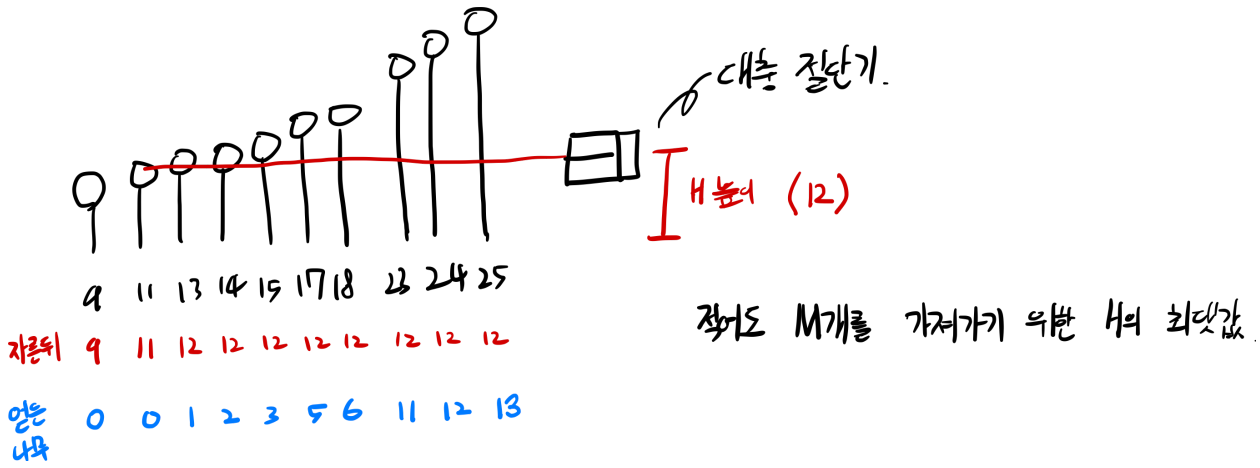
[Baekjoon] 2805 - 나무자르기

태그	baekjoon
공개여부	☑
작성일자	@January 20, 2022

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main(void)
{
    ios::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);

    int N, M;
    long long L, R, mid, sum = 0;
    vector<int> tree;

    cin >> N >> M;
    for (int i = 0; i < N; i++) {
        int tmp;

        cin >> tmp;
        tree.push_back(tmp);
    }
```

```
    sort(tree.begin(), tree.end());
    L = tree[0];
    R = tree[tree.size() - 1];
    mid = L + ((R - L) / 2);
    while (L != mid) {
        sum = 0;
        for (int i = 0; i < N; i++) {
            if (mid < tree[i]) {
                sum += tree[i] - mid;
            }
        }
        if (sum < M) {
            if (sum == M) {
                R = mid;
            }
            else {
                R = mid-1;
            }
        }
        else {
            if (sum == M) {
                L = mid;
            }
            else {
                L = mid+1;
            }
        }
        mid = L + ((R - L) / 2);
    }
    sum = 0;
    for (int i = 0; i < N; i++) {
        if (R < tree[i]) {
            sum += tree[i] - R;
        }
    }
    if (sum >= R)
        mid = R;

    cout << mid << '\n';

    return 0;
}
```

성찬이형 유진님 성빈님과 모각코하면서 풀어봤는데, 파라메트리서치에 대한 깊은 이해없이 짜봐서그런가?? 뭔가 잘못푼것같아 파라메트릭서치와 함께 다시 짜고 공부해보았다.



이 문제에서 파라메트릭 서치를 사용한다는 것은, 0부터 N까지의 수 중 M을 만족하는 최댓값 H를 찾는것이다. 0부터 N의 중간을 사용하여 나무를 자르고 모았을때, M보다 작다면 모자르니까 왼쪽에서 탐색해야된다는 의미(오른쪽범위는 현재 높이보다 더 크기때문에 M보다 작을 수 밖에 없기 때문.)이다. 왼쪽범위에서 또 중간을 찾아 이 과정을 반복하면, 최댓값을 구할 수 있게된다.

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main(void)
{
    ios::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);

    long long N, M, H, L, R, mid;
    vector<int> tree;

    cin >> N >> M;
    for(int i = 0; i < N; i++) {
        int tmp;

        cin >> tmp;
        tree.push_back(tmp);
    }

    sort(tree.begin(), tree.end(), greater<int>());
    H = 0;
    L = 0;
    R = tree[0];
    while(L <= R)
```

```
    {
        long long sum = 0;

        mid = (L + R) / 2;
        for(int i = 0; i < tree.size(); i++)
        {
            if(tree[i] >= mid)
                sum += tree[i] - mid;
            else
                break;
        }

        if(sum >= M) {
            if(H < mid)
                H = mid;
            L = mid + 1;
        }
        else {
            R = mid - 1;
        }
    }

    cout << H << '\n';

    return 0;
}
```

Mid 를 구하는 과정을 내가 좀 복잡하게 생각했던것같다. 파라메트리 서치 자체는 쉽게 이해할 수 있었다.