

Prefix Sum & Binary Search Basic

누적합, 이분탐색 기초

競技プログラミングの鉄則

KPSC Algorithm Study 24/11/14 Thu.

by Haru_101

Prefix Sum

- 누적합(Prefix Sum)은 배열의 특정한 범위의 합을 효율적으로 구하는 방법
- 쿼리가 다음과 같이 주어졌다고 생각해봅시다.
 - l 번째 값부터 r 번째 값까지의 $A_l + A_{l+1} + \dots + A_r$ 을 출력하라.
- 일단 반복문을 사용해서 풀어볼까요?

Prefix Sum

- $l = 2, r = 6$

now = 2
sum = 2



1	2	3	6	5	4
---	---	---	---	---	---

Prefix Sum

- $l = 2, r = 6$

now = 3
sum = 5



1	2	3	6	5	4
---	---	---	---	---	---

Prefix Sum

- $l = 2, r = 6$

now = 4
sum = 11



1	2	3	6	5	4
---	---	---	---	---	---

Prefix Sum

- $l = 2, r = 6$

now = 5
sum = 16

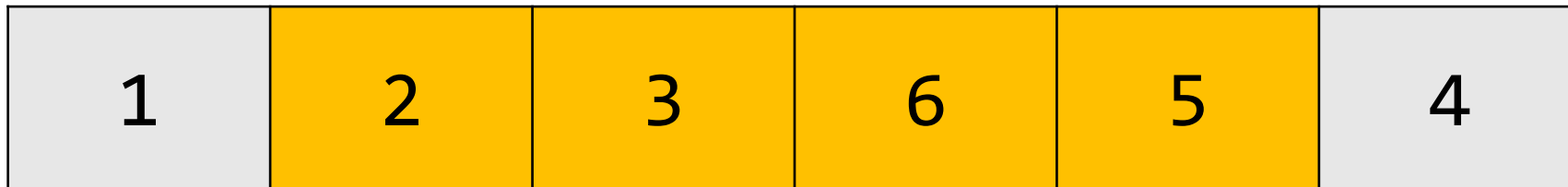


1	2	3	6	5	4
---	---	---	---	---	---

Prefix Sum

- $l = 2, r = 6$

now = 6
sum = 20



Prefix Sum

- $l = 2, r = 6 \rightarrow 20$

1	2	3	6	5	4
---	---	---	---	---	---

Prefix Sum

- 반복문을 사용하면, l 부터 r 까지의 모든 원소를 반복해서 더해나가는 과정이 필요합니다.
- $l = 1, r = n$ 인 쿼리가 Q 번 주어졌을 때 시간복잡도를 구해보면,
 - $O(Qn)$
 - $n = 100,000, Q = 100,000$ 이면 주어진 시간내에 해결하지 못합니다.
- 따라서 이를 효율적으로 계산하는 방법이 필요합니다.

Prefix Sum

- 아까 봤던 배열을 변형해서 봐봅시다.

1	2	3	6	5	4
---	---	---	---	---	---

- 노랗게 칠한 부분을 어떻게 빠르게 구할 수 있을까요?

Prefix Sum

1	2	3	6	5	4
---	---	---	---	---	---

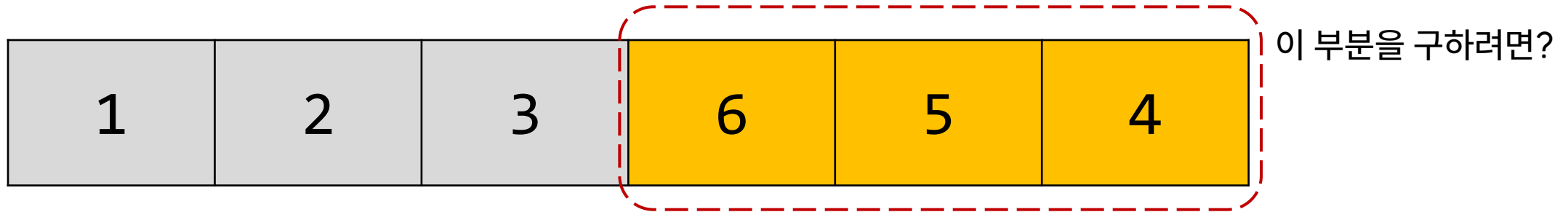
=

1	2	3	6	5	4
---	---	---	---	---	---

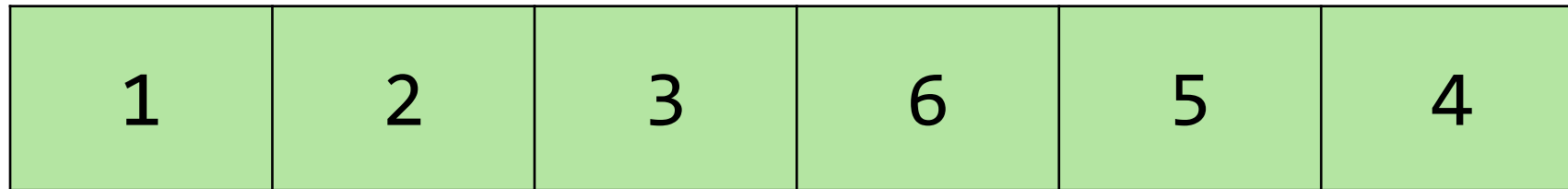
-

1	2	3			
---	---	---	--	--	--

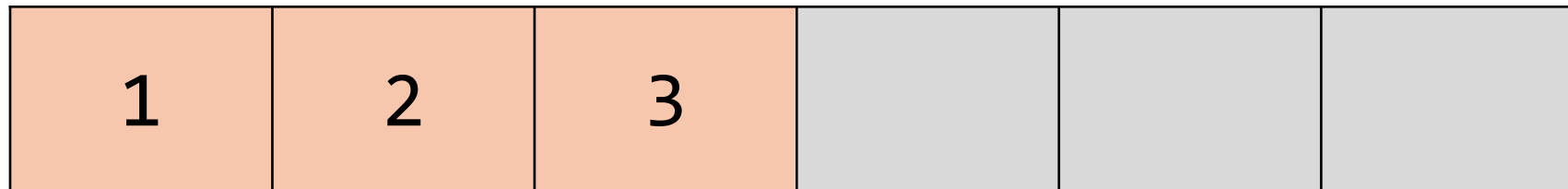
Prefix Sum



=



-



Prefix Sum

1	2	3	6	5	4
---	---	---	---	---	---

=

1	2	3	6	5	4
---	---	---	---	---	---

전체 합을 구하고

=

1	2	3			
---	---	---	--	--	--

Prefix Sum

1	2	3	6	5	4
---	---	---	---	---	---

=

1	2	3	6	5	4
---	---	---	---	---	---

1	2	3			
---	---	---	--	--	--

이 부분의 합을 빼면 된다!

Prefix Sum

- 누적합 알고리즘은 특정 범위에 대한 값을 빠르게 구할 수 있는 알고리즘입니다.
- 'l번째 값부터 r번째 값까지의 $A_l + A_{l+1} + \dots + A_r$ 을 출력하라.' 쿼리에 대해 다시 생각해보죠.
 - $l = 4, r = 6$ 일때,
 - 인덱스 1~6까지의 합에서 인덱스 1~3까지의 합을 빼면 인덱스 4~6의 합을 구할 수 있습니다!

1	2	3	6	5	4
---	---	---	---	---	---

Prefix Sum

- 반복문으로 구하는 것과 무슨 차이가 있냐 하면,
 - 일반적인 반복문으로 구한다 -> 각 쿼리에 최대 $O(n)$ 만큼 소요
 - 누적합 알고리즘을 사용한다 -> 초기에 합을 누적해서 더하고 기록하는데에 $O(n)$ 소요, 각 쿼리에 $O(1)$ 소요

원본 배열	1	2	3	6	5	4
누적합 배열	$S_1 = 1$	S_2	S_3	S_4	S_5	S_6
누적합 배열	1	3	6	12	17	21

Prefix Sum

- $l = 4, r = 6$ 일때, $S_6 - S_3 = 21 - 6 = 6 + 5 + 4 = 15$

원본 배열	1	2	3	6	5	4
누적합 배열	$S_1 = 1$	S_2	S_3	S_4	S_5	S_6
누적합 배열	1	3	6	12	17	21

Prefix Sum

- 문제를 풀어봅시다. [번역은 다음페이지]
- https://atcoder.jp/contests/tessoku-book/tasks/math_and_algorithm_ai
- 遊園地「ALGO-RESORT」では N 日間にわたるイベントが開催され、 i 日目($1 \leq i \leq N$)には A_i 人が来場しました。
以下の合計 Q 個の質問に答えるプログラムを作成してください。
 - 1個目の質問： L_1 日目から R_1 日目までの合計来場者数は？
 - 2個目の質問： L_2 日目から R_2 日目までの合計来場者数は？
 - ...
 - Q 個目の質問： L_Q 日目から R_Q 日目までの合計来場者数は？

Prefix Sum

- 문제를 풀어봅시다.
- https://atcoder.jp/contests/tessoku-book/tasks/math_and_algorithm_ai
- 유원지 'ALGO-RESORT'에는 N 일간에 걸쳐 이벤트가 개최되어, i 일째 ($1 \leq i \leq N$)에는 A_i 명의 사람이 왔습니다. 다음과 같이 Q 개의 쿼리에 대해 답하는 프로그램을 작성하시오.
 - 1번째 쿼리 : L_1 일부터 R_1 일까지 온 사람의 총합은?
 - 2번째 쿼리 : L_1 일부터 R_1 일까지 온 사람의 총합은?
 - ...
 - Q 번째 쿼리 : L_Q 일부터 R_Q 일까지 온 사람의 총합은?
- 출력 : Q 행에 걸쳐 각 쿼리의 결과를 출력하라.

Prefix Sum

- 이번엔 2차원 배열 상에서의 누적합 알고리즘을 사용하는 법을 알아보시다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

Prefix Sum

- 화살표 방향으로 보면, 각 행에 대해 누적합 배열을 작성할 수 있습니다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

Prefix Sum

원본 배열

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

누적합 배열(행단위)

1	4	9	16
2	6	12	20
1	3	6	10
5	11	18	26

Prefix Sum

- 하지만, 이걸로 끝나면 모든 문제가 쉽게 풀리겠지만... 현실은 그렇지 않습니다.
- 쿼리가 다음과 같다고 합시다.
 - x_1, y_1, x_2, y_2
배열의 (x_1, y_1) 부터 (x_2, y_2) 까지의 합을 구하라.
- 이 쿼리를 풀기 위해선 y_1 부터 y_2 까지 각 행마다 $S_{x_2} - S_{x_1-1}$ 을 구하면 됩니다.
 - 그러면 $y_1 = 1, y_2 = n$ 일때 $O(n)$ 의 시간이 걸리고,
이를 Q 번 처리해야하므로, $O(Qn)$ 의 시간이 걸립니다.
- 이를 어떻게 해결하면 좋을까요?

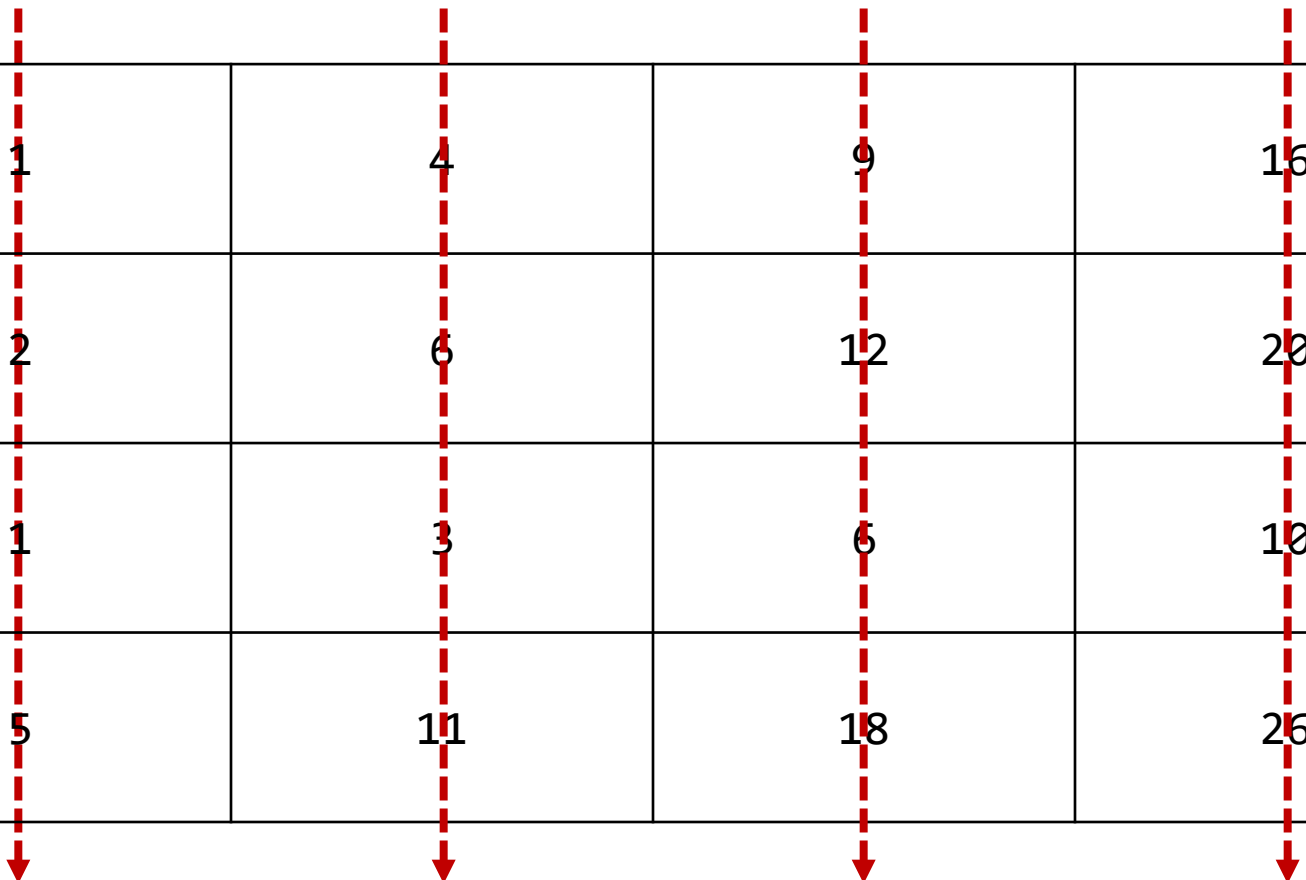
누적합 배열(행단위)

1	4	9	16
2	6	12	20
1	3	6	10
5	11	18	26

Prefix Sum

- 화살표 방향으로 한번 더 누적합을 수행하면 됩니다.

1	4	9	16
2	6	12	20
1	3	6	10
5	11	18	26



The diagram illustrates a 4x4 grid of numbers. Four vertical red dashed lines are drawn between the columns, with arrows pointing downwards at the bottom of each line, indicating a second cumulative sum operation being performed in the vertical direction.

Prefix Sum

원본 배열

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

최종 누적합 배열

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- 결국 이 최종 누적합 배열은, 각 (x, y) 에 대해, $(1, 1) \sim (x, y)$ 까지의 누적합을 저장하게 됩니다.
- 근데... $(1, 1) \sim (x, y)$ 말고 $(x_1, y_1) \sim (x_2, y_2)$ 의 합은 어떻게 구하죠?

최종 누적합 배열

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- $x_1 = 2, y_1 = 2, x_2 = 3, y_2 = 4$

원본 배열

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

최종 누적합 배열

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- 그렇다면, 하늘색 부분에서 노란색 부분을 빼면 초록색이 나오지 않을까요?

1	3	5	7	1	3	5	7
2	4	6	8	2	4	6	8
1	2	3	4	1	2	3	4
5	6	7	8	5	6	7	8

Prefix Sum

- 이를 행, 열 단위로 보면, 하늘색에서 노란색을 뺀 것과 같겠죠?

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

Prefix Sum

- 근데, 가장 왼쪽 윗부분이 2번 빠지게 됩니다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

Prefix Sum

- 그러면, 보라색 부분을 2번 빼고 1번 더하면, 1번 뺀 것과 같습니다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

Prefix Sum

- 이를 누적합 관점에서 봅시다.
- 하늘색의 합은 누적합 배열에서 노란색 부분으로 나타낼 수 있습니다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- 이를 누적합 관점에서 봅시다.
- 주황색은 합은 누적합 배열에서 보라색 부분으로 나타낼 수 있습니다. (가로방향)

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- 이를 누적합 관점에서 봅시다.
- 주황색은 합은 누적합 배열에서 보라색 부분으로 나타낼 수 있습니다. (세로방향)

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- 이를 누적합 관점에서 봅시다.
- 이제 2번 겹치는 부분을 보라색 부분으로 나타내봅시다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72

Prefix Sum

- 이를 누적합 관점에서 봅시다.
- 결론적으로, 쿼리에 대한 출력을 구하기 위해서는 노란색 - (보라색 2개) + 주황색을 하면 됩니다.

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	3	5	7
2	4	6	8
1	2	3	4
5	6	7	8

1	4	9	16
3	10	21	36
4	13	27	46
9	24	45	72