ALOHA

#2주차멘토링

Binary Search

Parametric Search

LIS

Binary Search의 구현

Value 가 Arr[mid]보다 크다 => Value가 Mid + 1 ~ Right 구간에 있음

Value 가 Arr[mid]보다 작다 => Value가 Left ~ Mid - 1 구간에 있음

Value 가 Arr[mid]와 같다 => 찾았으므로 끝낸다

Binary Search 체크리스트

- [ ] 배열 정렬
- [ ] 원하는 값을 찾지 못했을 때의 구현
- [ ] Integer overflow

Binary Search Tips

Binary Search를 할 때 가장 많이 하는 실수는 integer overflow 라고 생각해요.

예를 들어, left = 0, right = 2147483647 로 잡아두고, 우리가 구현한 대로 Binary Search를 해버리면

mid = (left + right)/2;

에서 integer overflow가 발생할 수 있어요! 이 점을 항상 주의합시다.

Parametric Search 체크리스트

- [ ] 무엇을 기준으로 이진탐색?
- [ ] Mid값을 판단하는 알고리즘이 충분히 빠른가?
- [ ] 답이 단조성을 가지는가?

Parametric Search Tips

Parametric Search를 구현하는 방법엔 여러가지가 있어요.

크게는 반복문을 통한 구현, 재귀함수를 이용한 구현이 있고,
반복문을 통한 구현에도 만족하는 mid 값 중 가장 큰 것을 구하는지 가장 작은 것을 구하는지에 따라 구현 방법이 달라 질 수 있어요.

하지만, 반복문을 통한 구현을 하는데 어떤 문제는 while(left < right) 라고 하고, 다른 문제는 while(left <= right) 라고 구현하면 반복문 안쪽을 구현하는게 힘들어질 수도 있어요. 그래서 이런 구현 방법을 통일하는 게 중요하다고 생각해요.

LIS 체크리스트

- [ ] 배운 두 알고리즘의 차이점은?
- [ ] 결과로 나온 dp배열의 원소 != LIS의 원소
- [ ] lower_bound 의 사용법을 숙지

다음 장은 퀴즈입니다

QUIZ

1. 오름차순으로 정렬 되어있는 배열 a[0, 1, …, n-1] 이 있다. a[mid] 가 찾고자 하는 값인 value 보다 작을 때 *(a[mid] < value)*, left와 right 변수를 어떻게 조절해야 하는가?

```
19  int main(void)
20  {
21      std::cin.tie(0);
22      std::ios_base::sync_with_stdio(false);
23
24      int N, x;
25
26      std::cin >> N;
27      for (int i = 0; i < N; i++)
28      {
29          std::cin >> x;
30          v.push_back(x);
31      }
32
33      std::
34
35      int M;
36      std::cin >> M;
37      for (int i = 0; i < M; i++)
38      {
39          std::cin >> x;
40          std::cout << binarySearch(0, N - 1, x) << "\n";
41      }
42
43      return 0;
44  }
```

QUIZ

2. 다음은 이 프로그램에 대한 설명이다. 빈칸에 맞는 코드를 작성하여라.

입력
첫 줄에는 정수 N이 주어진다.
두 번째 줄에는 수열 a[0, 1, ···, n-1]이 공백으로 구분되어 주어진다.
다음 줄에는 정수 M이 주어진다.
이후 M개의 수가 공백으로 구분되어 주어진다.

출력
M개의 주어진 수가 수열에 존재하면 1, 존재하지 않으면 0을 출력한다.

단, binarySearch 함수는 수열 a를 이진 탐색하여 value 가 수열에 존재하면 1, 존재하지 않으면 0을 반환한다.

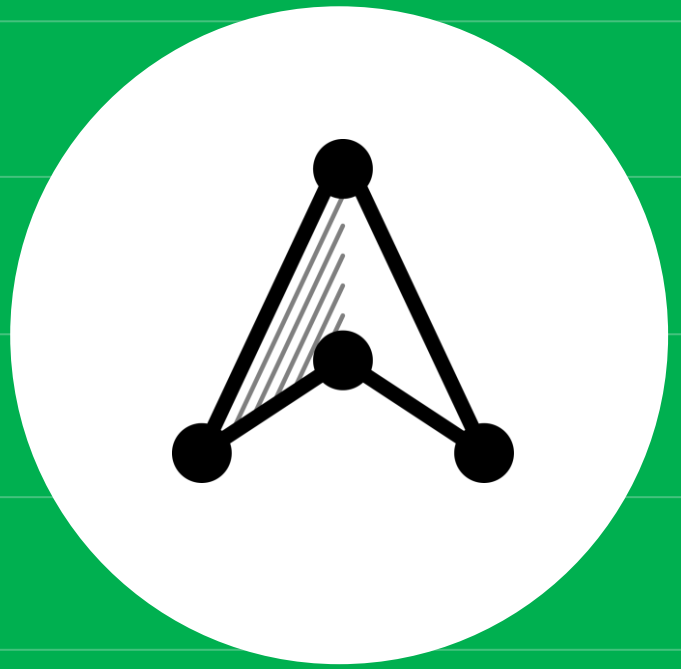
QUIZ

3. 다음 중 옳은 것을 모두 고르시오.

A. 이분탐색의 시간복잡도는 O(lgN)이다.

B. lower_bound 함수는 구간 [first, last)에서 처음으로 value 보다 큰 수가 나타나는 위치를 반환한다.

C. LIS를 O(NlgN)의 시간복잡도로 구하는 알고리즘은 LIS의 길이만 알 수 있다.



다음 시간에 만나요~