c++ upper_bound, lower_bound 활용하기

upper_bound, lower_bound 란?

<algorithm> 헤더에 존재한다.

C++ 에서 제공하는 이분 탐색 기반의 원소 탐색 함수

이분 탐색 기반이므로 배열 안의 원소들이 반드시 정렬되어 있어야 사용 가능하다.

강의에서는 vector 기준으로만 설명할거에요!

upper_bound

기본 형태

upper_bound(v.begin(), v.end(), value)

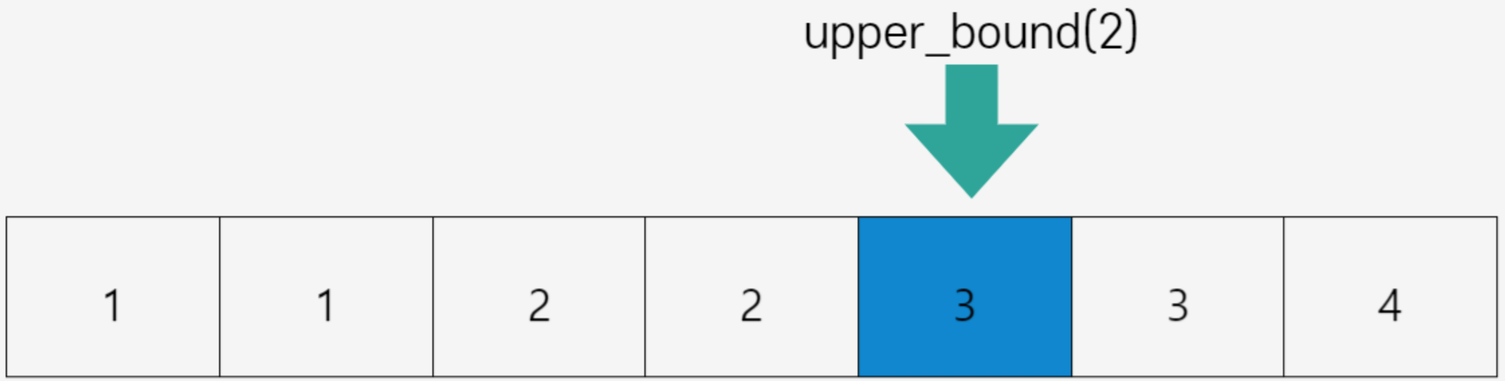
단순하게, 벡터의 시작(v.begin()) 과 끝(v.end()) 사이에 찾고자 하는 값 value를 넣으면 된다.

upper_bound

의미

배열에서 value 값을 초과하는 가장 첫 번째 원소의 위치를 구한다.

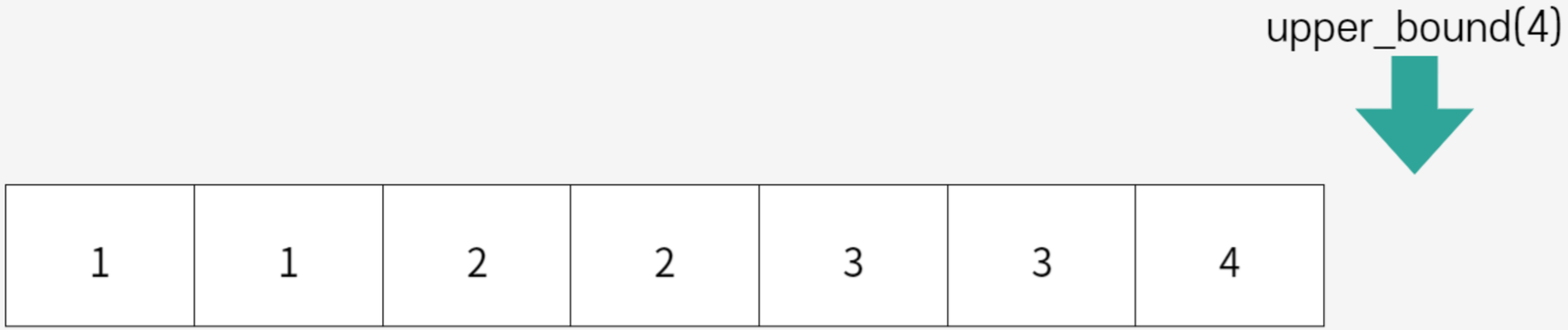
ex) value = 2



upper_bound

배열에서 가장 큰 값을 가리킬 때

ex) value = 4

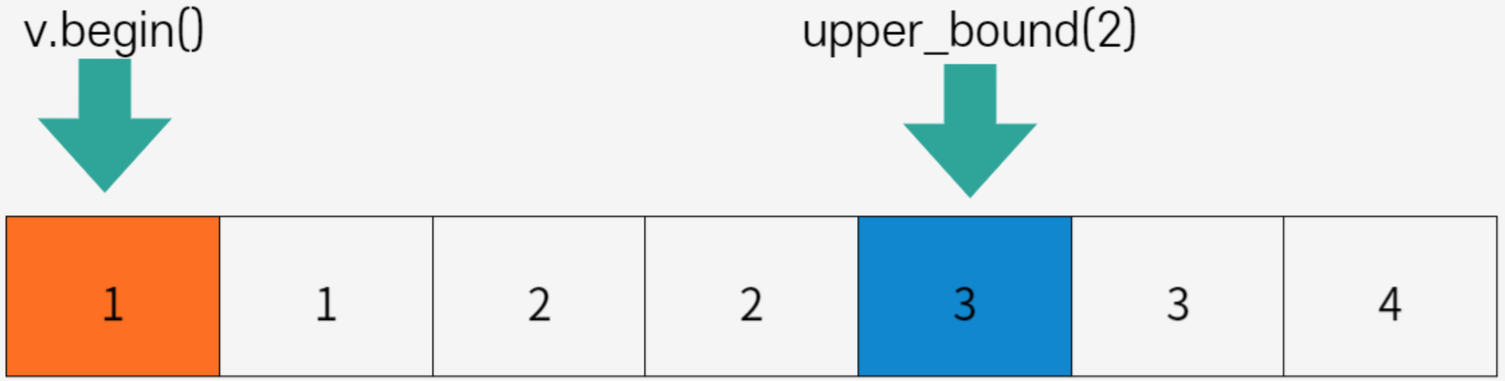


upper_bound

인덱스(위치)를 반환하려면?

upper_bound 의 리턴값은 iterator(반복자) 이므로 같은 반복자끼리 빼줘야 한다!

-> upper_bound() - v.begin()



upper_bound 직접 구현하기

```cpp
vector<int>v;

//val : 찾고자 하는 원소
int myupper_bound(int val)
{
	int left = 0, right = v.size() - 1;
	int ans = 0;
	while (left <= right)
	{
		int mid = (left + right) / 2;
		//찾은 원소가 value보다 작거나 같을 때
		//-> 같은 수 중 더 뒤에 있는 인덱스가 있는지 확인한다.
		if (v[mid] <= val) {
			left = mid + 1;
			ans = mid;
		}
		//찾은 원소가 더 크다면 -> 수를 낮춰본다.
		else {
			right = mid - 1;
		}
	}
	return ans + 1;
}
int main()
{
	ios_base::sync_with_stdio(0); cin.tie(0);
	v = { 1, 1, 2, 2, 3, 3, 4 };
	int idx = myupper_bound(2);
	cout << idx << "\n";
	return 0;
}
```

upper_bound 갖다 쓰기

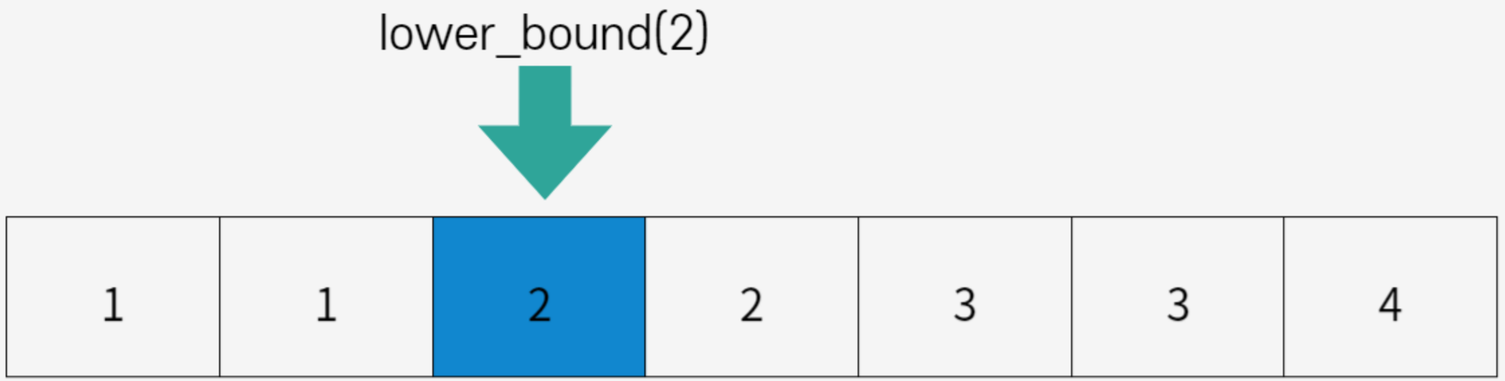
```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
vector<int>v;
int main()
{
    ios_base::sync_with_stdio(0); cin.tie(0);
    v = { 1, 1, 2, 2, 3, 3, 4 };
    int idx = upper_bound(v.begin(), v.end(), 2) - v.begin();
    cout << idx << "\n";
    return 0;
}
```

lower_bound

의미

value 값보다 크거나 같은 가장 첫 번째 원소의 위치를 구한다.

ex) value = 2



lower_bound

lower_bound는 "이상" 의 의미를, upper_bound는 "초과"의 의미를 갖는다고 보면 된다.

$\leq$ $<$

만약 찾는 value 값이 배열에 없다면?

-> "이상"에서 등호가 빠져도 상관없으니, 둘의 결괏값은 똑같다!

lower_bound 구현하기

upper_bound 코드 보면서 만들어봐요!

참고 문제

백준 10816 숫자 카드 2

2020 KAKAO BLIND RECRUITMENT - 가사 검색 (프로그래머스 lv.4 문제!)