프로그래머스 - 숨어있는 숫자의 덧셈 (1)

태그

Character.digit filter() map(v=>+v) mapToInt() mapToObj()
match(/\d/g) reduce() reduce() 콜백함수 replaceAll split

자바스크립트
숨어있는 숫자의 덧셈 (1)
문제 설명
문자열 my_string이 매개변수로 주어집니다. my_string안의 모든 자연수들의 합을 return하도록 solution 함수를 완성해주세요.

제한사항
1 ≤ my_string의 길이 ≤ 1,000
my_string은 소문자, 대문자 그리고 한자리 자연수로만 구성되어있습니다.

입출력 예 #1
my_string : "aAb1B2cC34oOp" , result: 10
"aAb1B2cC34oOp"안의 한자리 자연수는 1, 2, 3, 4 입니다. 따라서 1 + 2 + 3 + 4 = 10 을 return합니다.
입출력 예 #2
my_string : "1a2b3c4d123" , result: 16
"1a2b3c4d123Z"안의 한자리 자연수는 1, 2, 3, 4, 1, 2, 3 입니다. 따라서 1 + 2 + 3 + 4 + 1 + 2 + 3 = 16 을 return합니다.
유의사항
연속된 숫자도 각각 한 자리 숫자로 취급합니다.

```
function solution(my_string) {
  // 정규 표현식을 사용하여 문자열에서 모든 한 자리 자연수를 찾습니다.
  const numbers = my_string.match(/\d/g);

  // 숫자가 없으면 0을 반환
  if (!numbers) return 0;

  // 숫자를 합산, reduce() 함수
  return numbers.reduce((sum, number) => sum + parseInt(number, 10), 0);
}
```

정규식을 사용하여 문자열에서 모든 한 자리 자연수를 찾은 다음 reduce() 함수를 사용하여 합계를 낼 수 있습니다.먼저 정규식을 사용하여 문자열에서 모든 한 자리 자연수를 찾는 방식으로 작동합니다. 정규식 /\d/g는 문자열의 모든 숫자(0-9)와 일치하고 g 플래그는 모든 항목이 일치해야 함을 지정합니다(첫 번째 항목만 일치하는 것과는 반대). match() 함수는 모든 일치 항목의 배열을 반환하거나 일치 항목이 없으면 null을 반환합니다.
if 문을 사용하여 숫자가 있는지 확인합니다. 숫자가 없으면 함수는 0을 반환합니다. 그렇지 않으면 reduce() 함수를 사용하여 숫자를 합산합니다. reduce() 함수는 배열의 각 요소에 함수를 적용하여 배열을 단일 값으로 줄입니

다. (sum, number) => sum + parseInt(number, 10) 이며 현재 합을 취하여 현재 숫자를 더합니다. reduce()의 두 번째 인수는 합계의 초기 값이며 이 경우 0입니다. 마지막으로 함수는 숫자의 합계를 반환합니다.

reduce() 함수는 배열의 숫자를 합산하는 데 사용됩니다.

reduce() 함수는 콜백 함수와 초기 값의 두 가지 인수를 사용합니다. 콜백 함수는 배열의 각 요소에 적용되며 초기 값은 합계의 시작 값으로 사용됩니다. 콜백 함수는 현재 합계인 sum과 처리 중인 현재 요소인 number의 두 가지 인수를 사용합니다.

콜백 함수는 현재 합계(sum)에 현재 요소(숫자)를 더하여 새로운 합계를 반환합니다. parseInt() 함수는 합계에 더하기 전에 숫자 문자열을 정수로 변환하는 데 사용됩니다. 10 인수는 숫자가 10진법임을 지정합니다.
reduce() 함수는 배열을 계속 반복하면서 콜백 함수를 각 요소에 적용하고 배열 끝에 도달할 때까지 합계를 업데이트합니다.
합계의 최종 값은 reduce() 함수에 의해 반환됩니다.

```
function solution(my_string) {
    return my_string.replaceAll(/[^\d]/g, '').split('').map(v=>+v).reduce((a,v)=>a+v,
0);
}
```

my_string.replaceAll 호출은 정규식을 사용하여 문자열에서 모든 숫자가 아닌 문자를 제거합니다. g 플래그는 정규식이 전역적으로 적용되어야 하므로 문자열의 모든 숫자가 아닌 문자가 대체되도록 지정합니다.

그런 다음 결과 문자열은 split 메서드를 사용하여 개별 문자 배열로 분할됩니다. map 함수는 단항 더하기 연산자(+)를 사용하여 배열의 각 문자를 숫자로 변환하는 데 사용됩니다. 마지막으로 reduce 함수는 배열의 모든 숫자를 합산하는 데 사용됩니다. a 매개변수는 누산기를 나타내고 v 매개변수는 처리 중인 현재 값을 나타냅니다. reduce 함수는 제공된 함수를 배열의 첫 번째 요소부터 시작하여 각 요소에 적용하고 결과를 누적합니다. 이 경우 제공된 함수는 누산기(a)에 현재 값(v)을 더합니다. reduce에 전달된 0 인수는 누산기의 초기 값을 지정합니다.

```
class Solution {
    public int solution(String myString) {
        return myString.chars().mapToObj(i -> (char) i).filter(Character::isDigit).map
(String::valueOf).mapToInt(Integer::valueOf).sum();
    }
}
```

Java 8 Stream API 함수의 조합을 사용하여 문자열에서 모든 한 자리 자연수의 합계를 찾습니다. chars() 메서드는 문자열의 문자를 나타내는 정수 스트림을 반환합니다. mapToObj() 메서드는 정수 스트림을 문자 스트림에 매핑하는 데 사용됩니다.
filter() 메서드는 숫자만 유지하면서 문자 스트림을 필터링하는 데 사용됩니다. map() 메서드는 문자 스트림을 문자열 스트림으로 매핑하는 데 사용됩니다. mapToInt() 메서드는 문자열 스트림을 정수 스트림으로 매핑하는 데 사용됩니다. sum() 메서드는 스트림에 있는 모든 정수의 합을 찾는 데 사용됩니다.

```
class Solution {
    public int solution(String my_string) {
        int answer = 0;
        for(int i=0; i<my_string.length(); i++){
            if(Character.digit(my_string.charAt(i), 10) > 0){
                answer += Integer.parseInt(String.valueOf(my_string.charAt(i)));
            }
        }
        return answer;
    }
}
```

이 메서드는 문자열에 숫자의 합계를 저장하는 데 사용되는 지역 변수 answer을 0으로 초기화합니다.
그런 다음 메서드는 for 루프를 사용하여 문자열의 각 문자를 반복하는 루프에 들어갑니다.
현재 문자에서 Character.digit 메서드가 호출되어 숫자인지 확인합니다. 10 인수는 문자를 10진수로 해석하도록 지정합니다. 문자가 숫자인 경우 메서드는 숫자의 숫자 값을 나타내는 양의 정수를 반환합니다. 문자가 숫자가 아닌 경우 메서드는 -1을 반환합니다.

현재 문자가 숫자인 경우 메서드는 Integer.parseInt 메서드를 사용하여 문자를 int로 변환한 다음 answer 변수에 추가합니다.루프가 완료되면 메서드는 문자열에 있는 모든 숫자의 합계인 answer 변수의 최종 값을 반환합니다.