백준 2343 기타레슨 – 파라매트릭 서치

2025-05-25 14:30~

강토는 자신의 기타 강의 동영상을 블루레이로 만들어 판매하려고 한다. 블루레이에는 **총 N개의 강의**가 들어가는데, 블루레이를 녹화할 때, **강의의 순서가 바뀌면 안 된다**. 순서가 뒤바뀌는 경우에는 강의의 흐름이 끊겨, 학생들이 대혼란에 빠질 수 있기 때문이다. 즉, i번 강의와 j번 강의를 같은 블루레이에 녹화하려면 i와 j 사이의 모든 강의도 같은 블루레이에 녹화해야 한다.
강토는 이 블루레이가 얼마나 팔릴지 아직 알 수 없기 때문에, **블루레이의 개수를 가급적 줄이려고 한다**. 오랜 고민 끝에 강토는 **M개의 블루레이에 모든 기타 강의 동영상을 녹화**하기로 했다. 이때, **블루레이의 크기(녹화 가능한 길이)를 최소로** 하려고 한다. 단, M개의 블루레이는 **모두 같은 크기**이어야 한다.
강토의 각 강의의 길이가 분 단위(자연수)로 주어진다. 이때, 가능한 블루레이의 크기 중 **최소를 구하는 프로그램**을 작성하시오.

문제 요약

- 강의 N개, 각각의 길이 존재
- M개의 블루레이로 강의를 순서대로 나눠 담는다
- 각 블루레이에는 연속된 강의만 담아야 함
- 블루레이 용량 최소화

파라매트릭 서치인 이유?

- 원하는 결과가 '최소한의 블루레이 용량'
- 정답이 정수 범위
- 만족 여부 판단 가능 하므로, 이진 탐색 가능
- 블루레이 용량을 기준으로, 가능/불가능 여부를 판단할 수 있음
- compare함수로 구간 줄이기

compare함수의 목적
주어진 블루레이 용량 capacity로 강의를 나눠 몇 개의 블루레이가 필요한지 계산,
그 결과를 기준으로 이 용량이 충분한지 아닌지를 판단하는 함수.

어떤 블루레이 용량을 기준으로 강의를 나눌 때, 블루레이가 몇 개 필요한지를 계산,

문제에서 요구한 블루레이 개수 M보다 많은지 / 적은지 / 같은지를 판단하는 함수

강의는 순서대로만 블루레이에 넣을 수 있고,
➔ 앞에서 끊고 뒤에서 다시 넣는
하나의 블루레이에 들어갈 수 있는 강의들의 합은 capacity 이하여야 한다.
한 블루레이 넘으면 새로운 블루레이 필요

Capacity = 14;
강의들을 왼쪽부터 하나씩 넣으며 누적합 sum 만들고
Sum이 Capacity를 넘는 순간, 새 블루레이를 하나 꺼내서 다시 누적 시작,
이렇게 하면 블루레이 총 몇 개 필요한지 알 수 있고
이걸 계산하면 compare(capacity)결과

용량 기준으로 블루레이 개수 세기

```
int count = 1;  // 블루레이는 최소 1개 필요
int sum = 0;    // 현재 블루레이에 담긴 총합

for (int len : A) {
    if (sum + len > capacity) {
        // 용량 초과 → 새 블루레이 시작
        count++;
        sum = len;
    } else {
        sum += len;
    }
}
```

```
return count - M;
```
이 비교 결과를 통해 파라매트릭 서치의 방향을 결정

Compare(capacity) 결과	의미	탐색 방향
>0	블루레이 부족 -> 용량 작음	용량 늘려야함 mid+1
==0	블루레이 딱 맞음	정답 후보 -> 줄여보기
<0	블루레이 남음 -> 용량 큼	더 줄여도 됨 mid-1

```
public class Main {

    static int[] A;
    static int M;

    public static void main(String[] args) {

        A = new int[]{1, 2, 3, 4, 5, 6, 7, 8, 9};
        M = 3;

        // 다양한 용량으로 compare 함수 테스트
        compare(9);   // 블루레이 용량이 너무 작을 때
        compare(14);  // 블루레이 용량이 딱 맞을 때
        compare(27);  // 블루레이 용량이 클 때
    }

    // 디버깅용 compare 함수
    static int compare(int capacity) {
        int count = 1;  // 블루레이 개수 (최소 1개 필요)
        int sum = 0;    // 현재 블루레이에 담긴 강의 길이 합

        System.out.println("블루레이 용량: " + capacity);
        System.out.print(" 블루레이 " + count + ": ");

        for (int len : A) {
            if (sum + len > capacity) {

                count++;
                sum = len;
                System.out.println();
                System.out.print(" 블루레이 " + count + ": " + len + " ");
            } else {
                sum += len;
                System.out.print(len + " ");
            }
        }

        System.out.println("\n필요한 블루레이 수: " + count);
        System.out.println("compare() 결과: " + (count - M));
        System.out.println("--------------");

        return count - M;
    }
}
```

```
블루레이 용량: 9
 블루레이 1: 1 2 3
 블루레이 2: 4 5
 블루레이 3: 6
 블루레이 4: 7
 블루레이 5: 8
 블루레이 6: 9
필요한 블루레이 수: 6
compare() 결과: 3
--------------
블루레이 용량: 14
 블루레이 1: 1 2 3 4
 블루레이 2: 5 6
 블루레이 3: 7
 블루레이 4: 8
 블루레이 5: 9
필요한 블루레이 수: 5
compare() 결과: 2
--------------
블루레이 용량: 27
 블루레이 1: 1 2 3 4 5 6
 블루레이 2: 7 8 9
필요한 블루레이 수: 2
compare() 결과: -1
--------------
```

Capacity = 9

필요한 블루레이 수: 6

Compare()결과: 6 – 3 = 3

➔ 3개나 더 필요함, 용량이 너무 작음

➔ 이진 탐색에서 left를 늘려 left = mid + 1

Capacity = 14

필요한 블루레이 수: 5

Compare()결과: 5 – 3 = 2

➔ 아직도 더 필요함, 용량이 작음. 더 늘려야 함

➔ 이진 탐색에서 left를 늘려 left = mid + 1

Capacity = 27

필요한 블루레이 수: 2

Compare()결과: 2 – 3 = -1

➔ 용량이 너무 큼

➔ 더 줄여도 가능할 수 있음 right = mid - 1

최종 목표

→ 이런 과정을 반복해서

→ 가능한 용량 중 **블루레이 개수 ≤ M이 되면서 가장 작은 용량**을 찾아내는 것