

# 8. [Java] ??? ????? - Character.isDigit(c)

<https://school.programmers.co.kr/learn/courses/30/lessons/120902>

문제 (10 분 )

```
class Solution {
    public int solution(String my_string) {
        char[] arr = my_string.toCharArray();
        int sum = 0;
        for (char c : arr ) {
            if (c != '+' && c != '-' && c != ' ') {
                sum+= Integer.parseInt(c);
            }
        }
        return sum;
    }
}
```

실행 결과

```
/Solution.java:7: error: incompatible types: char cannot be converted to String
        sum+= Integer.parseInt(c);
                        ^
```

Note: Some messages have been simplified; recompile with -Xdiags:verbose to get full output  
1 error

테스트 결과 (~^~)~

1개 중 0개 성공

## ? Integer.parseInt()

- String → int → Integer.parseInt()
- char → String → Integer.parseInt()

## String → int VS char → int

### 1. String → int    int    Integer.parseInt()

```
String str = "123";

// String → int
int num = Integer.parseInt(str);        // 123
```

### 2. char → int    int    char → int - '0'

```
char c = '7';

// char → int (int)
int digit = c - '0';    // '7' - '0' = 7
```

## 2. 2 (2 2 )

```
class Solution {
    public int solution(String my_string) {
        char[] arr = my_string.toCharArray();
        int sum = 0;
        for (char c : arr ) {
            if (c != '+' && c != '-' && c != ' ') {
                sum+= c - '0';
            }
        }
        return sum;
    }
}
```

## 실행 결과

채점을 시작합니다.

정확성 테스트

```
테스트 1 > 실패 (0.03ms, 98.6MB)
테스트 2 > 실패 (0.03ms, 86.9MB)
테스트 3 > 실패 (0.04ms, 72.1MB)
테스트 4 > 실패 (0.02ms, 83.6MB)
테스트 5 > 실패 (0.03ms, 75MB)
테스트 6 > 실패 (0.04ms, 77.7MB)
테스트 7 > 실패 (0.03ms, 75.8MB)
테스트 8 > 통과 (0.03ms, 72.3MB)
테스트 9 > 실패 (0.02ms, 89.5MB)
테스트 10 > 실패 (0.02ms, 82.1MB)
```

채점 결과

정확성: 10.0

합계: 10.0 / 100.0

문자열 "12 + 3"의 길이를 구하는 프로그램을 작성하시오. 문자열의 길이는 문자의 개수를 의미한다. 예를 들어, "12"의 길이는 2, "12 + 3"의 길이는 6이다.

```
String my_string = "a12b3";
→ arr = ['a', '1', '2', 'b', '3']
→ arr의 길이: 1 + 2 + 3 = 6 (문자 'a'는 1, '1'은 2, '2'는 3)
```

문자열 "12 + 3"의 길이를 구하는 프로그램을 작성하시오. 문자열의 길이는 문자의 개수를 의미한다. 예를 들어, "12"의 길이는 2, "12 + 3"의 길이는 6이다.

문자열 num의 길이를 구하는 프로그램을 작성하시오. Character.isDigit()은 문자 c가 0~9의 숫자인지 여부를 반환한다. num의 길이는 num에 포함된 숫자의 개수를 의미한다.

문자열의 길이 (3점)

```
class Solution {
    public int solution(String my_string) {
        char[] arr = my_string.toCharArray();
        int sum = 0;
        String num = "";
        for (char c : arr ) {
```

```

        if (Character.isDigit(c)) {
            num += c;
        } else {
            if (!num.isEmpty()) {
                sum += Integer.parseInt(num);
                num = "";
            }
        }
    }

    if (!num.isEmpty()) {
        sum += Integer.parseInt(num);
    }

    return sum;
}

```

채점을 시작합니다.

□□□ □□□ □□□ □□□ □□□ . □□□ (+, -) □ □□ □□ □□ □□□ □□ □□ □□ .

4. 문자열이 "+", "-", "\*" 또는 "/"로 구성된 수식 문자열이 주어진다. 이 문자열을 계산하여 결과를 반환하라.

문자열이 주어진다. (4점)

```
class Solution {
    public int solution(String my_string) {
        String[] splited = my_string.split(" ");
        int answer = Integer.parseInt(splited[0]);

        for(int i = 1; i < splited.length; i += 2) {
            String op = splited[i];
            int num = Integer.parseInt(splited[i + 1]);

            if (op == '+') {
                answer += num;
            } else if (op == '-') {
                answer -= num;
            }
        }
        return answer;
    }
}
```

? if(op == '+')

- op 문자열이 "+"일 때, answer에 num을 더한다.
- op 문자열이 "-"일 때, answer에 num을 빼준다.
- Java에서 String을 비교할 때는 op.equals(num)을 사용한다. op == num은 String과 int를 비교하는 것이므로 에러가 발생한다.
- op 문자열이 "\*"일 때, answer에 num을 곱한다.
- op 문자열이 "/"일 때, answer에 num을 나눈다.

문자열이 주어진다. (4점)

```
class Solution {
    public int solution(String my_string) {
        String[] splited = my_string.split(" ");
        int answer = Integer.parseInt(splited[0]);
```

```
for(int i = 1; i < splited.length; i += 2) {  
    String op = splited[i];  
    int num = Integer.parseInt(splited[i + 1]);  
  
    if (op.equals("+")) {  
        answer += num;  
    } else if (op.equals("-")) {  
        answer -= num;  
    }  
}  
return answer;  
}  
}
```

---

Revision #13

Created 19 May 2025 05:34:15 by Dain

Updated 23 May 2025 04:25:10 by Dain