[알고리즘 강의] 6주차. 이분탐색과 LIS(최대증가부분수열)

관리자 관리자 2021.3.23.오후 1:13:41
수정하기

6주차

6-6

시간탐색(이진탐색)

이분탐색(binary search)

이분탐색은 어떤 경우라 볼 수 있을까요? 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30

이분탐색이란?

이분탐색은 어떤 범위에서 찾고자 하는 대상을 찾기하는것을 log2의 복잡도로 시간안에 찾을수 있다.

이분탐색은 시작과 종료의 중간지점을 찾아서 조건이 맞는지 판별하고 맞으면 시작과 중간 사이를 찾고 아니면 중간 지점 부터 종료까지 찾는다.

이분탐색의 조건은 찾고자 하는 대상이 정렬되어 있어야 한다는 것이다.

정렬되지 않은 경우에는 sort를 해서 이러한 형태 상태로 만들어 주고 이분탐색을 시작한다. 그 중요한 이유는 찾아 보면 알겠지만 이분탐색은 중간지점 부터 시작하기 때문 이다 예를 들어 1,2,3 숫자가 있다고 가정 했을 때 이 숫자들을 1,2,3 순서대로 찾는다고 하면 1 찾고 2 찾고 3 찾고 이렇게 3번 찾아야 하지만 이분탐색은 2 부터 찾는다 그러면 1 찾고 3 찾는 것 이렇게 2번만 찾아도 된다.

이 방법대로 정렬되지 않은 상태로 이분탐색을 하면 이분탐색이 제대로 작동하지 않는다 왜냐하면 lower_bound 또는 upper_bound 함수 자체가 정렬 되지 않은 상태 에서 이분탐색을 하기 때문에 이분탐색이 제대로 작동하지 않는다 그렇기 때문에 꼭 정렬되지 않은 상태 에서 이분탐색을 하려면 lower_bound 또는 upper_bound 함수 자체를 직접 만들어 주어야 한다.

백준 암기왕을 풀어보도록 하겠습니다.

```
#include bits/stdc++.h
using namespace std;
typedef long long ll;
int t, n, m, temp;
unordered_map<int int> mp;
int check(int temp, vector<int> &v){
    int l = 0, r = v.size();
    int mid;
    while(l <= r){
        mid = (l + r) / 2;
        if(v[mid] == temp) r = mid - 1;
        else if(v[mid] > temp) return 1;
        else l = mid + 1;
    }
    return 0;
}
int main(){
    scanf("%d", &t);
    while(t--){
        vector<int> v;
        scanf("%d", &n); for(int i = 0; i < n; i++)scanf("%d", &temp), v.push_ba
        sort(v.begin(), v.end());
        scanf("%d", &m); for(int i = 0; i < m; i++)scanf("%d", &temp), cout << c
    }
    return 0;
}
```

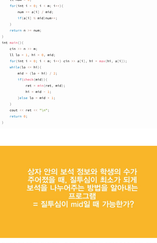
이 이분탐색의 수도코드는 이 부분입니다.

```
while(l <= r){
    mid = (l + r) / 2;
    if(v[mid] == temp) r = mid - 1;
    else if(v[mid] > temp) return 1;
    else l = mid + 1;
}
```

이렇게 sort라는 함수를 사용하지 않고 이분탐색 하는 것입니다.

```
function binary_search(a, n, t) {
    l = 0
    r = n - 1
    while (l <= r) {
        m = Math.floor((l + r) / 2)
        if (a[m] == t)
            return 1;
        else if (a[m] < t)
            l = m + 1;
        else
            return 0
    }
    return unsuccessful;
}
```

이렇게는 자바 언어 또는 파이썬 문법으로 하겠습니다.



백준 보석상자 이번 문제를 풀어보도록 하겠습니다.

이번문제 문제의 특징은 두개의 문제를 결합한 문제로 바뀐다는 것입니다. 이러한 경우 분리해야 하는지를 아는 것이 중요합니다 문제에서 말하는 대로 이렇게 할게 하고 이렇게 하면 안되는 것이 있는지를 잘 따져 보고 이렇게 하는 것입니다. 만일 그러한 일이 있다면 분할 정복 이라고 하는데 이것이 중요합니다.

```
#include bits/stdc++.h
using namespace std;
#define first f
#define second s
typedef long long ll;
ll n, m, arr[100001], ret = 1e12;
bool check(ll mid){
    ll cnt = 0;
    for(int i = 1; i <= n; i++){
        ll tmp = arr[i] / mid;
        if(arr[i] % mid != 0)
            tmp++;
        cnt += tmp;
    }
    return cnt <= m;
}
int main(){
    cin >> n >> m;
    for(int i = 1; i <= n; i++)
        cin >> arr[i];
    for(ll l = 1, r = 1e9, mid; l <= r; mid = (l + r) / 2, r = mid - 1, a[0]=1)
        if(check(mid))
            ret = min(ret, mid);
        else l = mid + 1;
    cout << ret << "\n";
    return 0;
}
```

이 문제는 이렇게 하면 됩니다. 제가 지금부터 대기업 코테 동물 버전이라 자칭하는거 그런 나쁜습니다. 저만 코테에서 나왔는지는 저도 모릅니다.

하지만 이 LIS 최대증가부분수열은 꼭 알아 두면 좋을거다. 정말로.

LIS 최대증가부분수열 이 문제는 정말로 꼭 알아두셔야 합니다 정말로 꼭 알아두셔야 하죠 그것 같다기보다 하는 거죠 이것이 이거 이거 이거