**Ví dụ**

**Input**

4 7

20 15 10 17

**Output**

15

Nếu đặt cưa ở vị trí có chiều cao là 15:

20 - 15 = 5 m gỗ

15 - 15 → không thu được

10 - 15 → không thu được

17 - 15 = 2 m gỗ

5 + 0 + 2 = 7

→ Đặt cưa ở độ cao **15** thỏa mãn số lượng gỗ yêu cầu

→ Đặt cưa ở độ cao 16 chỉ thu được 5m gỗ

**Input**

5 20

4 42 40 26 46

**Output**

36

Khi đặt cưa ở chiều cao 36

4 - 36 → không thu được gỗ

42 - 36 = 6 m gỗ

40 - 36 = 4 m gỗ

26 - 36 → không thu được gỗ

46 - 36 = 10 m gỗ

Khi đặt cưa ở chiều cao 36, thì thu được 6 + 4 + 10 = 20

→ **36**

**Yêu cầu:** Tìm 1 vị trí cao nhất, mà khi tiến hành cưa gỗ, ta thu được ít nhất M mét gỗ

Cách đơn giản nhất: For duyệt từng vị trí để đặt cưa

→ O(N \* max(H)) → 10^6 \* 10^9

Giả sử đặt cưa ở vị trí x, và thu được sum mét gỗ

Khi x tăng, sum giảm

Khi x giảm, sum tăng

→ x và sum nghịch biến

giả sử, ở một vị trí mid, thì chúng ta thu được 1 lượng gỗ là sum

Nếu sum > M → kết quả > mid

Nếu sum < M → kết quả < mid

**Tính tổng gỗ:**

Khi đặt cưa ở vị trí x, thì nếu một cái cây thứ i có chiều cao cao hơn x, thì từ cái cây đó, ta thu được h[i] - x

getSum(h, x) {

sum = 0;

for i = 1 to N:

sum += max(0, h[i] - x);

return sum;

}

**Giải thuật**

**Bước 1:** Nhập dữ liệu

**Bước 2:** Chặt nhị phân để tìm kết quả

* left = 0, right = max(H[i])
* Với mỗi giá trị mid:
  + Nếu tổng số gỗ thu được >= M → gán kết quả bằng mid, đồng thời gán left = mid + 1
  + Nếu tổng số gỗ thu được < M → gán right = mid - 1
* Lặp lại tới khi left > right

**Bước 3:** In kết quả.

**Mã giả**

**function getSum(h, x): {**

**sum = 0;**

**for i = 1 to N:**

**sum += max(0, h[i] - x);**

**return sum;**

**}**

**fucntion BS\_Search(h, left, right, M): {**

**ans = left;**

**while left <= right:**

**mid = (left + right) div 2;**

**if getSum(h, mid) >= M:**

**ans = mid;**

**left = mid + 1;**

**else:**

**right = mid - 1;**

**return ans;**

**}**

**main() {**

**read(N, M);**

**for i = 1 to N:**

**read(h[i]);**

**left = 0; right = max(h);**

**print(BS\_Search(h, left, right, M));**

**}**

**Độ phức tạp: O(log(max(h)) \* N)**

**max(h) là chiều cao cao nhất**

**N là số lượng cây**