브루트 포스 - 기타

최백준 choi@startlink.io

투 포인터

수들의 합 2

https://www.acmicpc.net/problem/2003

- N개의 수로 된 수열 A[1], A[2], …, A[N] 이 있다
- 이 수열의 i번째 수부터 j번째 수까지의 합 A[i]+A[i+1]+…+A[j-1]+A[j]가 M이 되는 경우의 수를 구하는 문제

수들의 합 2

https://www.acmicpc.net/problem/2003

- 이 문제를 풀 수 있는 총 3가지 시간복잡도로 해결할 수 있다.

수들의 합 2

https://www.acmicpc.net/problem/2003

- A[i]+A[i+1]+⋯+A[j-1]+A[j] == M 이 되는 (i, j) 쌍의 개수를 찾는 문제와 같다.
- i를 정하고, j를 정하고, 합을 계산하면 $O(N^3)$로 계산할 수 있다.

```
for (int i=0; i<n; i++) {
    for (int j=i; j<n; j++) {
        int sum = 0;
        for (int k=i; k<=j; k++) {
            sum += a[k];
        }
        if (sum == m) ans += 1;
    }
}
```

수들의 합 2

https://www.acmicpc.net/problem/2003

- i = a, j = b인 경우에 합을 구한 다음 과정은
- i = a, j = b+1의 합을 구하는 과정이다.
- 그런데, A[a] + A[a+1] + ⋯ + A[b]와 A[a] + A[a+1] + ⋯ + A[b] + A[b+1]의 차이는 A[b+1]밖에 없다.
- 합은 변하지 않는데 여러 번 구하는 것은 중복된 연산으로 없앨 수 있다.

수들의 합 2

https://www.acmicpc.net/problem/2003

- A[i]+A[i+1]+⋯+A[j-1]+A[j] == M 이 되는 (i, j) 쌍의 개수를 찾는 문제와 같다.
- 합을 계산할 때, 합을 각각의 i에 대해서 누적하면 $O(N^2)$로 계산할 수 있다.

```
for (int i=0; i<n; i++) {
    int sum = 0;
    for (int j=i; j<n; j++) {
        sum += a[j];
        if (sum == m) ans += 1;
    }
}
```

수들의 합 2

https://www.acmicpc.net/problem/2003

- i = a, j = b의 합이 M보다 작았고, i = a, j = b+1의 합이 M보다 큰 경우를 생각해보자
- 식으로 나타내면 다음과 같다.
    - A[a] + A[a+1] + … + A[b] < M
    - A[a] + A[a+1] + … + A[b+1] > M
- 이 경우 j를 계속 증가시키는 것은 의미가 없기 때문에, i를 증가시켜야 한다.
- 그런데
- i = a+1이고, a ≤ j ≤ b인 경우에서 합이 M이 되는 경우는 있을 수가 없다.
- A[a+1] + … + A[b] == M 이라면 A[a] + A[a+1] + … + A[b] > M 이기 때문에, 위의 조건에 모순이기 때문이다.
- 따라서, 이런 경우는 i만 1증가시키면 된다.

수들의 합 2

https://www.acmicpc.net/problem/2003

- A[i]+A[i+1]+…+A[j-1]+A[j] == M 이 되는 (i, j) 쌍의 개수를 찾는 문제와 같다.
- 합을 계산할 때, 합을 각각의 i에 대해서 누적하면 $O(N^2)$로 계산할 수 있다.

```
for (int i=0; i<n; i++) {
    int sum = 0;
    for (int j=i; j<n; j++) {
        sum += a[j];
        if (sum == m) ans += 1;
    }
}
```

수들의 합 2

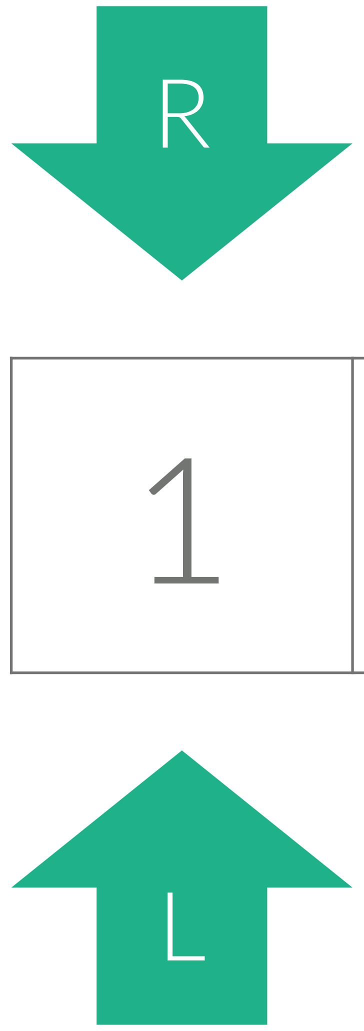
https://www.acmicpc.net/problem/2003

- 뒷 페이지의 설명에서 i는 L(왼쪽)로, j는 R(오른쪽)으로 표현했다.

수들의 합 2

https://www.acmicpc.net/problem/2003

- 찾으려고 하는 수: 5
- 합: 1

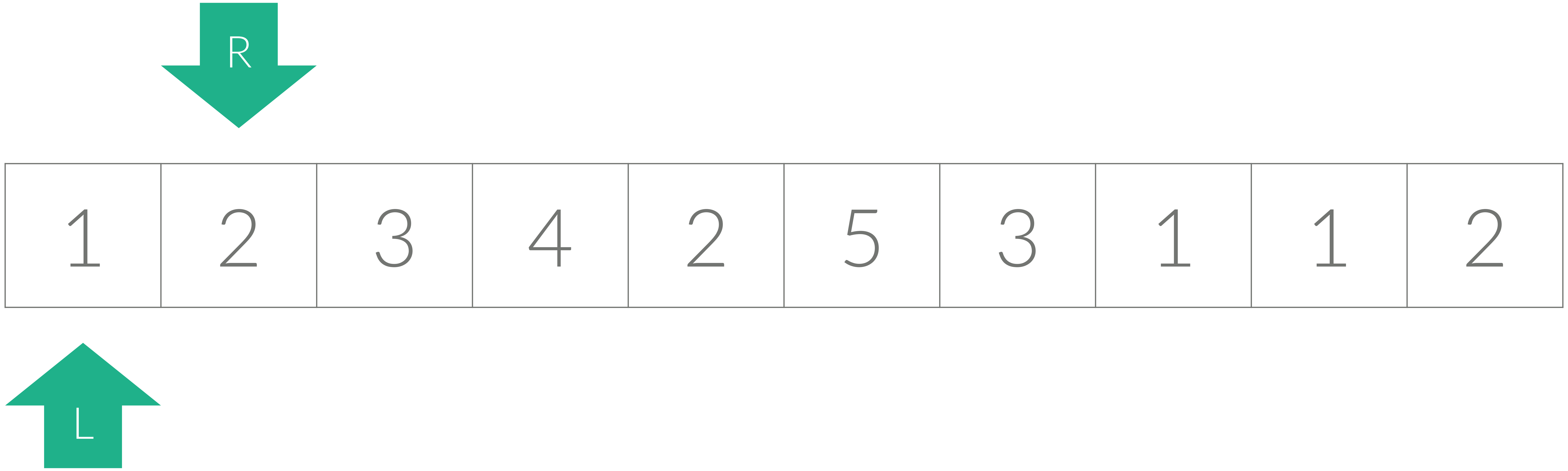


1	2	3	4	2	5	3	1	1	2

수들의 합 2

https://www.acmicpc.net/problem/2003

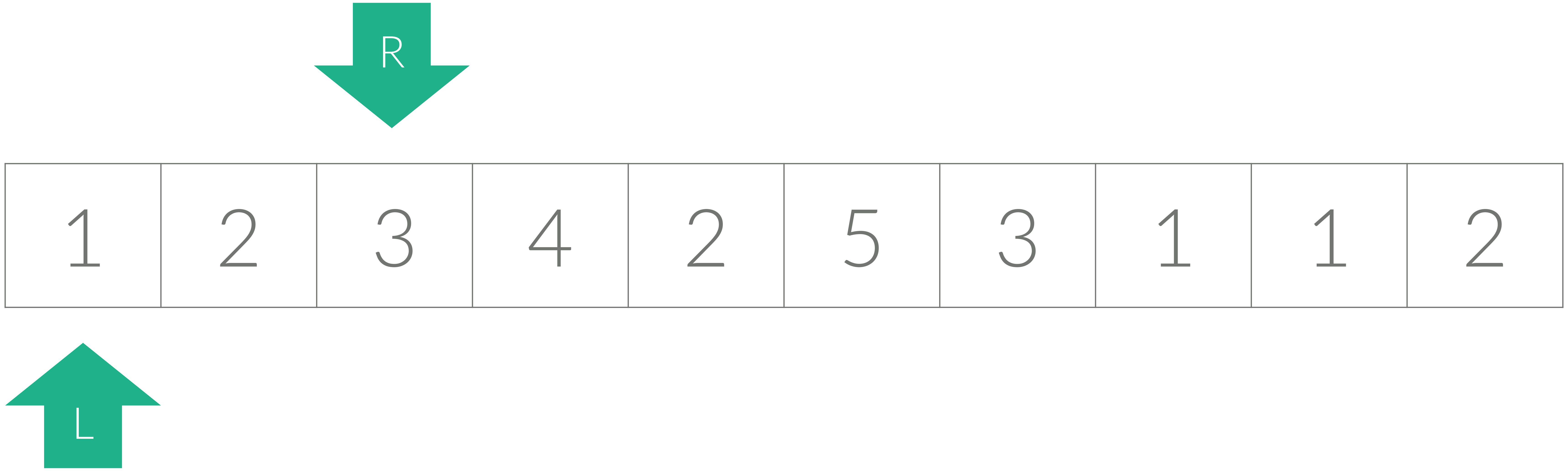
- 찾으려고 하는 수: 5
- 합: 3



수들의 합 2

https://www.acmicpc.net/problem/2003

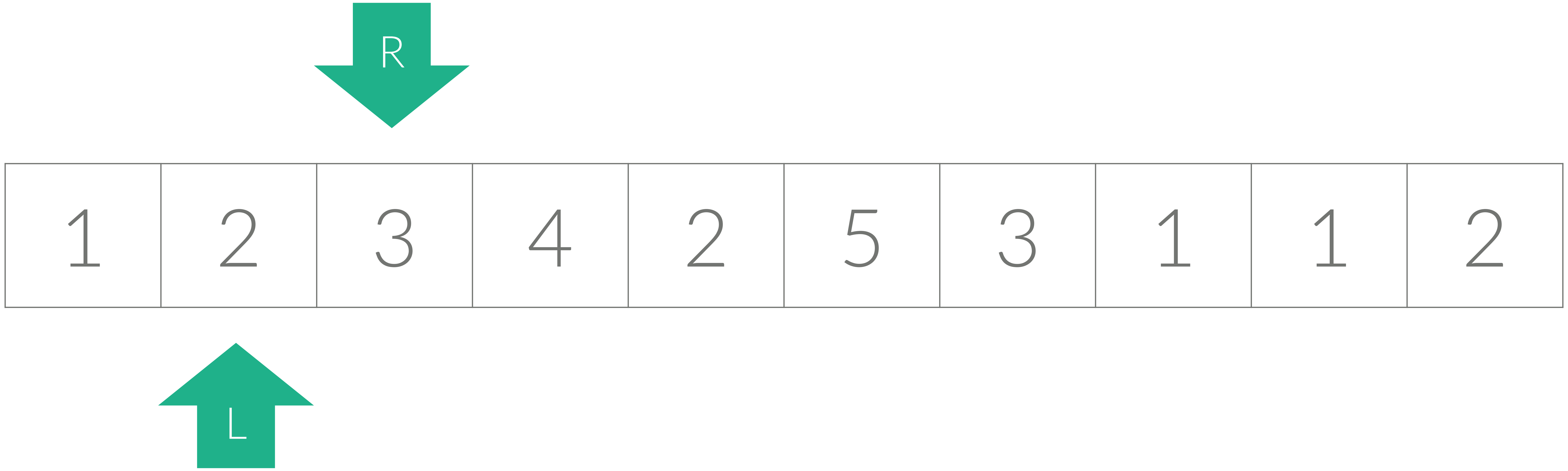
- 찾으려고 하는 수: 5
- 합: 6



수들의 합 2

https://www.acmicpc.net/problem/2003

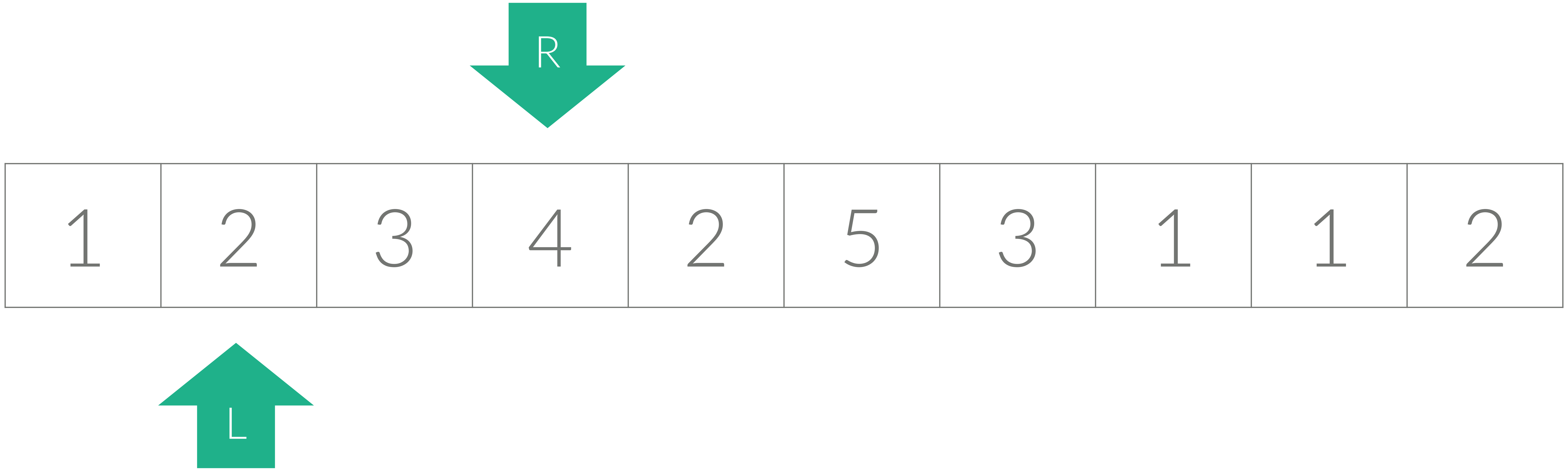
- 찾으려고 하는 수: 5
- 합: 5 (찾았다!)



수들의 합 2

https://www.acmicpc.net/problem/2003

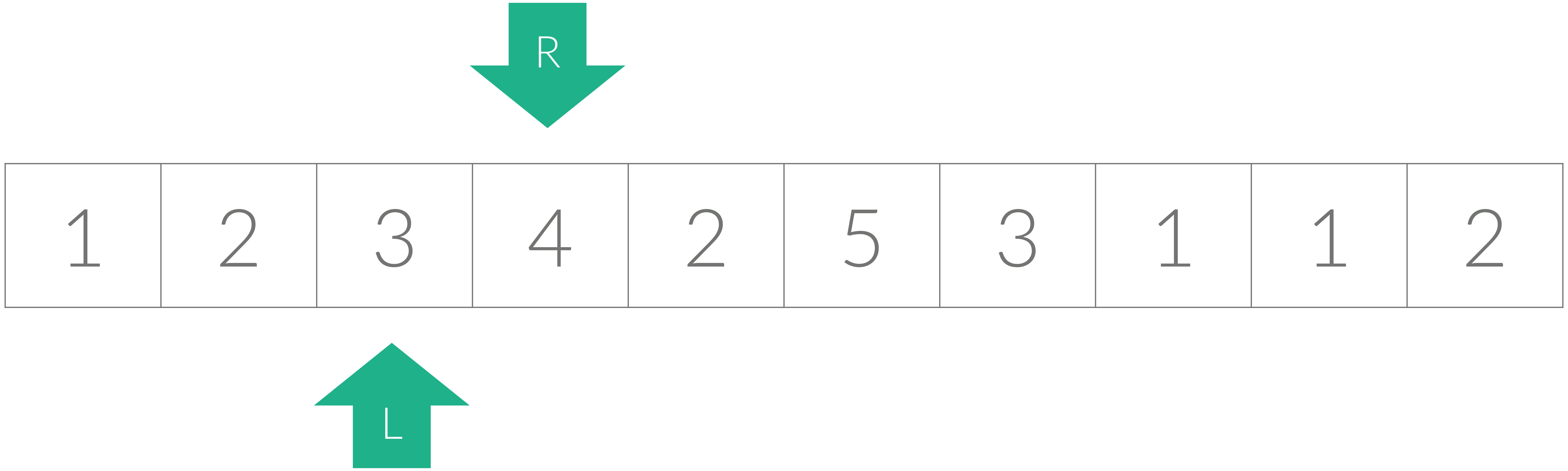
- 찾으려고 하는 수: 5
- 합: 9



수들의 합 2

https://www.acmicpc.net/problem/2003

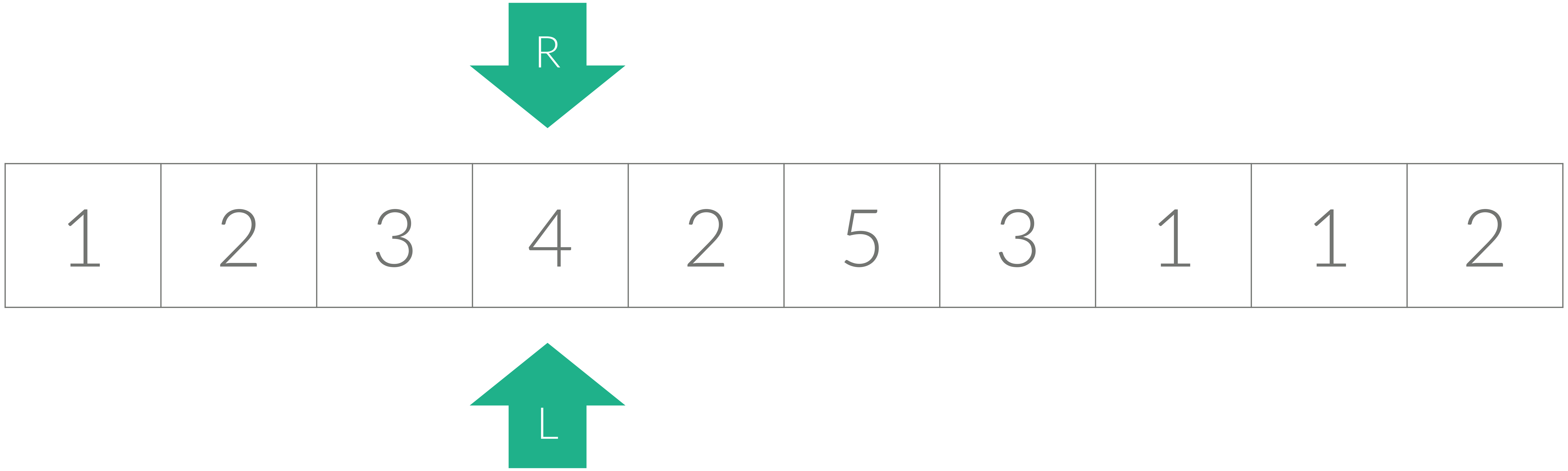
- 찾으려고 하는 수: 5
- 합: 7



수들의 합 2

https://www.acmicpc.net/problem/2003

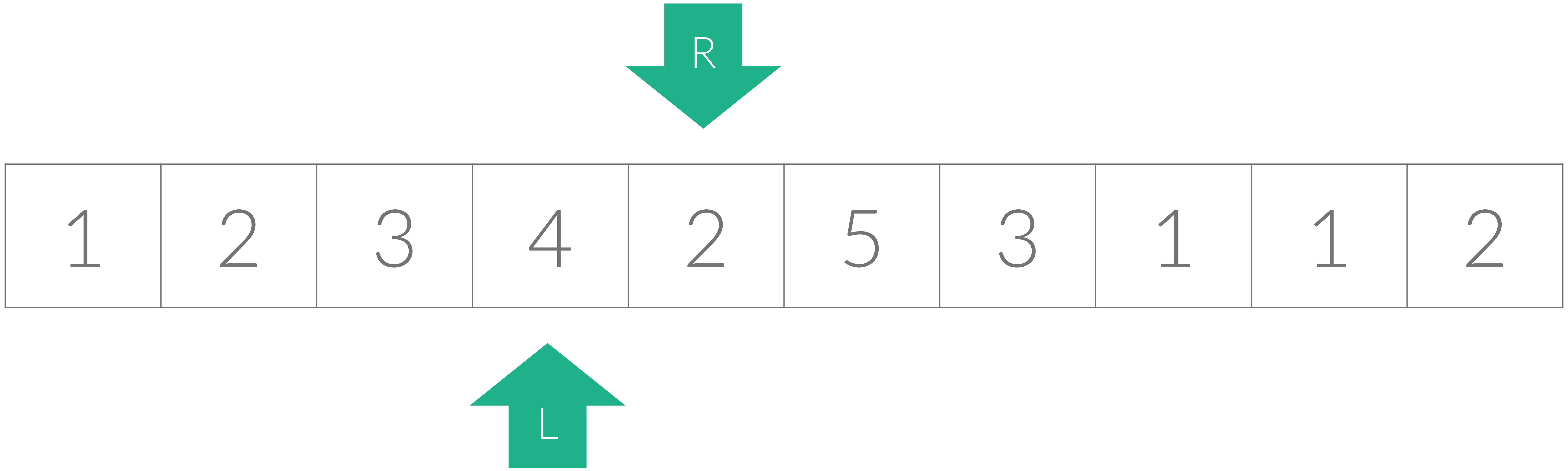
- 찾으려고 하는 수: 5
- 합: 4



수들의 합 2

https://www.acmicpc.net/problem/2003

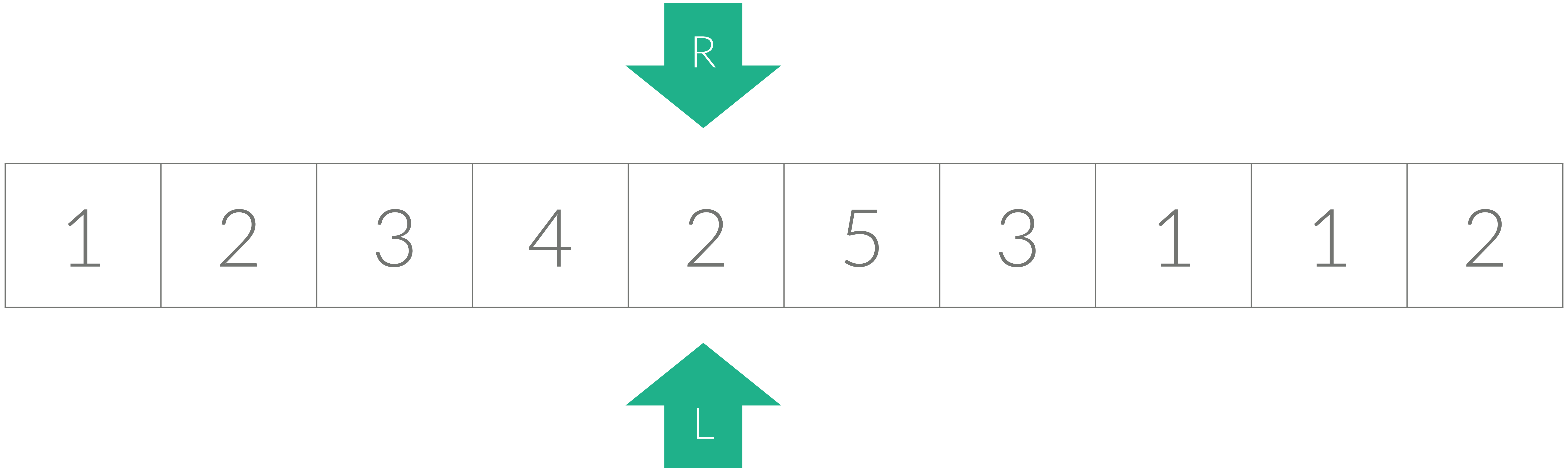
- 찾으려고 하는 수: 5
- 합: 6



수들의 합 2

https://www.acmicpc.net/problem/2003

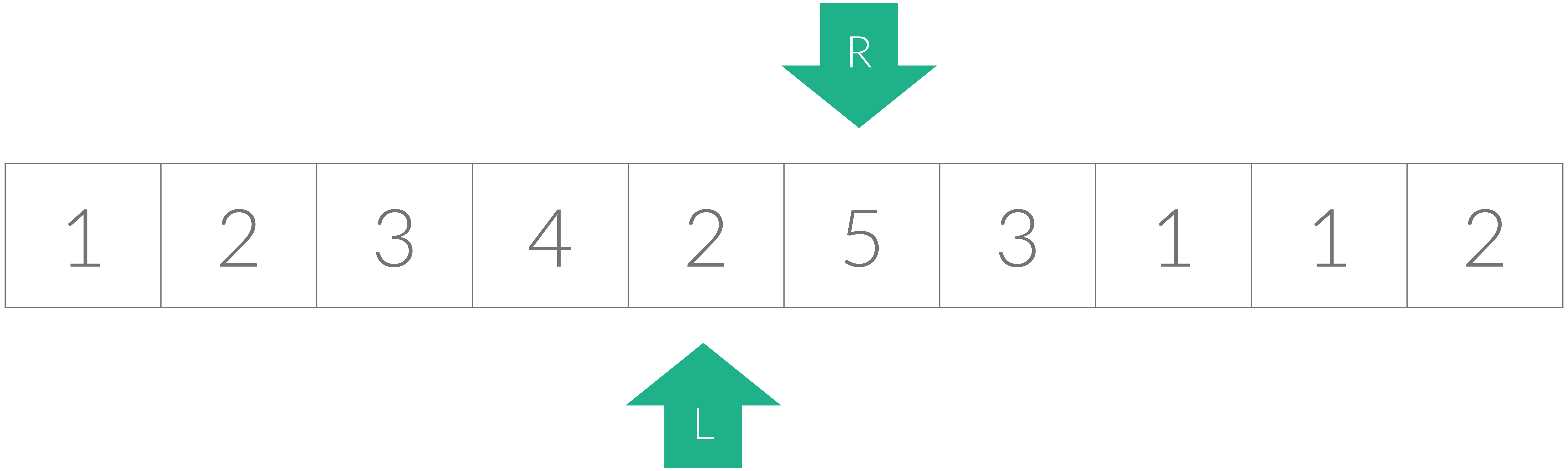
- 찾으려고 하는 수: 5
- 합: 2



수들의 합 2

https://www.acmicpc.net/problem/2003

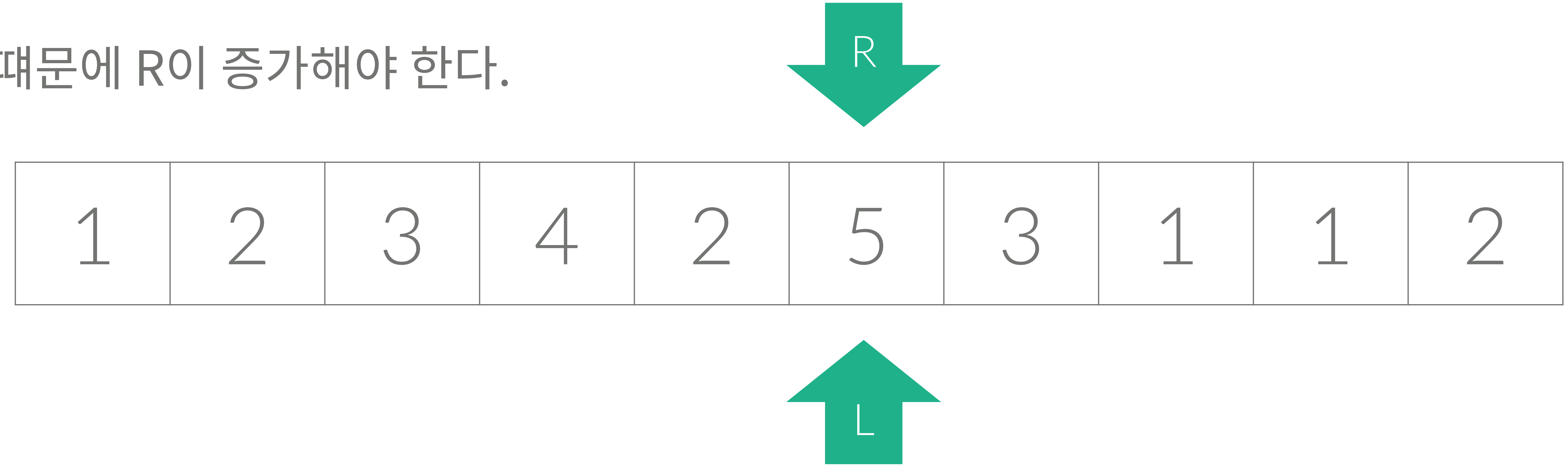
- 찾으려고 하는 수: 5
- 합: 7



수들의 합 2

https://www.acmicpc.net/problem/2003

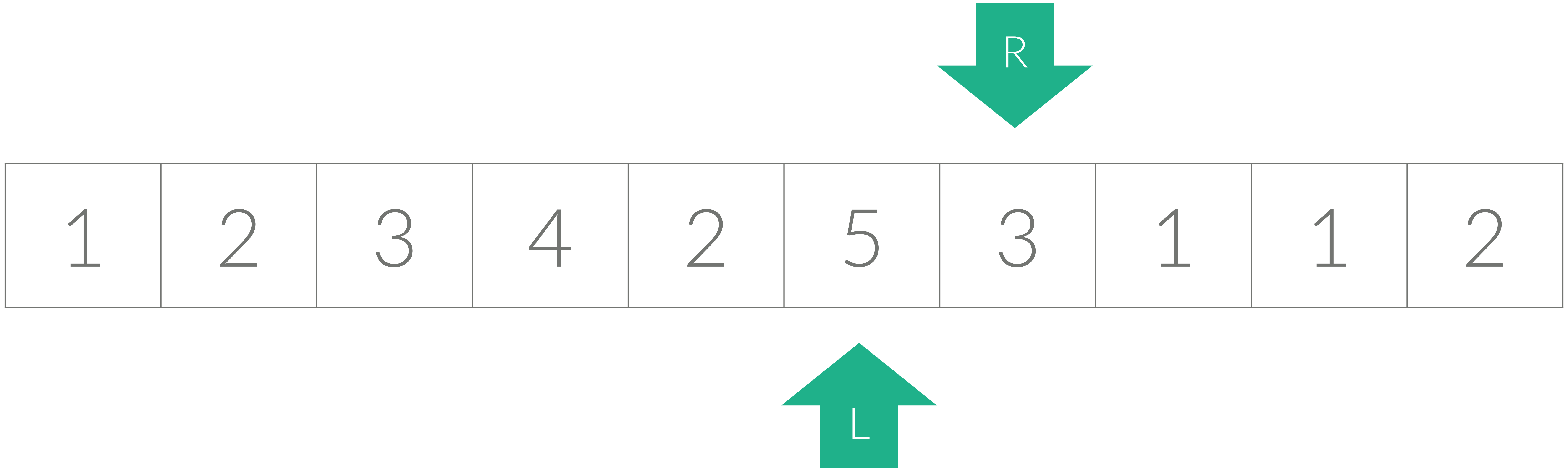
- 찾으려고 하는 수: 5
- 합: 5 (찾았다!)
- 같은 경우에는 L, R 둘 중에 아무거나 증가해도 상관없지만
- 이런 경우 때문에 R이 증가해야 한다.



수들의 합 2

https://www.acmicpc.net/problem/2003

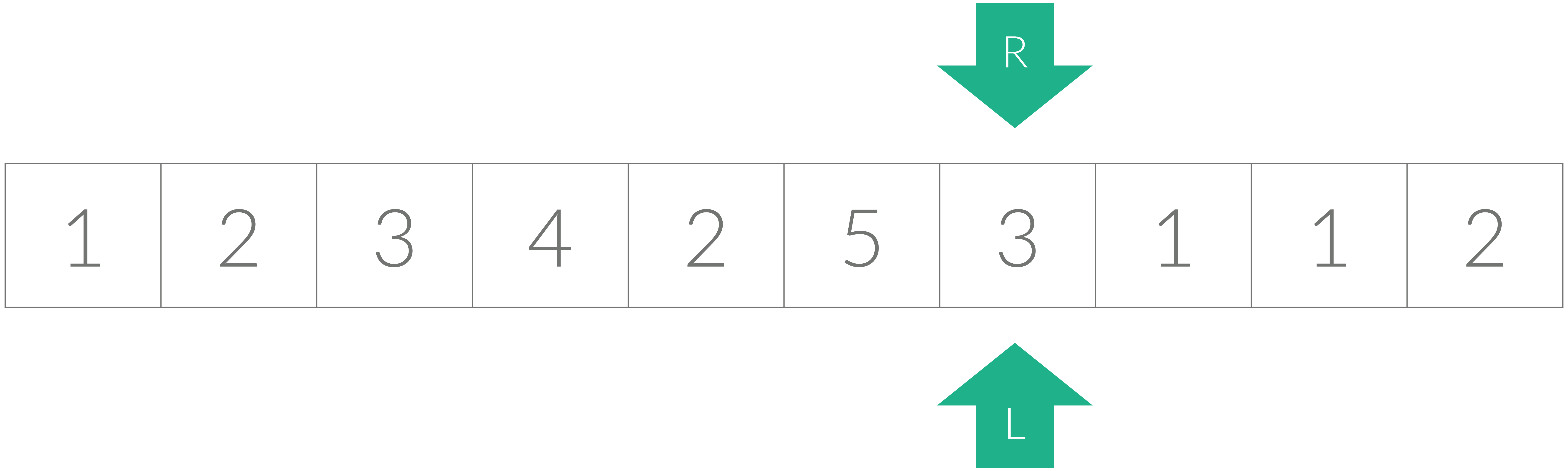
- 찾으려고 하는 수: 5
- 합: 8



수들의 합 2

https://www.acmicpc.net/problem/2003

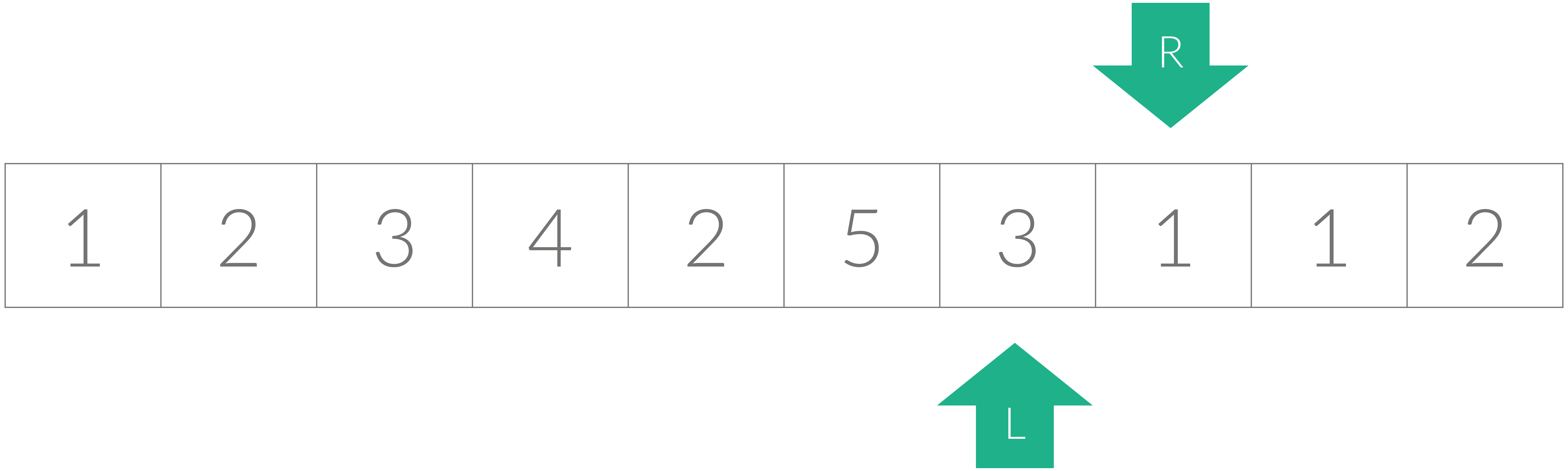
- 찾으려고 하는 수: 5
- 합: 3



수들의 합 2

https://www.acmicpc.net/problem/2003

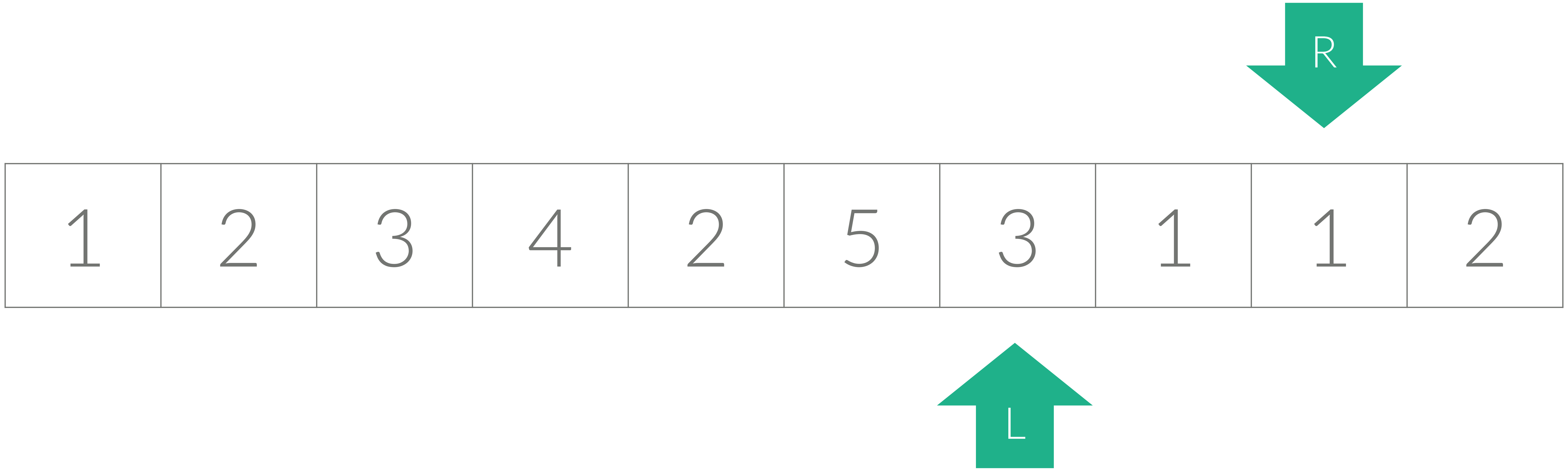
- 찾으려고 하는 수: 5
- 합: 4



수들의 합 2

https://www.acmicpc.net/problem/2003

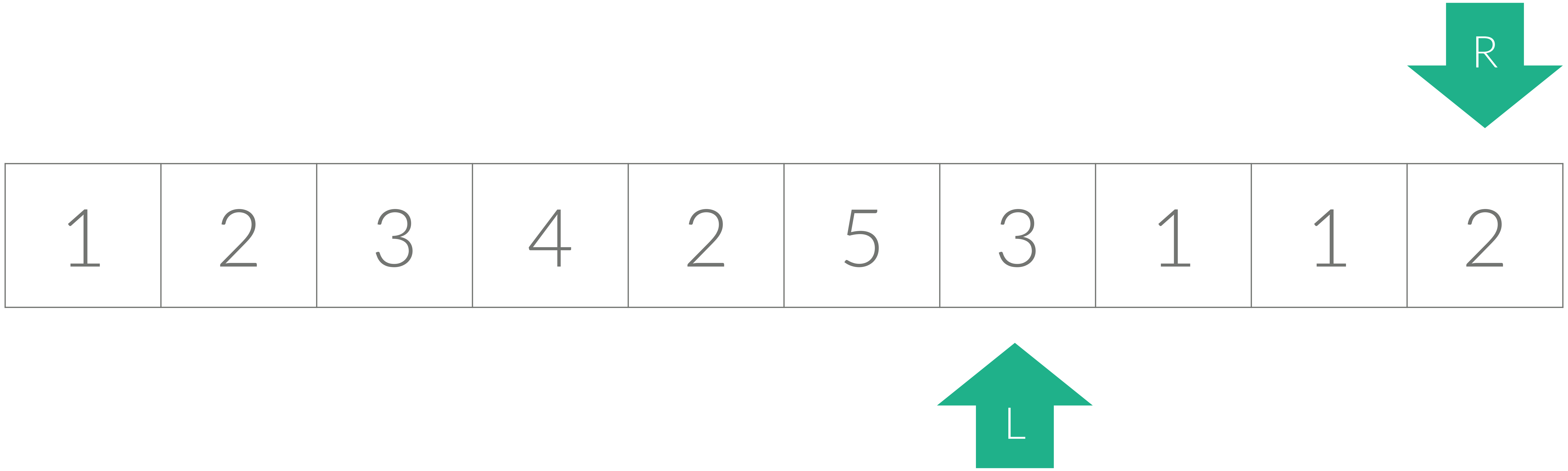
- 찾으려고 하는 수: 5
- 합: 5 (찾았다!)



수들의 합 2

https://www.acmicpc.net/problem/2003

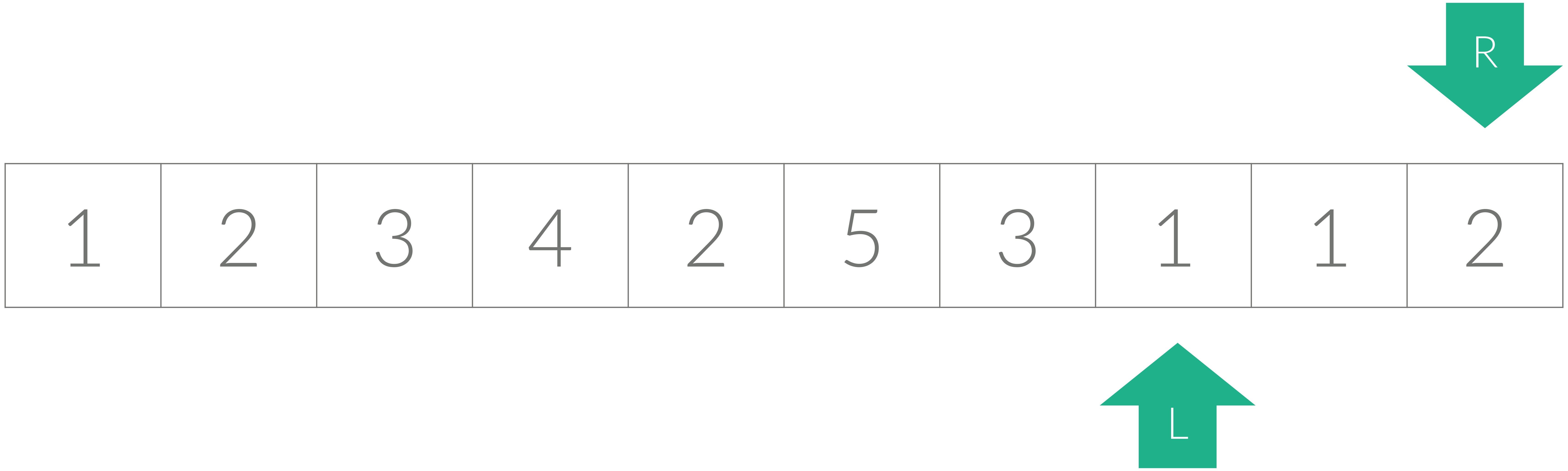
- 찾으려고 하는 수: 5
- 합: 7



수들의 합 2

https://www.acmicpc.net/problem/2003

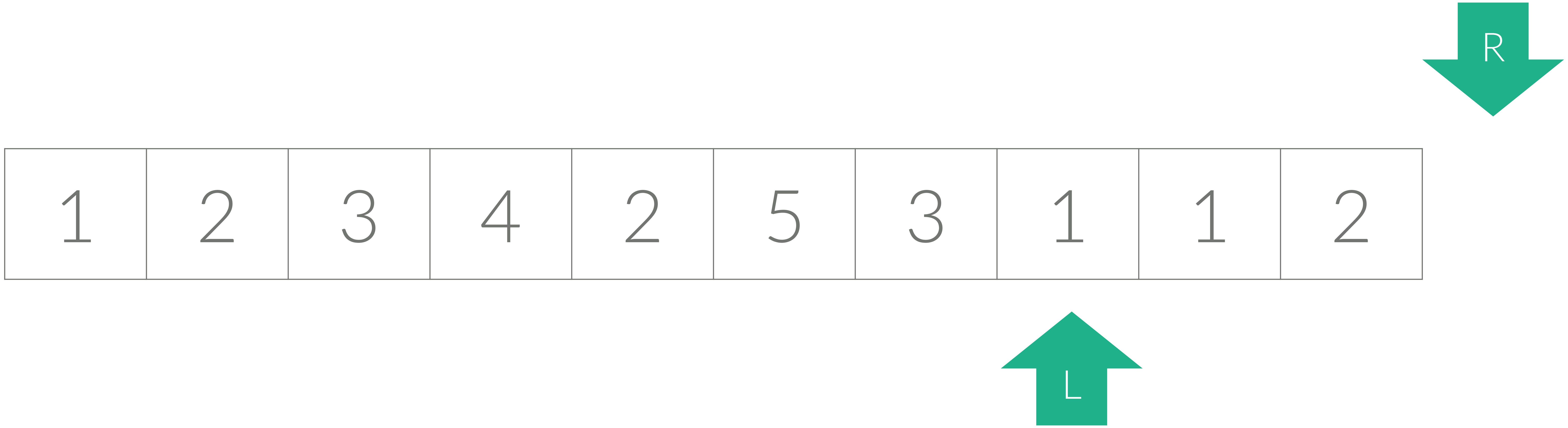
- 찾으려고 하는 수: 5
- 합: 4



수들의 합 2

https://www.acmicpc.net/problem/2003

- 찾으려고 하는 수: 5
- 끝



수들의 합 2

https://www.acmicpc.net/problem/2003

```
int left=0, right=0, sum=a[0], ans = 0;
while (left <= right && right < n) {
    if (sum < m) {
        right += 1;
        sum += a[right];
    } else if (sum == m) {
        ans += 1;
        right += 1;
        sum += a[right];
    } else if (sum > m) {
        sum -= a[left];
        left++;
    }
}
```

수들의 합 2

https://www.acmicpc.net/problem/2003

- 총 시간 복잡도는 L과 R이 $L \leq R$을 유지하면서 끝까지 가기 때문에, $O(N) + O(N) = O(N)$이다.

수들의 합 2

https://www.acmicpc.net/problem/2003

- 소스: http://codeplus.codes/5df1df7ff20e4496b849d8f00bf2f4ab

부분합

https://www.acmicpc.net/problem/1806

- 구간 합 중에서 합이 S이상인 것 중에서 가장 짧은 것을 구하는 문제

부분합

https://www.acmicpc.net/problem/1806

- 소스: http://codeplus.codes/9a10be6e12f04c62ac8268393a620b00

소수의 연속합

https://www.acmicpc.net/problem/1644

- 수들의 합 2 문제와 같지만, 소수를 구해서 답을 구해야 하는 문제

소수의 연속합

https://www.acmicpc.net/problem/1644

- 소스: http://codeplus.codes/09a52b93d1fa4355b0c64e868b7461e5

중간에서 만나기

중간에서 만나기

Meet in the Middle

- 문제를 절반으로 나눠서
- 양쪽 절반에서 모든 경우를 다 해보는 방법이다.
- 탐색의 크기가 많이 줄어든다.
- 문제의 크기가 N인 경우에 2^N 에서
- M = N/2 라고 했을 때, 2^M + 2^M 으로 줄어들게 된다.

부분수열의 합 2

https://www.acmicpc.net/problem/1208

- 서로 다른 N개의 정수로 이루어진 수열이 있을 때, 크기가 양수인 부분수열 중에서 그 수열의 원소를 다 더한 값이 S가 되는 경우의 수를 구하는 문제
- $1 \le N \le 40$

부분수열의 합 2

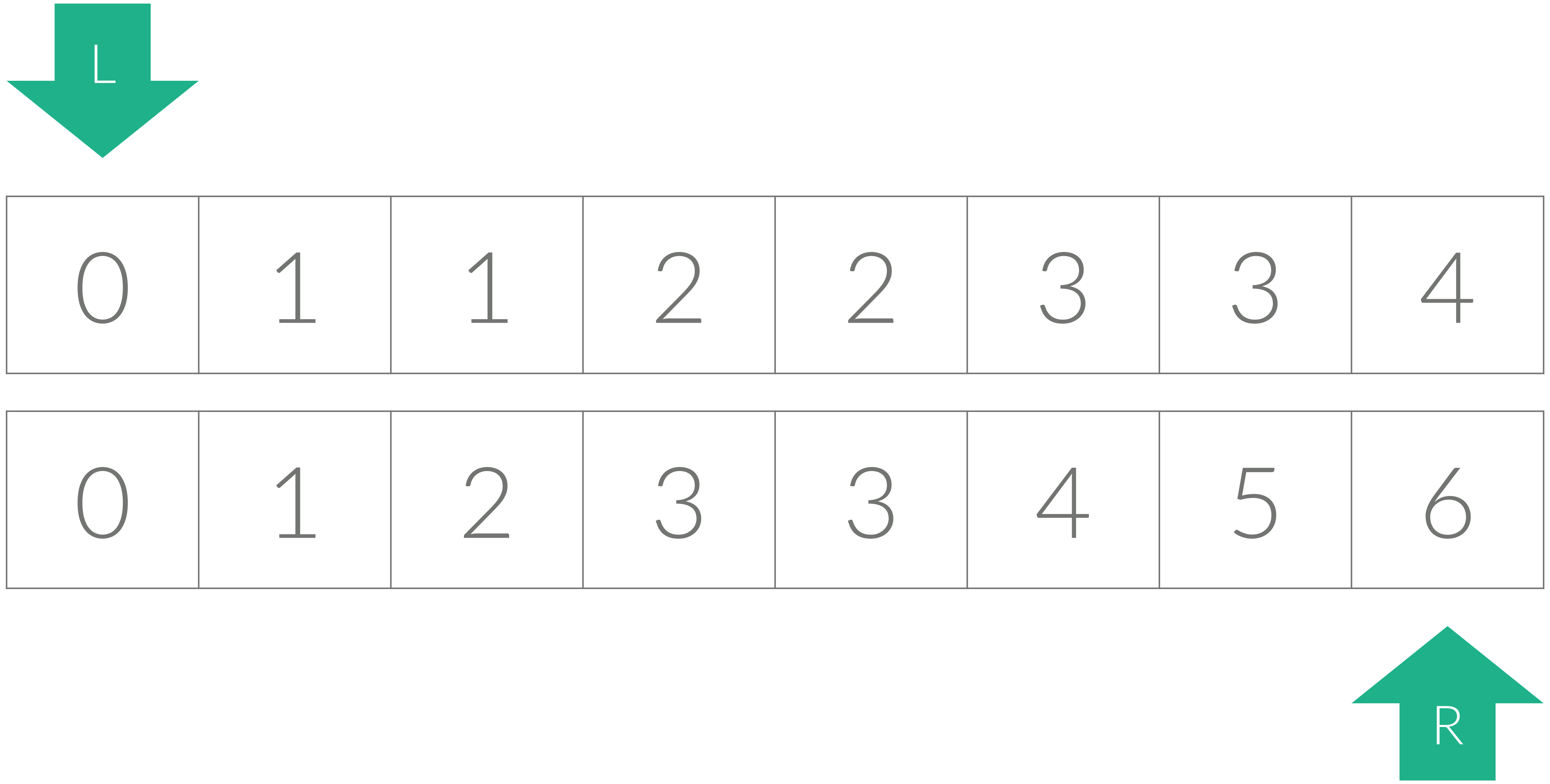
https://www.acmicpc.net/problem/1208

- 부분수열의 합 문제와 비슷하지만, 배열을 2개 써야하는 문제
- A = [1, 2, 1, 3, 2, 1], M = 4 인 경우를 생각해보자
- A를 절반으로 나누어서
- Up = [1, 2, 1]
- Down = [3, 2, 1]
- 에 대해서 각각 모든 경우를 나열한다.

부분수열의 합 2

https://www.acmicpc.net/problem/1208

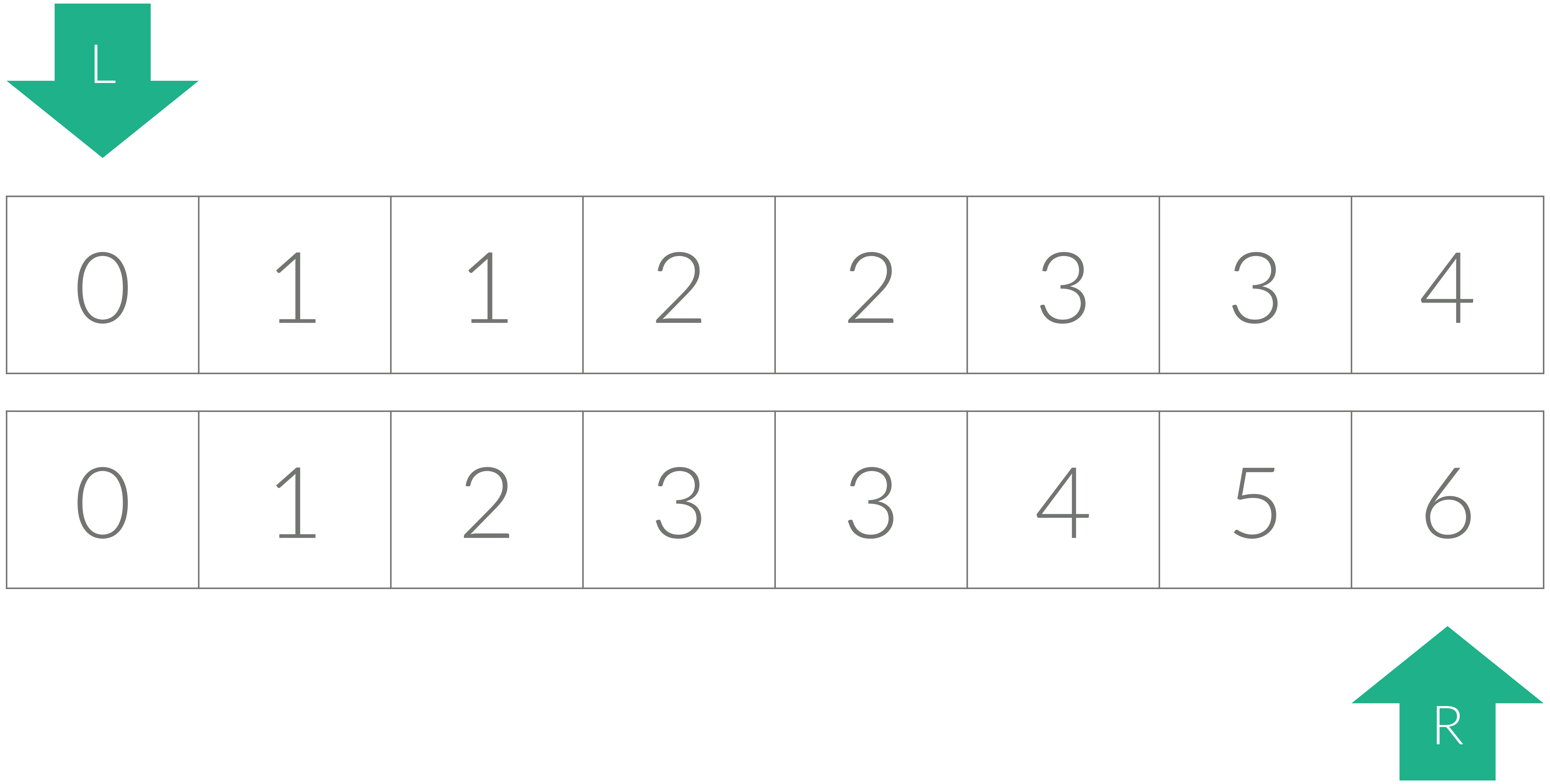
- Up = [1, 2, 1]
- Down = [1, 2, 3]
- 에 대해서 각각 모든 경우를 나열한다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

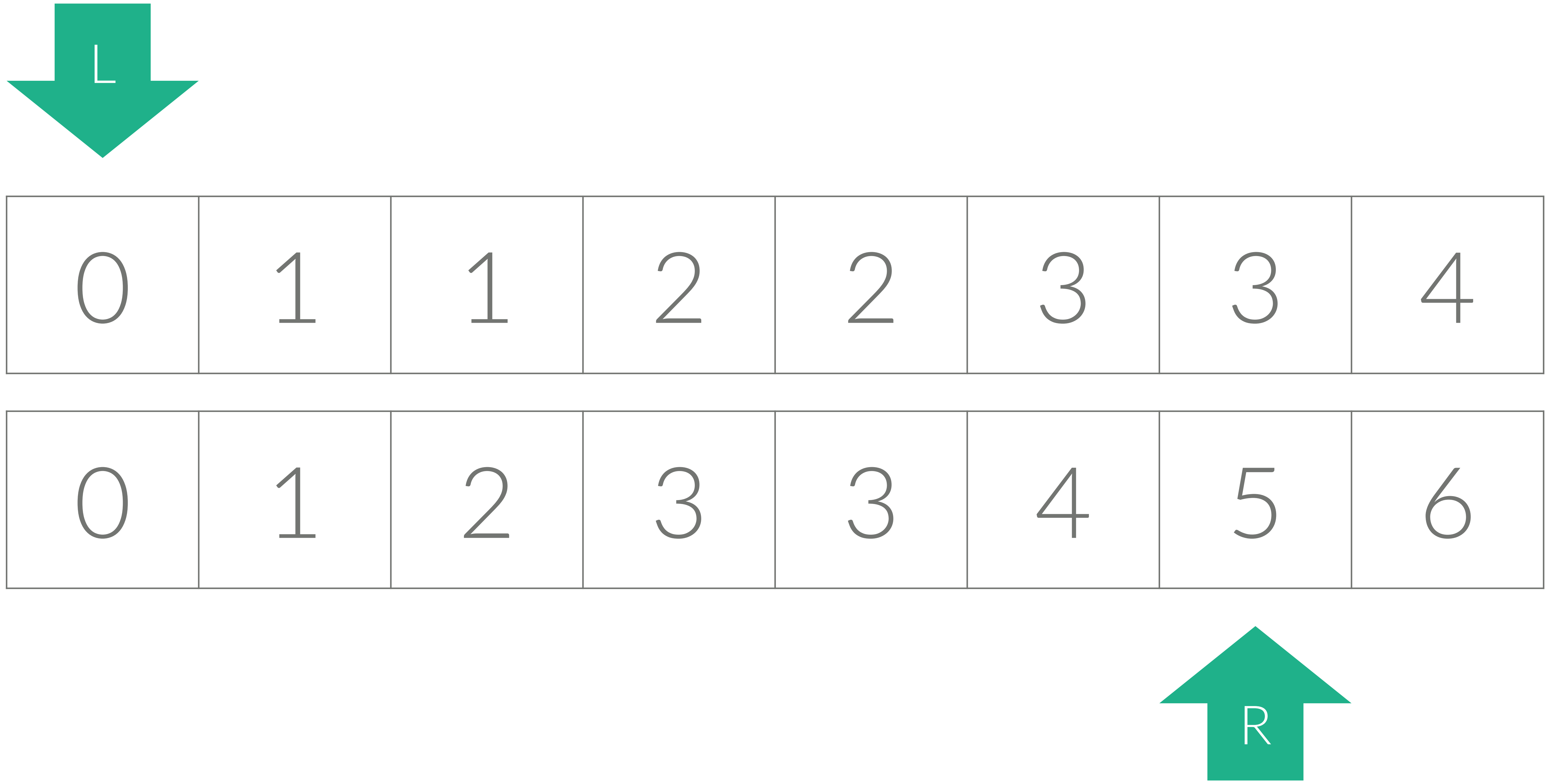
- 0 + 6 = 6이고, 찾으려고 하는 수 4보다 크기 때문에, R을 1칸 당긴다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

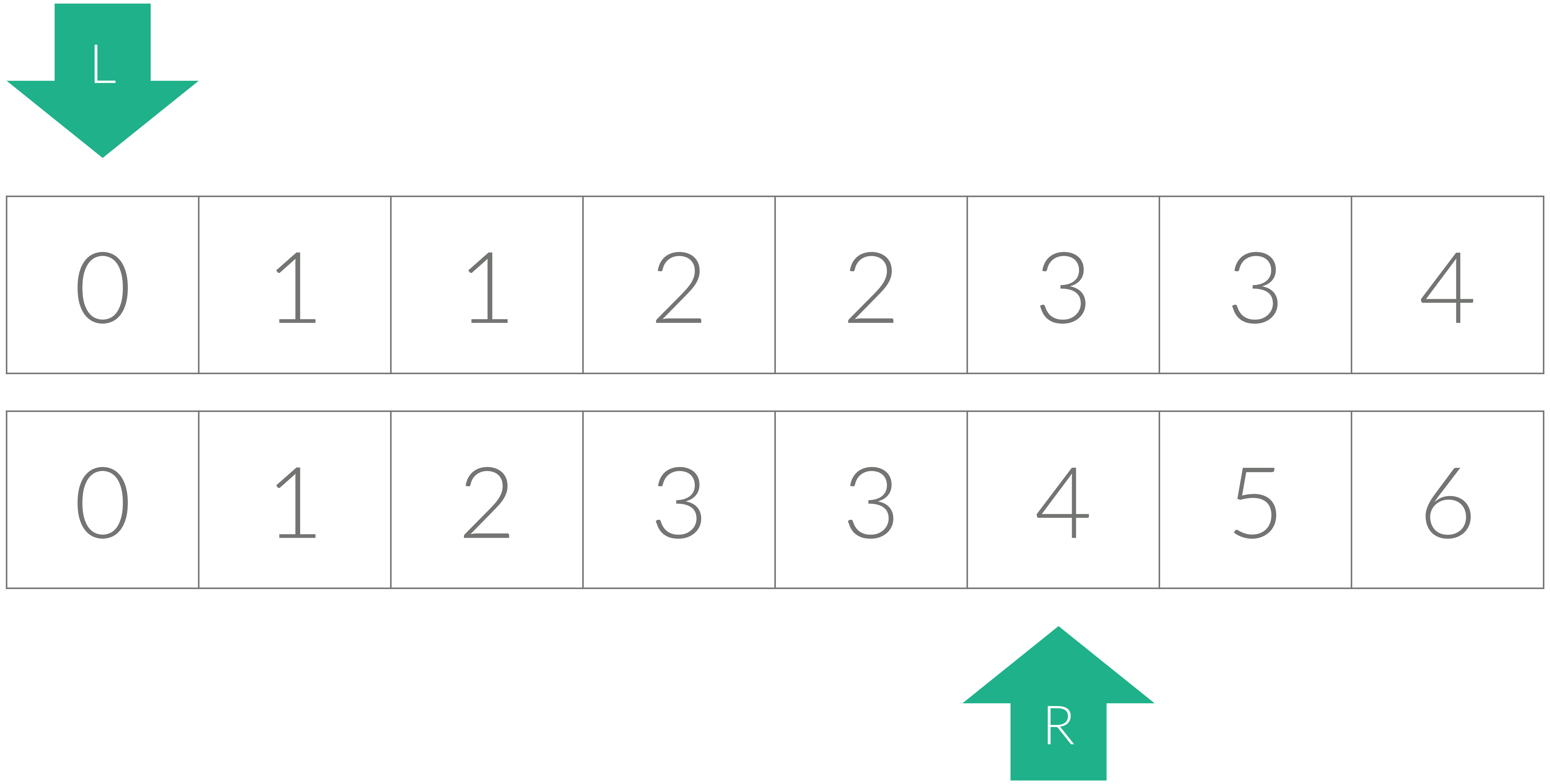
- 0 + 5 = 5이고, 찾으려고 하는 수 4보다 크기 때문에, R을 1칸 당긴다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

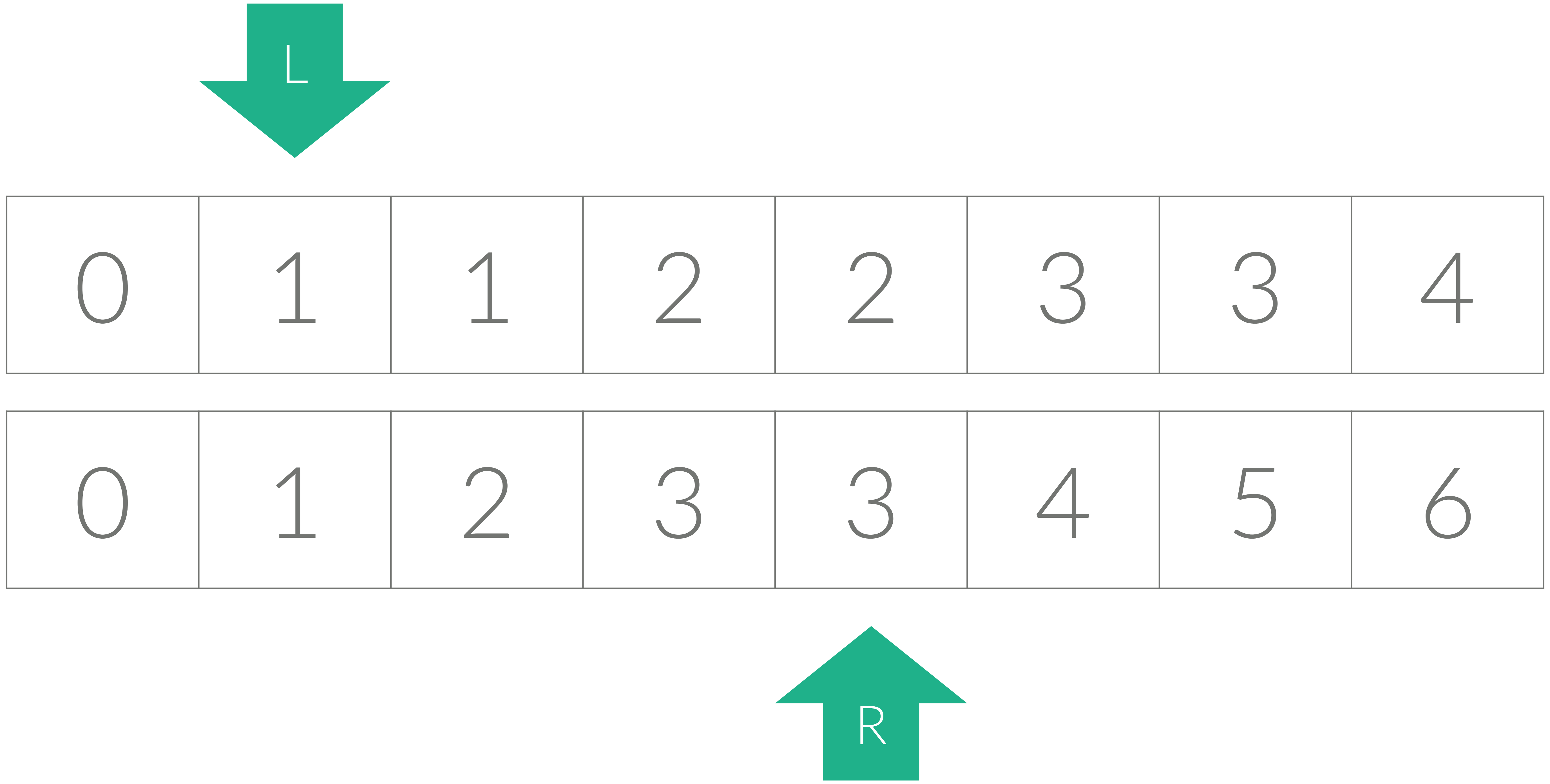
- 0 + 4 = 4이고, 찾으려고 하는 수 4 이다.
- 위에 0이 1개, 아래에 4가 1개 있기 때문에, 4는 총 1×1 = 1개이다.
- 이제, L과 R을 이동시킨다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

- 1 + 3 = 4이고, 찾으려고 하는 수 4 이다.
- 위에 1이 2개, 아래에 3가 2개 있기 때문에, 4는 총 2×2 = 4개이다.
- 이제, L과 R을 이동시킨다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

- 2 + 2 = 4이고, 찾으려고 하는 수 4 이다.
- 위에 2가 2개, 아래에 2가 1개 있기 때문에, 4는 총 2×1 = 2개이다.
- 이제, L과 R을 이동시킨다.

L

0	1	1	2	2	3	3	4

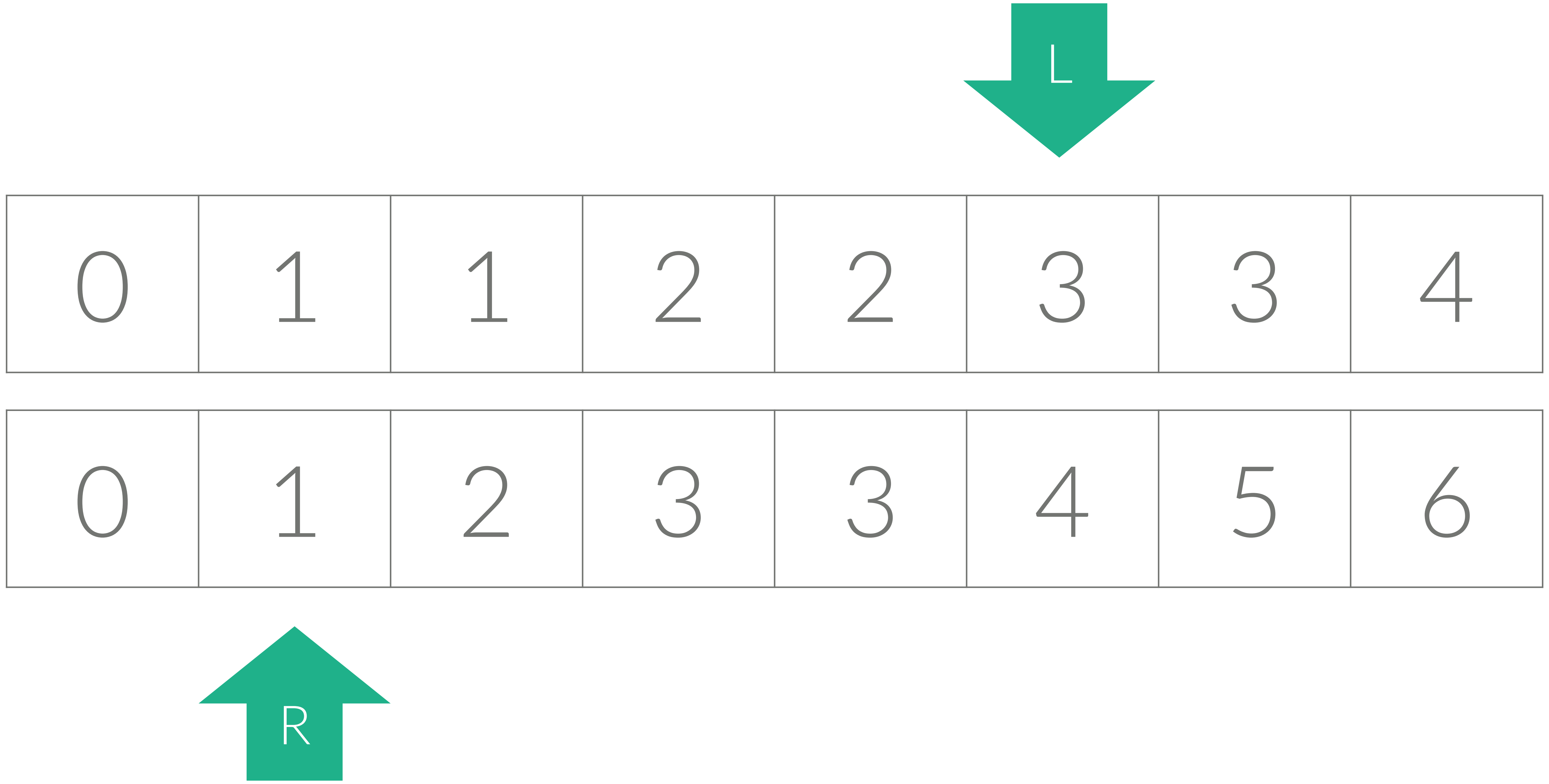
0	1	2	3	3	4	5	6

R

부분수열의 합 2

https://www.acmicpc.net/problem/1208

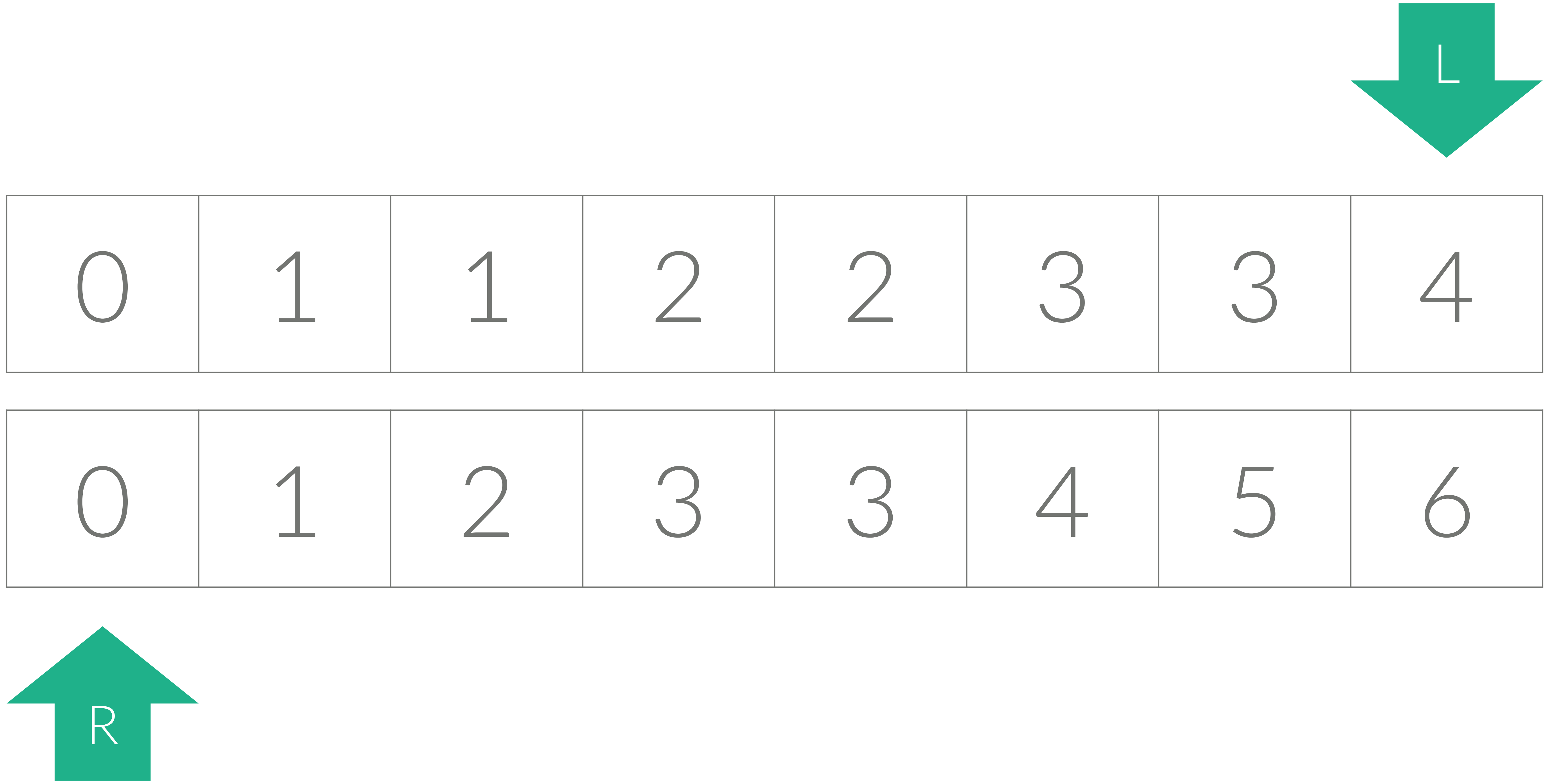
- 3 + 1 = 4이고, 찾으려고 하는 수 4 이다.
- 위에 3이 2개, 아래에 1이 1개 있기 때문에, 4는 총 2×1 = 2개이다.
- 이제, L과 R을 이동시킨다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

- 4 + 0 = 4이고, 찾으려고 하는 수 4 이다.
- 위에 4가 1개, 아래에 0이 1개 있기 때문에, 4는 총 1×1 = 1개이다.
- 이제, L과 R을 이동시킨다.



부분수열의 합 2

https://www.acmicpc.net/problem/1208

- 소스: http://codeplus.codes/79804aa3582a4567b2c2c4109982e535

두 배열의 합

https://www.acmicpc.net/problem/2143

- 배열 A[1], A[2], ..., A[n]의 부 배열은 A[i], A[i+1], ..., A[j] ($1 \le i \le j \le n$) 의 합
- 두 배열 A와 B가 주어졌을 때
- A의 부 배열의 합과 B의 부 배열의 합을 더한 것이 T가 되는 경우의 수를 구하는 문제

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다

L

1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 1 + B = T, B = 4의 개수 (1개)

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 1 + B = T, B = 4의 개수 (1개)

1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 2 + B = T, B = 3의 개수 (1개)

L

1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 3 + B = T, B = 2의 개수 (1개)

L

1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 3 + B = T, B = 2의 개수 (1개)

L

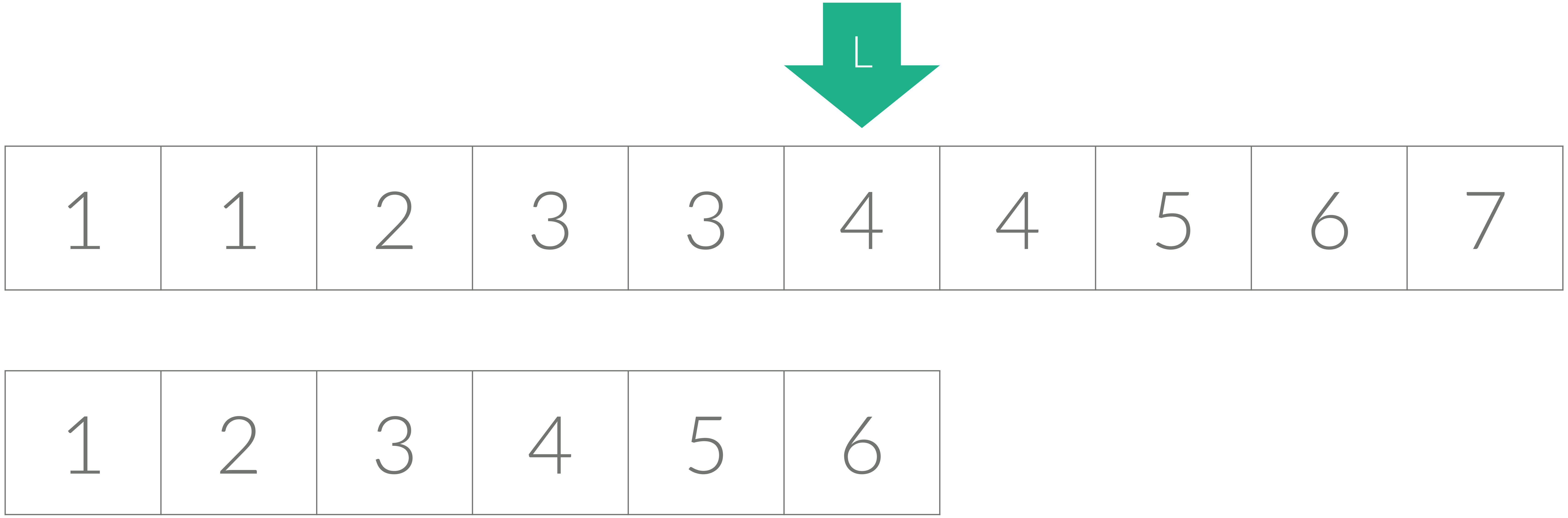
1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 4 + B = T, B = 1의 개수 (1개)



두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 4 + B = T, B = 1의 개수 (1개)

						L			
1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 5 + B = T, B = 0의 개수 (1개)

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 6 + B = T, B = -1의 개수 (0개)

L

1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- T = 5, 각각의 A의 부분 합에 대해서, 해당하는 B의 부분 합의 개수를 세어보는 방법도 있다
- 7 + B = T, B = -2의 개수 (0개)

L

1	1	2	3	3	4	4	5	6	7

1	2	3	4	5	6

두 배열의 합

https://www.acmicpc.net/problem/2143

- 소스: http://codeplus.codes/b91235e28d344d8b9e507fc37cdd3e84

합이 0인 네 정수

https://www.acmicpc.net/problem/7453

- 크기가 N인 배열 A, B, C, D가 있다
- 이 때, A[a], B[b], C[c], D[d]의 합이 0인 (a, b, c, d) 쌍의 개수를 구하는 문제
- $1 \leq N \leq 4{,}000$

합이 0인 네 정수

https://www.acmicpc.net/problem/7453

- 총 가능한 경우의 수: N^4 가지
- A[a] + B[b] + C[c] + D[d] = 0

합이 0인 네 정수

https://www.acmicpc.net/problem/7453

- 총 가능한 경우의 수: N^4 가지
- A[a] + B[b] + C[c] + D[d] = 0
- A[a] + B[b] = -C[c] - D[d]
- A[a] + B[b] = -(C[c] + D[d])

합이 0인 네 정수

https://www.acmicpc.net/problem/7453

- 총 가능한 경우의 수: N^4 가지
- 절반으로 나눠서
- A[a]+B[b] = N^2가지
- C[c]+D[d] = N^2가지
- 계산해볼 수 있다.

합이 0인 네 정수

https://www.acmicpc.net/problem/7453

- 소스: http://codeplus.codes/a35451012def4100a7a4cb79460f9c16