Sqrt Decomposition

sqrt
제곱근 분할법

Algorithm Session
dev.shinkeonkim@gmail.com

김신건
Koa

어디에 쓸 것인가?

어디에 쓸 것인가?

어떤 상황에 쓰이는가?

2042번 : 구간 합 구하기

https://www.acmicpc.net/problem/2042

문제 설명

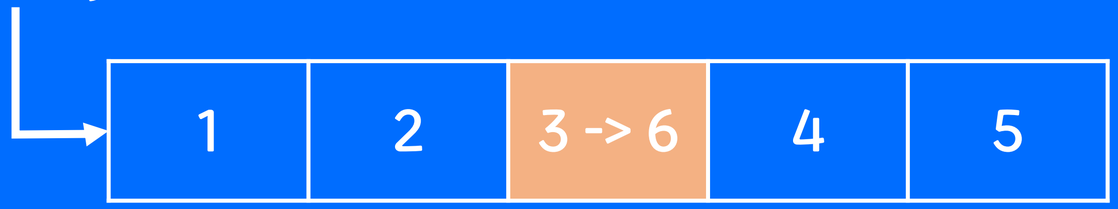
어떤 N개의 수가 주어져 있다.
그런데 중간에 수의 변경이 빈번히 일어나고 그 중간에 어떤 부분의 합을 구하려 한다.

만약에 1,2,3,4,5 라는 수가 있고, 3번째 수를 6으로 바꾸고 2번째부터 5번째까지 합을 구하라고 한다면 17을 출력하면 되는 것이다.
그리고 그 상태에서 다섯 번째 수를 2로 바꾸고 3번째부터 5번째까지 합을 구하라고 한다면 12가 될 것이다.

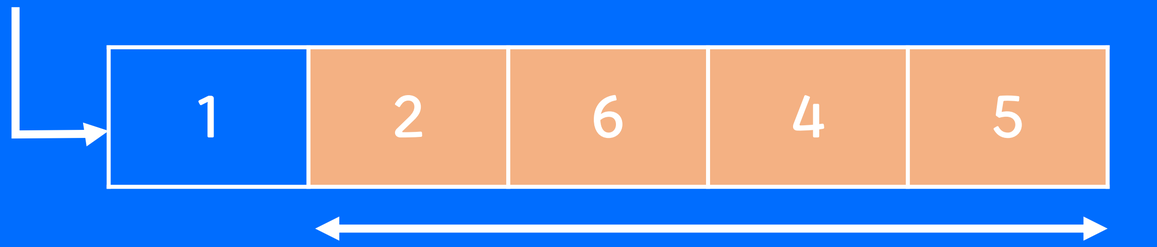
어디에 쓸 것인가?

어떤 상황에 쓰이는가?

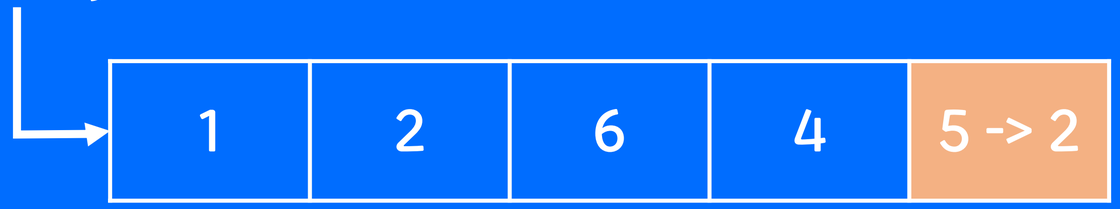
Query (1) : 3번째 수를 6으로 바꿔라!



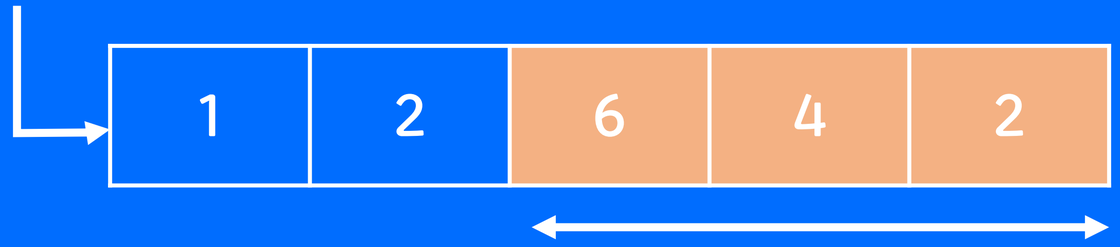
Query (2): 2번째부터 5번째까지의 합을 구해라!



Query (3): 5번째 수를 2로 바꿔라!



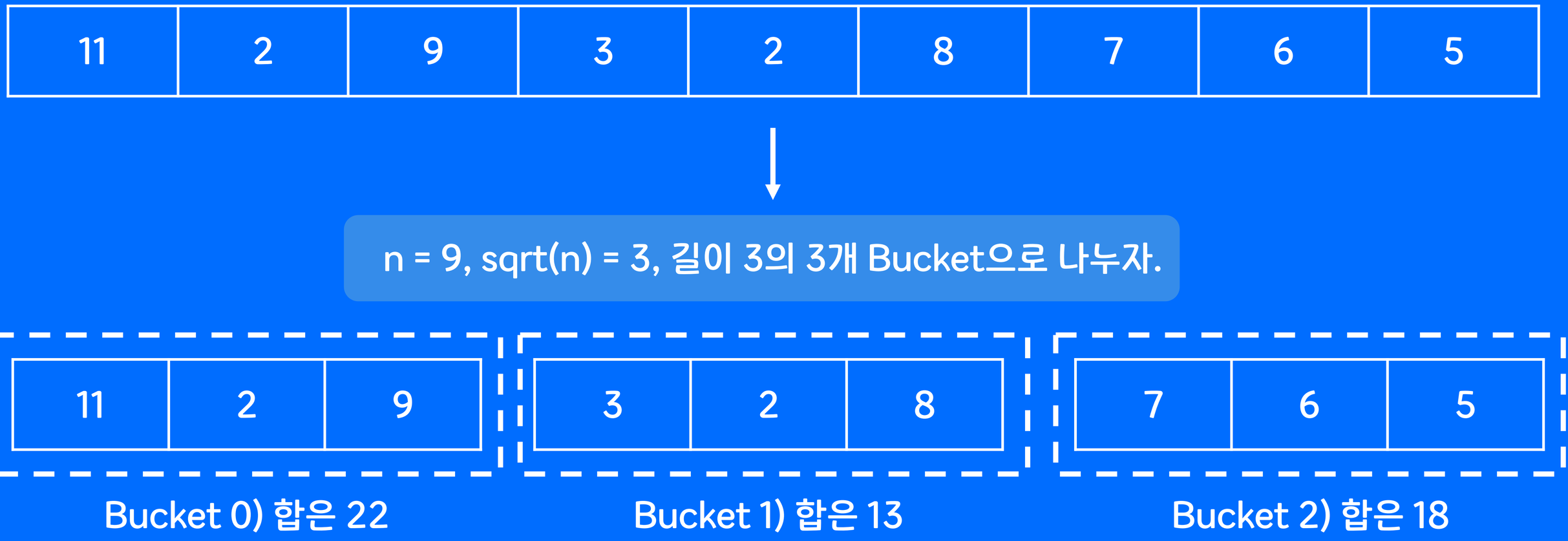
Query (4): 3번째부터 5번째까지의 합을 구해라!



기본 원리

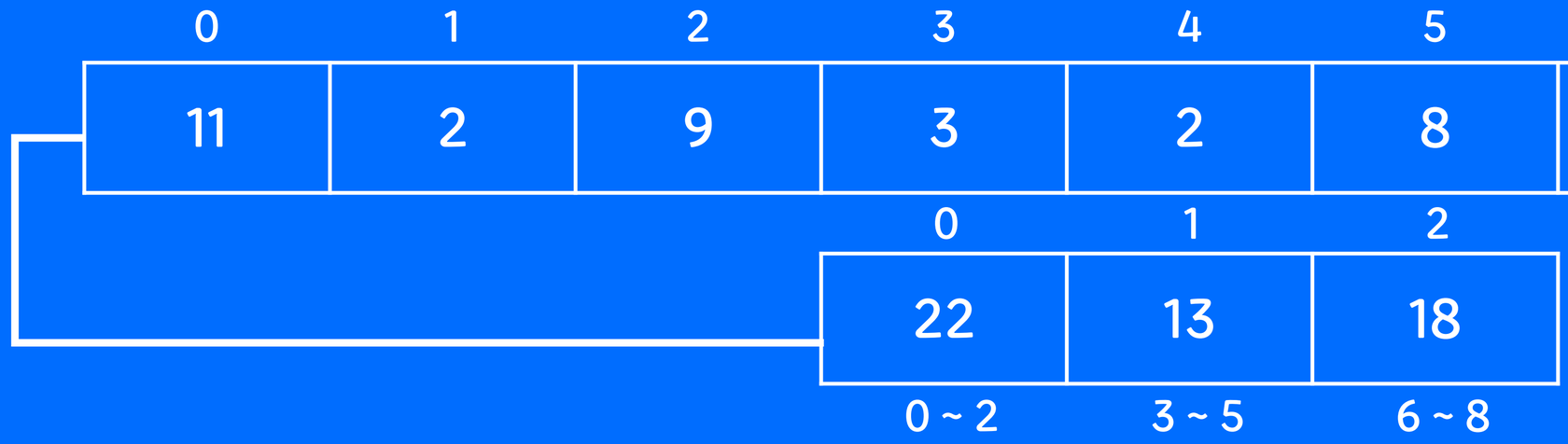
기본 원리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어

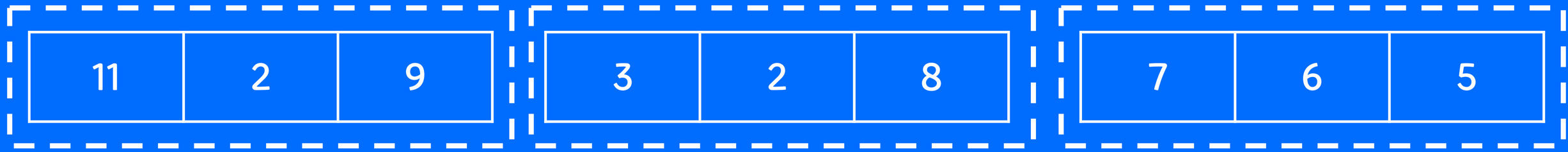


기본 원리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어



각 bucket마다 대표값(합, 최대값, 최소값 등)을 저장한다.



Bucket 0) 합은 22

Bucket 1) 합은 13

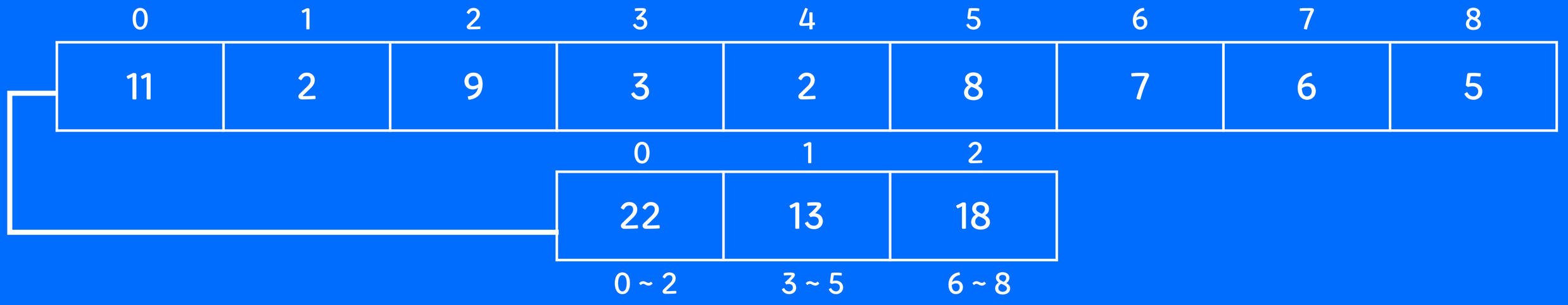
Bucket 2) 합은 18

기본 원리 - 구간 쿼리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어

정리하자면, 아래와 같은 배열이 남아있는 것이고, 이를 활용해보자.

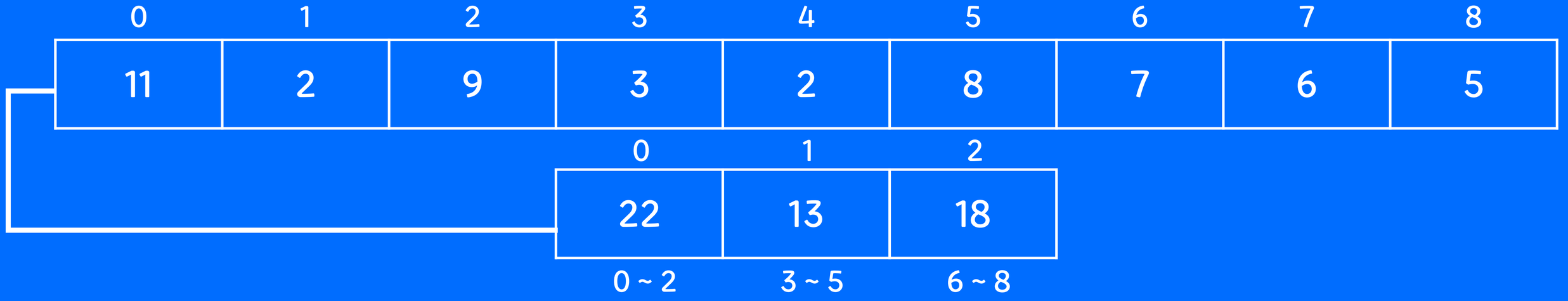
n = 9, sqrt(n) = 3



기본 원리 - 구간 쿼리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어

n = 9, sqrt(n) = 3



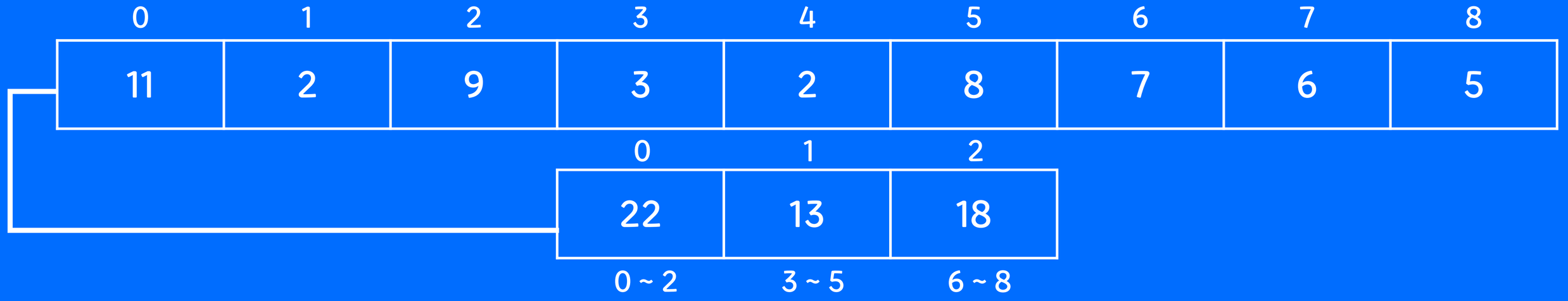
Query : 1번째 ~ 5번째 까지의 구간합을 구해보자.

1. 먼저 각 bucket을 살펴봅니다.
2. 0번째 버킷은 0 ~ 2로 원하는 쿼리의 구간(1 ~ 5)와 겹치지만, 0번째 값은 포함되면 안되는 상황입니다.
   이때문에 바로 해당 버킷의 값을 더할 수는 없습니다.
   따라서, 1번째 칸 (2), 2번째 칸 (9)은 각각 더해나갑니다.
3. 1번째 버킷은 3 ~ 5로 원하는 쿼리의 구간에 완벽히 포함됩니다. 바로 버킷에 있는 대표값 13을 그대로 더합니다.
4. 2번째 버킷은 구간을 아예 벗어났습니다. 그대로 종료하여, 답은 2 + 9 + 13 = 24 입니다.

기본 원리 - 갱신 쿼리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어

n = 9, sqrt(n) = 3



Query : 4번째 값을 5로 바꾸자.

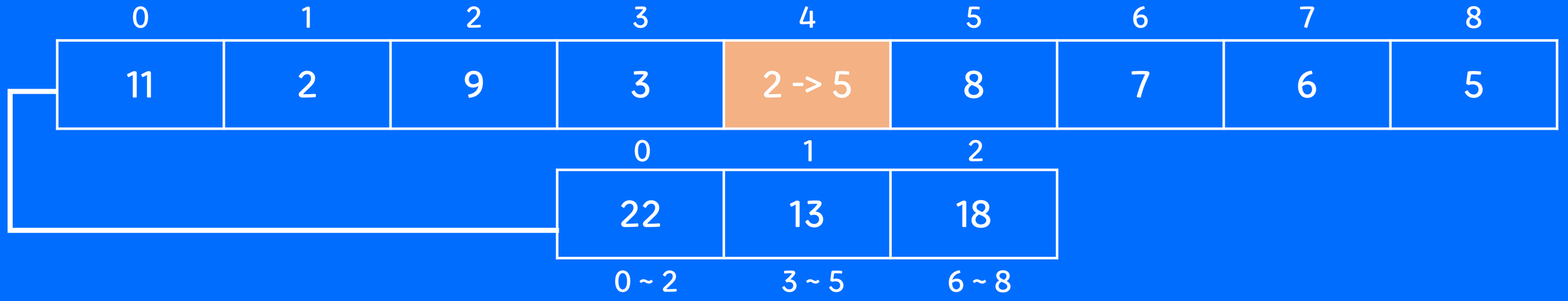
우리가 가지고 있는 배열은 딱 2개 입니다. 이 배열 2개만 갱신해가면 됩니다.

위의 원본 배열은 간단합니다. 곧바로 값을 바꾸면 됩니다.

기본 원리 - 갱신 쿼리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어

n = 9, sqrt(n) = 3



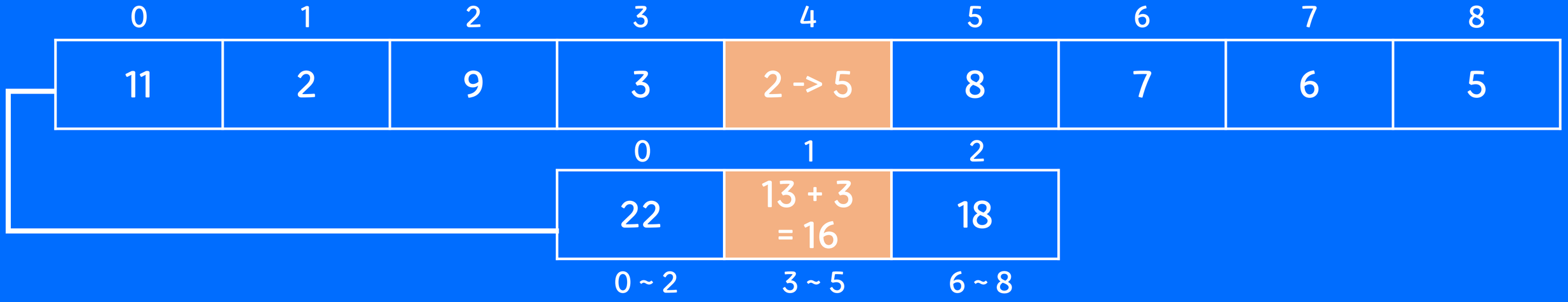
Query : 4번째 값을 5로 바꾸자.
우리가 가지고 있는 배열은 딱 2개 입니다. 이 배열 2개만 갱신해가면 됩니다.

위의 원본 배열을 갱신하는 것은 간단합니다. 곧바로 값을 바꾸면 됩니다.
위와 같이 바뀔 것입니다.

기본 원리 - 갱신 쿼리

제곱근(sqrt) 길이로, 제곱근 개의 Bucket으로 나누자는 아이디어

n = 9, sqrt(n) = 3



Query : 4번째 값을 5로 바꾸자.

우리가 가지고 있는 배열은 딱 2개 입니다. 이 배열 2개만 갱신해가면 됩니다.

bucket 배열은 어떻게 해야할까요? 원본 배열에 주어진 변화를 대표값에도 반영하면 됩니다.

2 -> 5로 3 증가 했기 때문에, 그대로 +3을 적용해주면 됩니다. (13 + 3 = 16)

구현 방법

구현 방법 (기초)

C++ 코드

```
class SqrtSumDecomposition {
public:
  vector<ll> arr, buckets;
  ll n, bucketSize;

  int bucketIndex(int idx) {
    return idx / bucketSize;
  }

  SqrtSumDecomposition(const vector<ll>& input)
{   n = input.size();
    bucketSize = sqrt(n) + 1;

    arr.resize(n);
    buckets.resize(bucketSize + 1, 0);

    for1(0, n) {
      arr[i] = input[i];
      buckets[bucketIndex(i)] += arr[i];
    }
  }
```

SqrtSumDecomposition 이라는 클래스를 통해 구현된 코드입니다.
다음 슬라이드에 이어서 메서드들이 작성되어 있습니다.

- arr : 원본 배열입니다.
- buckets : 각 버킷마다 대푯값들을 저장하는 배열입니다.

- buncketIndex() : 특정 idx가 포함될 bucket index를 반환합니다.
- SqrtSumDecomposition() : 벡터 (리스트)가 들어오면 이에 대해 초기화합니다.

구현 방법 (기초)

C++ 코드

```
void update(int idx, ll value) {
    buckets[bucketIndex(idx)] += value - arr[idx];
    arr[idx] = value;
 }
```

SqrtSumDecomposition 이라는 클래스 중
update와 query 메서드 내용입니다.

- update(idx, value) : 특정 indexㄹㄹ valude로 업데이트 합니다.
  (bucket에는 현재 특정 구간의 합이 들어가있기 때문에 새로이 갱신하는 것이 아니라 차분을 더해줍니다.)
- query(l, r) : l ~ r 구간의 합을 구합니다.

해당 코드는 구간합에 대한 쿼리를 다룹니다. 최대, 최소 등 다양한 쿼리를 대비하는 코드도 만들어보면 좋습니다.

```
ll query(ll l, ll r) {
    ll sum = 0;
    int startBucket = bucketIndex(l);
    int endBucket = bucketIndex(r);

    if(startBucket == endBucket) {
      for1(l, r + 1) {
        sum += arr[i];
      }
    } else {
      for1(l, (startBucket + 1) * bucketSize)
{       sum += arr[i];
      }
      for1(startBucket + 1, endBucket) {
        sum += buckets[i];
      }
      for1(endBucket * bucketSize, r + 1) {
        sum += arr[i];
      }
    }

    return sum;
  }
};
```

연습 문제 (기초)

2042번 : 구간 합 구하기

https://www.acmicpc.net/problem/2042

사실, sqrt decomposition은 Segment Tree를 이용해서 풀이가 가능합니다.

2357번 : 최솟값과 최댓값

https://www.acmicpc.net/problem/2357

최소값, 최대값에 대해 쿼리를 구성하면 됩니다.

갱신 쿼리의 경우, 앞서 구간합과 다르게 차분을 적용하는 것이 아니라 곧바로 최대, 최소값으로 갱신해나가면 됩니다.

연습 문제 (심화)

13545번 : 수열과 쿼리 0

https://www.acmicpc.net/problem/13545

i j 쿼리가 주어지면
Ai, Ai+1, ..., Aj로 이루어진 부분 수열 중에서 합이 0이면서 가장 긴 연속하는 부분 수열의 길이를 반환하는 문제입니다.

Mo’s algorithm도 같이 알아볼 수 있는 문제입니다.

9426번 : 중앙값 측정

https://www.acmicpc.net/problem/9426

길이가 k인 모든 연속 구간에 대해서 중앙값을 출력해야 합니다.
직접적인 값의 갱신은 없어 보입니다. 곧바로 중앙값을 각자 구할 수 있지 않나 싶지만 그렇지 않은 문제입니다.

N과 K가 큰 편이라 그때그때 중앙값을 구할 수 없습니다.

힌트) 각 구간을 슬라이딩 하면서 현재까지 포함된 숫자 자체의 개수를가 원본 배열이 되고, 이를 버킷에 나누어 기록해가면서 처리할 수 있습니다.

Thank You

모두의 알고리즘 학습 여정을 응원합니다!

Algorithm Session
dev.shinkeonkim@gmail.com

김신건
Koa