 방법

```
# 문자순 정렬
data = ['1 A', '1 B', '6 A', '2 D', '4 B']

data.sort(key=lambda x: ( x.split()[1], x.split()[0] ))
```

문자열 유형 (애너그램)

```
import collections

data = ["eat","tea","tan","ate","nat","bat"]
sort_data = collections.defaultdict(list)
for word in data:
    # 'a' 'e' 't' -> 'aet'
    sort_data[''.join(sorted(word))].append(word)
```

문자를 재배열하여 다른 뜻의 단어로 바꾸는 것

알짜배기 문자열 처리 함수들

- isalnum() : 영문자, 숫자 여부 판별할 때
- isalpha() : 문자인지 확인할 때
- isdigit() : 숫자인지 확인할 때

- islower() : 소문자인지 확인할 때
- isupper(): 대문자인지 확인할 때

알짜배기 문자열 처리 함수들

- find() : 찾는 문자열이 처음 나오는 index 리턴 / 없을 경우 -1
- count() : 해당 문자열의 개수를 리턴
- .replace(문자열1, 문자열2) : 문자열 1을 문자열 2로 변환

- .split(문자열) : 문자열을 기준으로 분리 / 생략할 경우 공백기준
- .join(문자열) : 리스트 원소 사이 문자열을 삽입

정규표현식

import re *# 파이썬 파일내에서 사용할 경우*

```
p = re.compile('[a-z]+')
```

함수	기능	사용 예시	반환값
`match`	문자열의 처음부터 정규식과 매치되는지 조사	`p.match('python').group()`	'python'
`search`	문자열 전체를 검색하여 정규식과 매치되는지 조사	`p.search('123 python').group()`	'python'
`findall`	정규식과 매치되는 모든 문자열을 리스트로 반환	`p.findall('hello my name is')`	['hello', 'my', 'name', 'is']

실제 출제된 문자열 1

2018 카카오 블라인드 채용 1차 "다트게임“

1. 다트 게임은 총 3번의 기회로 구성된다.
2. 각 기회마다 얻을 수 있는 점수는 0점에서 10점까지이다.
3. 점수와 함께 Single(`S`), Double(`D`), Triple(`T`) 영역이 존재하고 각 영역 당첨 시 점수에서 1제곱, 2제곱, 3제곱 (점수^1 , 점수^2 , 점수^3)으로 계산된다.
4. 옵션으로 스타상(`*`) , 아차상(`#`)이 존재하며 스타상(`*`) 당첨 시 해당 점수와 바로 전에 얻은 점수를 각 2배로 만든다. 아차상(`#`) 당첨 시 해당 점수는 마이너스된다.
5. 스타상(`*`)은 첫 번째 기회에서도 나올 수 있다. 이 경우 첫 번째 스타상(`*`)의 점수만 2배가 된다. (예제 4번 참고)
6. 스타상(`*`)의 효과는 다른 스타상(`*`)의 효과와 중첩될 수 있다. 이 경우 중첩된 스타상(`*`) 점수는 4배가 된다. (예제 4번 참고)
7. 스타상(`*`)의 효과는 아차상(`#`)의 효과와 중첩될 수 있다. 이 경우 중첩된 아차상(`#`)의 점수는 -2배가 된다. (예제 5번 참고)
8. Single(`S`), Double(`D`), Triple(`T`)은 점수마다 하나씩 존재한다.
9. 스타상(`*`), 아차상(`#`)은 점수마다 둘 중 하나만 존재할 수 있으며, 존재하지 않을 수도 있다.

전체 탐색하여 구현

2018 카카오 블라인드 채용 1차 "다트게임"

```
def solution(dartResult):
    n = ''
    score = []
    for i in dartResult:
        if i.isnumeric():
            n += i
        elif i == 'S':
            n = int(n)**1
            score.append(n)
            n = ''
        elif i == 'D':
            n = int(n)**2
            score.append(n)
            n = ''
        elif i == 'T':
            n = int(n)**3
            score.append(n)
            n = ''
        elif i == '*':
            if len(score) > 1:
                score[-2] = score[-2] * 2
                score[-1] = score[-1] * 2
            else:
                score[-1] = score[-1] * 2
        elif i == '#':
            score[-1] = score[-1] * -1

    return sum(score)
```

정규표현식을 사용하기

2018 카카오 블라인드 채용 1차 "다트게임"

```python
import re



def solution(dartResult):
    bonus = {'S' : 1, 'D' : 2, 'T' : 3}
    option = {'' : 1, '*' : 2, '#' : -1}
    p = re.compile('(\d+)([SDT])([*#]?)')
    dart = p.findall(dartResult)
    for i in range(len(dart)):
        if dart[i][2] == '*' and i > 0:
            dart[i-1] *= 2
        dart[i] = int(dart[i][0]) ** bonus[dart[i][1]] * option[dart[i][2]]

    answer = sum(dart)
    return answer
```

실제 출제된 문자열 2

2019 카카오 개발자 겨울 인턴쉽 "불량사용자"

[입출력 예]

user_id	banned_id	result
["frodo", "fradi", "crodo", "abc123", "frodoc"]	["fr*d*", "abc1**"]	2
["frodo", "fradi", "crodo", "abc123", "frodoc"]	["*rodo", "*rodo", "******"]	2
["frodo", "fradi", "crodo", "abc123", "frodoc"]	["fr*d*", "*rodo", "******", "******"]	3

Brute Force

[2019 카카오 개발자 겨울 인턴쉽 "불량사용자"]

```python
from itertools import permutations

def check(users,banned_id):
    for i in range(len(banned_id)):
        if len(users[i]) != len(banned_id[i]):
            return False

        for j in range(len(users[i])):
            if banned_id[i][j] == "*":
                continue
            if banned_id[i][j] != users[i][j]:
                return False # 현재 튜플 불일치

    return True
```

```python
def solution(user_id, banned_id):
    user_permutation = list(permutations(user_id,len(banned_id)))
    banned_Set = []

    for users in user_permutation:
        # 하나의 튜플과 비교 시작
        if not check(users, banned_id):
            continue # 다음 튜플 가져오기
        else:
            users = set(users)
            if users not in banned_Set:
                banned_Set.append(users)

    return len(banned_Set)
```

정규표현식을 사용하기

2019 카카오 개발자 겨울 인턴쉽 "불량사용자"

```python
import re
import itertools

def solution(user_id, banned_id):

    answer = 0

    banned_id = ["'" + _.replace('*', '\w') + "'"  for _ in banned_id]

    possible_list = [re.findall(_, str(user_id)) for _ in banned_id]

    possible_list = list(itertools.product(*possible_list))

    possible_list = [frozenset(p) for p in possible_list if len(set(p)) == len(banned_id)]
    possible_list = set(possible_list)
    return len(possible_list)
```

실제 출제된 문자열 3

2021 카카오 블라인드 채용 " 신규 아이디 추천“

```
1단계 new_id의 모든 대문자를 대응되는 소문자로 치환합니다.
2단계 new_id에서 알파벳 소문자, 숫자, 빼기(-), 밑줄(_), 마침표(.)를 제외한 모든 문자를 제거합니다.
3단계 new_id에서 마침표(.)가 2번 이상 연속된 부분을 하나의 마침표(.)로 치환합니다.
4단계 new_id에서 마침표(.)가 처음이나 끝에 위치한다면 제거합니다.
5단계 new_id가 빈 문자열이라면, new_id에 "a"를 대입합니다.
6단계 new_id의 길이가 16자 이상이면, new_id의 첫 15개의 문자를 제외한 나머지 문자들을 모두 제거합니다.
      만약 제거 후 마침표(.)가 new_id의 끝에 위치한다면 끝에 위치한 마침표(.) 문자를 제거합니다.
7단계 new_id의 길이가 2자 이하라면, new_id의 마지막 문자를 new_id의 길이가 3이 될 때까지 반복해서 끝에 붙입니다.
```

Lower() replace() 등 슬라이싱으로 단순 구현

2021 카카오 블라인드 채용 " 신규 아이디 추천"

```python
def solution(new_id):
    # 1단계
    new_id = new_id.lower()
    # 2단계
    answer = ''
    for word in new_id:
        if word.isalnum() or word in '-_.':
            answer += word
    # 3단계
    while '..' in answer:
        answer = answer.replace('..', '.')
    # 4단계
    answer = answer[1:] if answer[0] == '.' and len(answer) > 1 else answer
    answer = answer[:-1] if answer[-1] == '.' else answer
    # 5단계
    answer = 'a' if answer == '' else answer
    # 6단계
    if len(answer) >= 16:
        answer = answer[:15]
        if answer[-1] == '.':
            answer = answer[:-1]
    # 7단계
    if len(answer) < 3:
        answer = answer + answer[-1] * (3-len(answer))
    return answer
```

정규표현식을 사용하기

2021 카카오 블라인드 채용 " 신규 아이디 추천"

```python
import re


def solution(new_id):
    st = new_id
    st = st.lower()
    st = re.sub('[^a-z0-9\-_.]', '', st)
    st = re.sub('\.+', '.', st)
    st = re.sub('^[.]|[.]$', '', st)
    st = 'a' if len(st) == 0 else st[:15]
    st = re.sub('^[.]|[.]$', '', st)
    st = st if len(st) > 2 else st + "".join([st[-1] for i in range(3-len(st))])
    return st
```

문자열 공통 문제

- 백준 1493 뒤집기 실버 4
- 백준 1181 단어 정렬 실버 5
- 백준 9935 문자열 폭발 골드 4

출처

실제 출제된 문자열 문제

- 2018 카카오 블라인드 채용 1차 "다트게임"
- 2019 카카오 개발자 겨울 인턴쉽 "불량사용자"
- 2019 카카오 블라인드 채용 1차 "매칭 점수"

정규표현식

- https://nachwon.github.io/regular-expressions/
- https://velog.io/@beanlove97/%ED%8C%8C%EC%9D%B4%EC%8D%AC-findall-%EB%A9%94%EC%84%9C%EB%93%9C
- https://yurimkoo.github.io/analytics/2019/10/26/regular_expression.html

감사합니다