1. 문제 요약

1. 목재 절단
2. 높이가 H 초과인 나무의 높이 H 위 부분을 채취
3. 나무를 필요한 만큼만 최소로 가져간다.
4. 따라서 절단기의 높이를 최대로 해야한다.

2. 접근 아이디어

1. 채취할 수 있는 나무의 양을 계산하는 메서드를 작성한다.
2. 이 메서드의 들어가는 인자를 조절한다.
3. 계산량을 줄이기 위해 이분 탐색을 사용한다.

3. 시간 복잡도

O(N log N)

136188KB, 536ms

4. 구현 코드

```
import java.io.*;
import java.util.Arrays;

public class Main {
    private static StringBuilder sb = new StringBuilder();
    private static int maxValue = 0;
    private static int[] arr;
    private static int M;

    public static void main(String[] args) throws IOException {
        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

        String[] split = br.readLine().split(" ");
        int N = Integer.parseInt(split[0]);
        M = Integer.parseInt(split[1]);

        arr = new int[N];

        split = br.readLine().split(" ");

        for (int i = 0; i < split.length; i++) {
            arr[i] = Integer.parseInt(split[i]);

            if (arr[i] >= maxValue) {
                maxValue = arr[i];
            }
        }
```

```
        int l = 0;
        int r = maxValue;

        int mid = -1;

        while (l < r) {
            mid = (l + r) / 2;

            long result = calculate(mid);

            if (result < M) {
                r = mid;
            }
            else {
                l = mid + 1;
            }
        }

        System.out.println(l - 1);
    }

    private static long calculate(int height) {
        long sum = 0;

        for (int i = 0; i < arr.length; i++) {
            int diff = arr[i] - height;

            if (diff > 0) {
                sum += diff;
            }

        }

        return sum;
    }
}
```