1로 만들기

$dp(N)$ = N을 1로 만드는 최소 경우의 수

$$dp(N//3)+1$$
$$dp(N//2)+1$$
$$dp(N-1)+1$$

최소?

$$\therefore\ dp(N) = \min(dp(N//3)+1, dp(N//2)+1, dp(N-1)+1)$$

쉬운 계단 수

N=1

1
2
3
4
5
6
7
8
9

N=2

12	21	10
23	32	
34	43	
45	54	
56	65	
67	76	
78	87	
89	98	

num

i-1	i	i+1
1	2	3
	0	1
		⋮
2	3	
	1	
3	4	
	2	
4	⋮	
⋮		
9		

$num[i] = num[i-1]+1$

or

$num[i-1]-1$

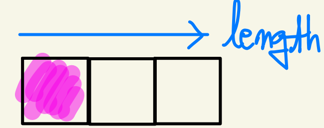
$\boxed{9 \rightarrow 8}$

else → 2개씩.

1~8 → 2개 → 2개
1~8

9 → 8 → 7
9

쉬운 계단 수 N=1



0 1 2 3 4 5 6 7 8 9

dp[length][맨앞숫자]

N=2

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

경우의 수 = 2 경우의 수 = 1

맨앞 숫자 "0"

→ 문제 조건 X

맨앞 숫자 "1~8"

→ +1 or -1

경우의 수 2

N=3

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

경우의 수 2 경우의 수=1

맨앞 숫자 "9"

→ "8"

경우의 수 1

쉬운 계단 수 N=1

0 1 2 3 4 5 6 7 8 9

경우의 수=1

N=2

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

경우의 수 =2 경우의 수 = 1

N=3

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

경우의 수 2 경우의 수=1

ex) N=3, 맨 앞 숫자=4

→ N=2, 맨 앞 숫자=3일때

맨 앞숫자=5일때

경우의 수

i) 맨앞 숫자 = 1~8

dp[length][맨앞숫자]

= dp[length-1][맨앞숫자+1]

+ dp[length-1][맨앞숫자-1]

ii) 맨앞 숫자 = 9

dp[length][9]

= dp[length-1][8]

iii) 맨 앞 숫자=0

dp[length][0] = dp[length-1][1]

포도주 시식

틀린 풀이

"계단 오르기" 문제와 비슷함. 연속 3계단 금지 ~~& 막계단 밟기~~

이것만 사라짐.

잔: 마지막 잔 위치

ex) dp[5] = 5번 잔을 마지막으로 마실 때 포도주의 최대양

최대한 많은 양의 포도주를 마시기 위해서는 1잔씩 이동하며 전부 다 마시는 것이 좋지만 "연속 3잔 금지"라는 조건이 있기에 1잔 아 2잔 이동을 적절히 섞어서 마시는 것이 관건.

∴ "계단 오르기" 처럼 풀되 dp 리스트의 최대값을 출력한다.

포도주 시식

올바른 풀이

틀린 풀이에서의 문제점은 n잔일 때 최대 양이 dp[n]이 아니라는 점이다.

ex) $n=6 \rightarrow dp = [6, 16, 23, 28, 33, 32]$

$dp[6] = 32 \neq 33$ → 추후 dp[i]와 dp[j]를 비교하는 코드에서

오류 발생 가능성 O

실제 반례 : $n=9 \rightarrow dp = [6, 16, 23, 28, 33, 32, 34, 36, 38]$

만약 33으로 저장되었다면

39이고

이 배열의 max는

39가 되었을 것.

가장 긴 증가하는 부분 수열

$A = \{10, 20, 10, 30, 20, 50\}$ 4

① ② ③

dp[i] : i번째 위치한 원소에서 제일 긴 LIS 길이

※ n번째 위치한 원소에서 제일 긴 LIS 길이가 (dp[n]) dp[n-1]보다 작을 수도 있음.

$30 > ① \Rightarrow dp[4] = dp[1]+1 \cdots \mathrm{I}$

$30 > ② \Rightarrow dp[4] = \max(\mathrm{I}, dp[2]+1) \cdots \mathrm{II}$

$30 > ③ \Rightarrow dp[4] = \max(\mathrm{II}, dp[3]+1)$

$\therefore$ if $arr[i] > arr[j]$:

$$dp[i] = \max(dp[i], dp[j]+1)$$

가장 긴 바이토닉 부분 수열

바이토닉 = 증가 + 감소 - 1

∵ 기준 숫자가 중복되므로

ex)	1	5	2	1	4	3	4	5	2	1	
증가	v		v			v	v	v			5
감소								v	v	v	3

$\downarrow$

-1

가장 긴 바이토닉 부분 수열

$A = \{10, 20, 10, 30, 20, 50\}$

$A' = \{50, 20, 30, 10, 20, 10\}$

① ② ③

dp[i] : i번째 위치한 원소에서 제일 긴 LDS 길이

※ n번째 위치한 원소에서 제일 긴 LDS 길이가 (dp[n]) dp[n-1] 보다 작을 수도 있음.

$10 > ① \Rightarrow dp[4] = dp[1] + 1 \cdots Ⅰ$

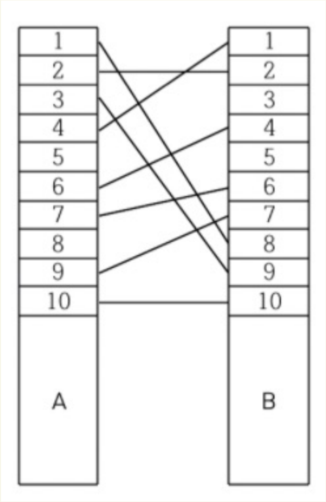
$10 > ② \Rightarrow dp[4] = max(Ⅰ, dp[2] + 1) \cdots Ⅱ$

$10 > ③ \Rightarrow dp[4] = max(Ⅱ, dp[3] + 1)$

∴ if arr[i] > arr[j] :

dp[i] = max(dp[i], dp[j]+1)

전깃줄



1 → 8 , 2는 9이상이어야 한다.

2 → 2 , 3은 3이상이어야 한다. 1~i-1를 다 만족해야함.

⋮

i → N , i+1은 N+1이상이어야 한다. → 유효 조건

dp = [1, 1,

" LIS "

dp[i] = i번 지점에서 최대로 연결할 수 있는 줄의 개수.

```
dp[1] = 1
    if 유효 조건:
        dp[2] = dp[1]+1
    else:
        dp[2] = dp[1]
```

LCS

$$string1 = [ACAYKP] \quad string2 = [CAPCAK] \Rightarrow \begin{matrix} A \\ C \end{matrix} \Rightarrow \begin{matrix} A \\ CA \end{matrix} \Rightarrow \begin{matrix} A \\ CAP \end{matrix} \cdots \Rightarrow \begin{matrix} AC \\ C \end{matrix} \Rightarrow \begin{matrix} AC \\ CA \end{matrix} \cdots$$

$$matrix = [[ACAYKP], [CAPCAK]]$$

if string1[-1] == string2[-1] :

  matrix[i][j] = matrix[i-1][j-1] + 1

$$\left\{\begin{matrix} AC \\ CAPC \end{matrix}\right\} LCS = \left\{\begin{matrix} A \\ CAP \end{matrix}\right\} LCS + 1$$

else :

  matrix[i][j] = max(matrix[i][j-1], matrix[i-1][j])

$$\left\{\begin{matrix} ACAYKP \\ CAPCAK \end{matrix}\right\} LCS = \underbrace{\left\{\begin{matrix} ACAYKP \\ CAPCA \end{matrix}\right\} LCS, \left\{\begin{matrix} ACAYK \\ CAPCAK \end{matrix}\right\} LCS}_{\text{더 큰거}}$$

연속합

10, -4, 3, 1, 5, 6, -35, 12, 21, -1

index

0 → 음수 : Sum

음수 → 음수 : Sum

음수 → 음수 : Sum

⋮

List 끝까지

이 중에서 제일 큰 값

nums_minus = [-4, -35, -1]