

BOJ 20181 꿈틀꿈틀 호석 애벌레

- 효율성

Step1 접근

문제

꿈틀꿈틀 호식 애벌레는 N 개의 먹이가 일렬로 나열된 나뭇가지를 오른쪽으로 기어가려고 한다. 시작하는 순간의 호식 애벌레가 0의 위치에 있고 i 번째 먹이는 오른쪽으로 i 초 기어가야 도달할 수 있다. 또한 매초 1 만큼 오른쪽으로 무조건 진행한다.

호식 애벌레는 i 번째 먹이가 맛있을수록 높은 만족도를 얻는다. 호식 애벌레는 절제라는 것을 모르는 욕심쟁이기 때문에 한번 먹이를 먹기 시작하면 연속적으로 계속 먹어야 하며, 누적된 만족도가 최소 만족도 K 이상이 되거나 더 이상 먹을 먹이가 없을 때에 연속적으로 먹는 것을 멈춘다. 만약 최소 만족도 이상이 되면 K 를 초과한 만족도만큼 탈피 에너지를 축적한다. 직후에 호식 애벌레의 만족도는 다시 0 이 되고 먹이를 먹을 수 있게 된다. 나뭇가지를 전부 통과했을 때에 소화를 다 못 했을 경우에도 탈피 에너지는 최소 만족도를 넘기는 순간 이미 축적한 것으로 생각하자.

$N=9, K=6$

1	5	4	4	2	3	10	3	5
---	---	---	---	---	---	----	---	---

1	5	4	4	2	3	10	3	5
---	---	---	---	---	---	----	---	---

만족도: 9

만족도: 15

만족도: 8

5

$$10 - 5 = 5$$

Step1 접근

예를 들어 위와 같이 9개의 먹이가 존재하면, 호석 애벌레는 미래를 도모하여 1번 먹이를 과감하게 포기한다. 그리고 2번부터 먹기 시작해서 3번까지 먹으면 만족도가 9가 되어 3의 에너지를 축적하게 된다. 같은 이유로 4번 먹이도 포기하고 5번부터 먹으면 7번까지 연속으로 먹어서 15의 만족도를 얻는다. 이를 통해 9의 탈피 에너지가 쌓인다. 8, 9번 먹이까지 먹게 되면 2의 탈피 에너지가 축적된다. 이렇게 얻은 총 14의 탈피 에너지가 위의 예제에서는 최대치이다.

매초마다 호석 애벌레는 오른쪽으로 이동하면서 먹이를 지나치거나 먹기 시작할 수 있다. 먹기 시작하면 만족도가 채워질때까지 먹게 될것이다. 어떤 먹이들을 대해 먹어야 축적된 탈피 에너지가 최대가 될 수 있을까?

입력

첫번째 줄에 먹이 개수 N , 최소 만족도 K 가 공백으로 주어진다.

두번째 줄에는 1 번부터 N 번 먹이의 만족도가 순서대로 주어진다.

출력

축적된 탈피 에너지의 최댓값을 구하라. 만약 탈피를 한 번도 할 수 없다면 0을 출력한다.

Step1 접근

일단 알고리즘 보고 시작(。・∀・)ﾉ ✕

- ~~★~~ Dynamic Programming
- ~~Binary Search~~ }
- ~~★~~ Prefix Sum
- ~~★~~ Two-pointer

원래 내가 의도한 건 투포인터 하나 뿐...

Step1 접근 - DP

계단 오르기 문제 등등

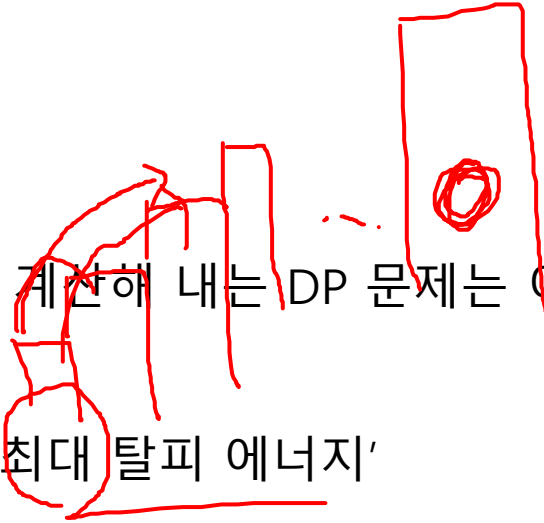
특정 위치까지 왔을 때 얻을 수 있는 최대 이득을 계산해 내는 DP 문제는 여러 개 풀어봤음!

문제에서 구하고자 하는 것은 결과적으로
'마지막 먹이 위치'에 도달했을 때 얻을 수 있는 '최대 탈피 에너지'

따라서 ✕

dp[k]는 **k위치까지 왔을 때 얻을 수 있는 최대 탈피 에너지**를 의미

결국 문제의 정답은 dp[N]에 저장될 것임



Step1 접근 – Prefix Sum

문제에서는 누적 만족도가 최소 만족도를 넘어설 때 그 차 만큼 탈피 에너지를 축적한다고 했음

누적!

해야되니까 prefix sum~~

따라서

먹이를 먹는 구간에 대한 누적 만족도를 저장하는 변수를 하나 만들자

그것이 바로 satis

Step1 접근 – Two Pointer

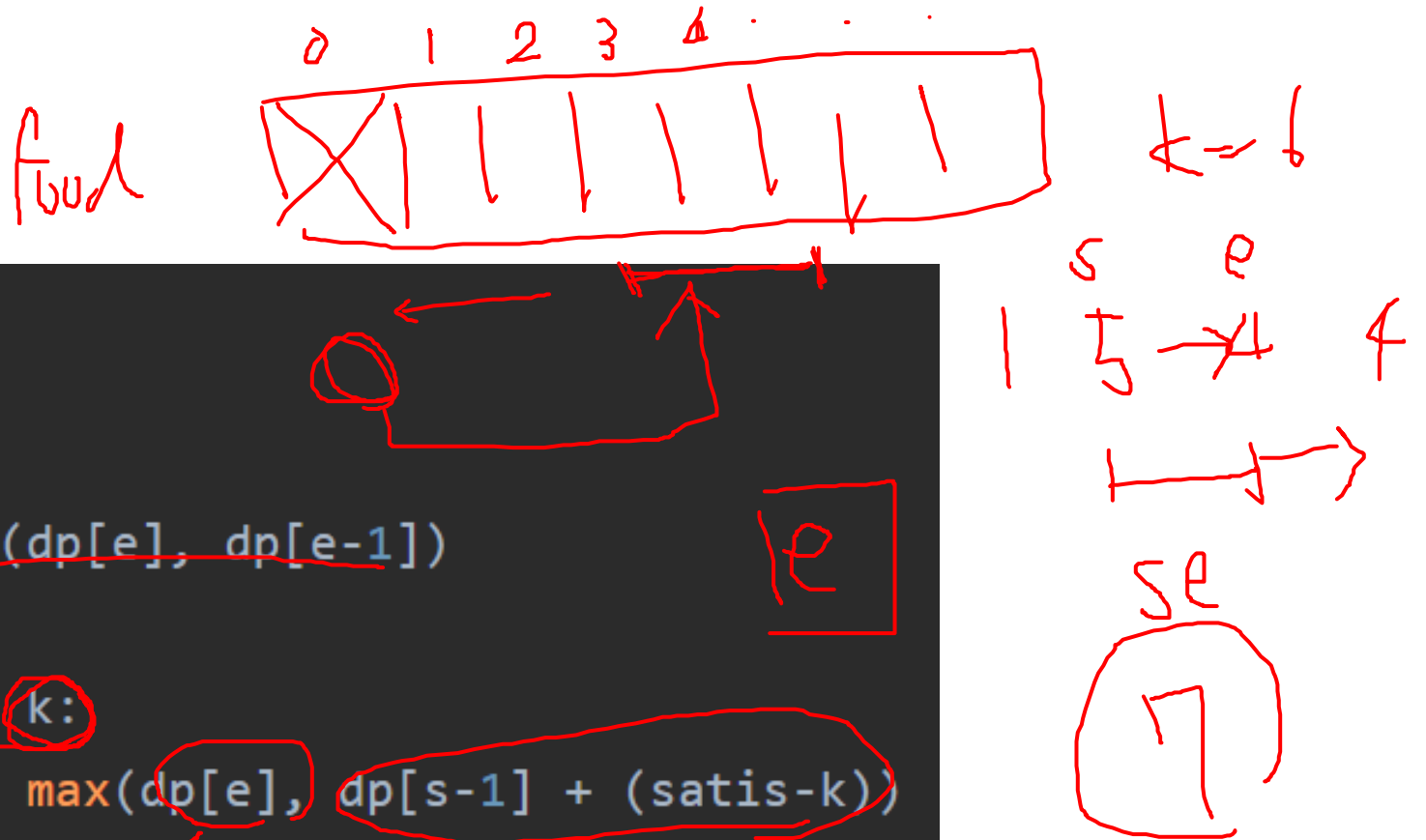


호식 애벌레는 욕심이 많아서
먹이를 한번 먹으면 누적 만족도가 최소 만족도 k 이상이 될 때까지 먹음

따라서
먹이는 먹는 시작과 끝이 존재하게 됨

먹이는 처음 (누적 만족도가 0일 때) 먹는 위치가 s
누적 만족도 s 가 최소 만족도 k 이상이 되어서
멈추는 위치가 e

Step2 구현



```
s, e = 1, 1
while True:

    dp[e] = max(dp[e], dp[e-1])

    if satis >= k:
        dp[e] = max(dp[e], dp[s-1] + (satis-k))

        if s < e:
            satis -= food[s]
        elif s == e:
            e += 1
```



```
if e > n:
```

```
    break
```

```
    satis = food[e]
```

```
    s += 1 //
```

```
else:
```

```
    e += 1
```

```
if e > n:
```

```
    break
```

```
satis += food[e]
```

```
print(dp[-1])
```

dp[n]

Step2 구현




입력 룰루~

```
n, k = map(int, stdin.readline().split())
food = [0]
food.extend([*map(int, stdin.readline().split())])

dp = [0] * (n+1)
satis = food[1]
```

Step3 결과

아이디	문제	결과	메모리	시간	언어	코드 길이
january	 20181	맞았습니다!! (66/66)	144584 KB	168 ms	PyPy3 / 수정	569 B

이걸해냄...!!!