14. [Java] ??? ?? 3

> https://school.programmers.co.kr/learn/courses/30/lessons/181916

1. ??? ?? ?? ??

1. 네 주사위가 모두 같을 경우
   - 조건 : 하나의 숫자가 4번 등장
   - 점수 : 1111 × p
2. 세 개가 같고 하나가 다를 경우
   - 조건 : 한 숫자가 3번 , 다른 숫자가 1번 등장
   - 점수 : $(10 \times p + q)^2$
3. 두 개씩 같은 숫자가 나올 경우
   - 조건 : 두 숫자가 각각 2번씩 등장
   - 점수 : $(p + q) \times |p - q|$
4. 한 쌍과 나머지 두 숫자가 각각 다를 경우
   - 조건 : 한 숫자 2번 , 나머지 두 숫자는 각각 1번 (총 세 가지 숫자)
   - 점수 : 나머지 두 숫자의 곱 $(q \times r)$
5. 모두 다른 숫자가 나올 경우
   - 조건 : 네 숫자가 모두 다름
   - 점수 : 가장 작은 숫자

---

2. ?? ??

```
import java.util.*;


class Solution {
    public int solution(int a, int b, int c, int d) {
```

```
int[] dice = {a, b, c, d};
Map<Integer, Integer> count = new HashMap<>();

// □□ □□
for (int num : dice) {
    count.put(num, count.getOrDefault(num, 0) + 1);
}

int size = count.size();
List<Integer> keys = new ArrayList<>(count.keySet());

if (size == 1) {
    // case 1: □ □ □□ □□ □
    int p = keys.get(0);
    return 1111 * p;

} else if (size == 2) {
    Collection<Integer> freqs = count.values();
    if (freqs.contains(3)) {
        // case 2: (p, p, p, q)
        int p = 0, q = 0;
        for (int key : keys) {
            if (count.get(key) == 3) p = key;
            else q = key;
        }
        return (int) Math.pow(10 * p + q, 2);
    } else {
        // case 3: (p, p, q, q)
        int p = keys.get(0);
        int q = keys.get(1);
        return (p + q) * Math.abs(p - q);
    }

} else if (size == 3) {
    // case 4: (p, p, q, r)
    int result = 1;
    for (int key : keys) {
        if (count.get(key) == 1) result *= key;
    }
    return result;
```

```
        } else {
            // case 5: 모두 다름
            return Arrays.stream(dice).min().getAsInt();
        }
    }
}
```

Map 쓰는 이유?

문제에서 "같은 숫자가 몇 개 나왔는지"를 기준으로 조건이 분기됨

문제 조건 다시 보면:

- 숫자가 4개 모두 같을 때
- 숫자가 3개 같고 1개 다를 때
- 숫자가 2개, 2개 같을 때
- 숫자가 2개 같고 나머지 각각 다를 때
- 숫자 4개가 모두 다를 때

즉, "각 숫자가 몇 번 나왔는지"를 기준으로 판단해야 한다.
이 말은 곧 "숫자 → 빈도 수"를 빠르게 셀 수 있는 구조가 필요하다는 뜻이다.

그래서 Map<Integer, Integer> 구조가 적절하다. Map은 키(숫자)마다 몇 번 나왔는지(빈도)를 저장할 수 있어서 조건 분기문(예: if (count.size() == 2) 등)을 짜기 매우 직관적이다.

물론 이 문제에서는 숫자가 1~6 범위로 작고, 고정이므로 빈도 배열 int[7] freq도 가능하다.

예를 들어:

```
int[] freq = new int[7];
freq[a]++;
freq[b]++;
freq[c]++;
freq[d]++;
```

→ 숫자가 1~6로 제한되어 있다는 걸 사전에 알 때만 배열이 가능하고, 그 외 범위에서는 Map이 더 일반적이고 확장성 있다.

언제 Map을 써야 하는가?

“값 → 등장 횟수” 구조가 필요할 때 `Map`을 쓴다. 즉, Map은 "무엇이 얼마나 나왔는가"라는 질문에 답해야 할 때 가장 먼저 떠올려야 할 자료구조이다. 특히 문제에서 값의 개수, 빈도, 중복 여부가 중요한 역할을 한다면, Map은 거의 정답에 가깝다고 보면 된다.

- 값이 몇 번 나왔는지 셀 때
  예: 숫자 3번 등장 여부 확인
- 중복된 값 개수로 조건을 나눌 때
  예: 주사위 2개씩 같은 경우 등
- 문자나 숫자의 빈도 계산이 필요할 때
  예: 문자열 문자 수 세기

> 프로그래머스에서 좋아요 제일 많이 받은 코드

??? , flao1 , plotojjy@gmail.com , sksmsgla2@gmail.com ? 146 ?

```
import java.util.Arrays;

class Solution {
    public int solution(int a, int b, int c, int d) {

        int[] dice = { a, b, c, d };
        Arrays.sort(dice);

        int ans = 0;

        if (dice[0] == dice[3]) {
            ans = 1111 * dice[3];
        } else if (dice[0] == dice[2] || dice[1] == dice[3]) {
            ans = (int) Math.pow(dice[1] * 10 + (dice[0] + dice[3] - dice[1]), 2);
        } else if (dice[0] == dice[1] && dice[2] == dice[3]) {
            ans = (dice[0] + dice[3]) * (dice[3] - dice[0]);
        } else if (dice[0] == dice[1]) {
            ans = dice[2] * dice[3];
        } else if (dice[1] == dice[2]) {
            ans = dice[0] * dice[3];
        } else if (dice[2] == dice[3]) {
            ans = dice[0] * dice[1];
        } else {
            ans = dice[0];
        }
```

```
        return ans;
    }
}
```

Revision #12
Created 10 July 2025 08:12:00 by Dain
Updated 13 July 2025 14:28:51 by Dain