- 문제 · 각 쿠키 바구니에 든 쿠키의 수가 주어진다 ($1 \sim 500$).
  · 바구니 배열의 길이는 N ($1 \le N \le 2000$)
  · A, B에게 같은 수의 쿠키를 주어야한다.
  · 바구니의 순서는 '유사도'인데 최대한 비슷한 쿠키끼리 묶어서 주고싶다. ⇒ 연속된 바구니를 선택한다는 뜻.
  · A에게 ($1 \le l \le m$)
    B에게 ($m+1 \le l \le N$)
  · 각 아들에게 줄 수 있는 쿠키의 최대값.

- 접근.
  · N의 최대값이 2000이기 때문에
  모든 가능한 기준 m에 대해, l에 대해, r에 대해
  탐색하면 $O(n^3) \fallingdotseq 2^3 * 10^9$. (시간초과).
  · 인덱스, 배열의 모든 원소가 자연수., 최대값=자연수, 연속=부분합.
  ⇒ m을 기준으로한 $[1, m]$, $(m, N]$의 부분합을 선형 탐색.

두 배열의

↑ 이 구간이 연속된 idx를 가지므로
$[l, m]$, $(m, r]$ 범위의
두 배열이라는 m을 기준으로 하는
l의 선형 '감소' (증가X)
r의 선형 '증가' (감소X)
⇒ '투 포인터' 알고리즘!!
$O(n) \times O(n)$
↳ m결정 $1 \sim n-1$.

- 구현

```
for(int i=0; i<n-1; ++i) { // m은 0 ~ n-2
    int lo=i, hi=i; // 초기 상태에는 [lo, m], (m, hi)
    int left=coo[i], right=0; // 불일치로 시작해야 디버깅 쉬움.
    while(1) { // lo:[0, i], hi:(i, n-1)
        if(left <= right) {
            lo--; // 이미 포함되어 있으니 먼저 --
            if(lo==-1) break;
            left += coo[lo];
        } // lo=i, hi=i+1, left=coo[i], right=coo[i+1]로 시작할 수도 있지만, 처음부터 일치해버리기 때문에 여기로 맨위로 보내야함.
        else {
            hi++;
            if(hi==n) break;
            right += coo[hi];
        }
        if(left==right) ans=max(ans, left);
    } // 여기서 break 되어버리기 때문.
}
```

↳ 조건을 두가지로 분기한 뒤, 마지막에 위의 if와 결합하는 '동등'에 대한 if를 추가하는 것이 '경우의수'를 놓치지 않는 Tip!!

* 이러한 이유로 반복문에 분기가 많을 때는 경우의 수를 놓치기 쉬우므로 첫 반복은 흘려보내고 시작하는것이 낫다.

불일치 경우로

우의 코드에서는

lo = i+1, left=0 으로 초기화하여

lo: [0, i+1)
hi: [i+1, n)

← 이렇게 구간을 서로 0으로하는 반열린구간으로 표현할 수도 있다.

10과 h1을 대칭적으로
둘다 길이 0 짜리에서 시작.

— 다른 풀이 : '이분 탐색'도 가능하다.