Chapter. 08

동적 프로그래밍

기초 문제풀이

FAST CAMPUS
ONLINE
유형별 문제풀이

강사. 나동빈

Chapter. 08

동적 프로그래밍(기초 문제풀이)

FAST CAMPUS
ONLINE
나동빈 강사.

I 혼자 힘으로 풀어 보기

01타일: https://www.acmicpc.net/problem/1904

문제 제목: 01타일

문제 난이도: 하(Easy)

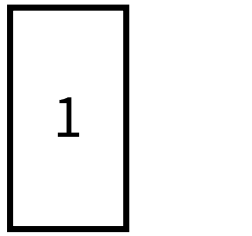
문제 유형: 동적 프로그래밍

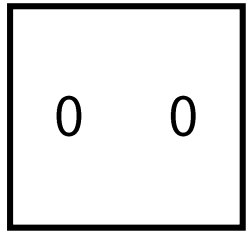
추천 풀이 시간: 20분

01타일: https://www.acmicpc.net/problem/1904

I 문제 풀이 핵심 아이디어

- 사용할 수 있는 타일의 종류는 2개입니다.
- 두 가지 종류의 타일을 이용하여, N 길이의 수열을 만드는 모든 경우의 수를 구해야 합니다.
- 가장 전형적인 동적 프로그래밍 문제입니다.
- N이 최대 1,000,000이므로 시간 복잡도 O(N)으로 해결해야 합니다.

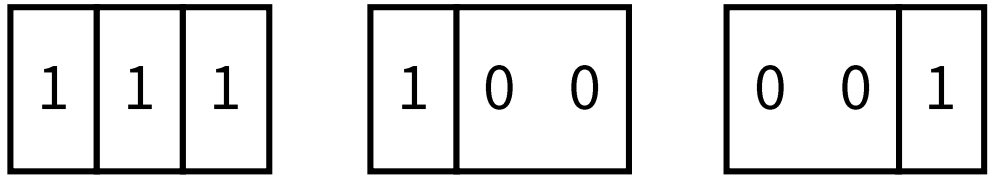




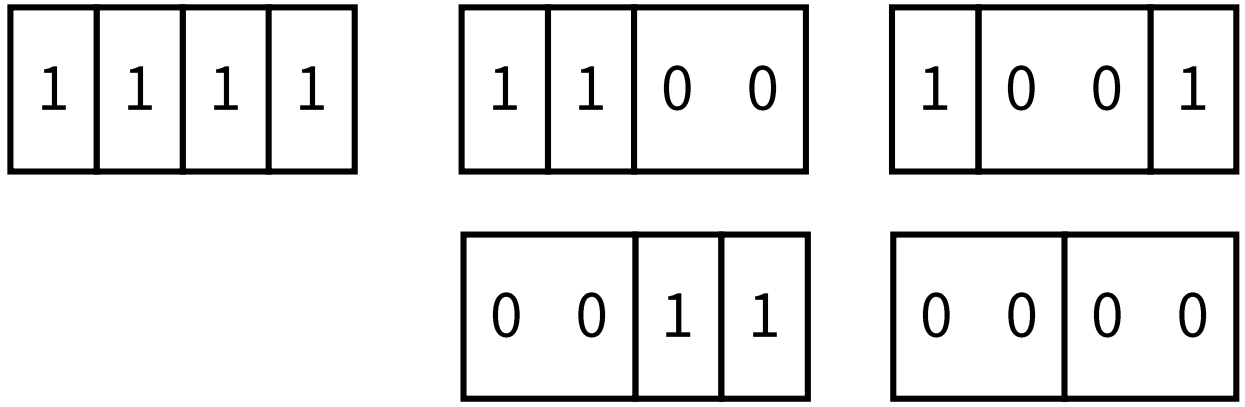
01타일: https://www.acmicpc.net/problem/1904

I 문제 풀이 핵심 아이디어

- 타일을 왼쪽에서부터 오른쪽으로 이어서 붙인다고 가정합니다.
- 예를 들어 N = 3일 때, 경우의 수는 3가지입니다.



- 예를 들어 N = 4일 때, 경우의 수는 5가지입니다.



01타일: https://www.acmicpc.net/problem/1904

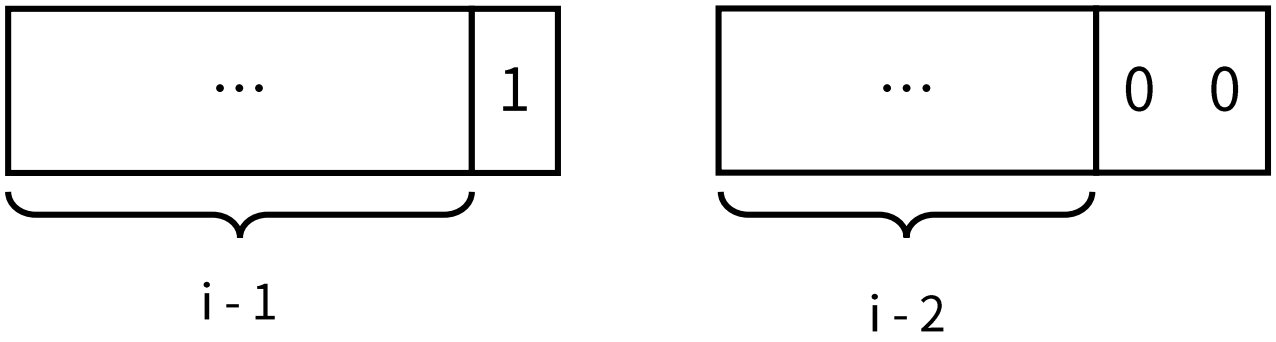
I 문제 풀이 핵심 아이디어

- 동적 프로그래밍 문제를 풀기 위해서는 **점화식**(인접한 항들 사이의 관계식)을 세워야 합니다.
- D[i] = '수열의 길이가 i일 때의 경우의 수'라고 합시다.
- 예를 들어 D[3] = 3, D[4] = 5 입니다.

01타일: https://www.acmicpc.net/problem/1904

I 문제 풀이 핵심 아이디어

- 타일을 왼쪽에서부터 오른쪽으로 이어서 붙인다고 가정합니다.
- 생각해 보면, 길이가 i인 수열을 형성하는 방법은 다음의 두 가지 뿐입니다.



- 따라서 D[i] = '수열의 길이가 i일 때의 경우의 수'라고 하면,
- $D[i] = D[i - 1] + D[i - 2]$
- 다시 말해 이 문제는 피보나치 수열과 동일한 문제입니다.

01타일: https://www.acmicpc.net/problem/1904

소스코드

```python
n = int(input())

dp = [0] * 1000001
dp[1] = 1
dp[2] = 2

for i in range(3, n + 1):
    dp[i] = (dp[i - 2] + dp[i - 1]) % 15746

print(dp[n])
```

I 혼자 힘으로 풀어 보기

평범한 배낭: https://www.acmicpc.net/problem/12865

문제 제목: 평범한 배낭

문제 난이도: 하(Easy)

문제 유형: 동적 프로그래밍

추천 풀이 시간: 30분

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- 배낭 문제(Knapsack Problem)으로도 알려져 있는, 전형적인 동적 프로그래밍 문제입니다.
- 물품의 수가 N, 배낭에 담을 수 있는 무게가 K입니다.
- 동적 프로그래밍을 이용하여 시간 복잡도 O(NK)로 문제를 해결할 수 있습니다.

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- **핵심 아이디어**: 모든 무게에 대하여 최대 가치를 저장하기
- D[i][j] = 배낭에 넣은 물품의 무게 합이 j일 때 얻을 수 있는 최대 가치
- 각 물품의 번호 i에 따라서 최대 가치 테이블 D[i][j]를 갱신하여 문제를 해결할 수 있습니다.

- $$D[i][j] = \begin{cases} D[i-1][j] & if \quad j < W[i] \\ \max\left(D[i-1][j], D[i-1][j-W[i]] + V[i]\right) & if \quad j \geq W[i] \end{cases}$$

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- 예를 들어 N = 4, K = 7일 때의 예시를 확인해 봅시다.

무게	가치
6	13
4	8
3	6
5	12

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
0							
0							
0							
0							

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- 예를 들어 N = 4, K = 7일 때의 예시를 확인해 봅시다.

무게	가치
6	13
4	8
3	6
5	12

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
0	0	0	0	0	0	13	13
0							
0							
0							

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- 예를 들어 N = 4, K = 7일 때의 예시를 확인해 봅시다.

무게	가치
6	13
4	8
3	6
5	12

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
0	0	0	0	0	0	13	13
0	0	0	0	8	8	13	13
0							
0							

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- 예를 들어 N = 4, K = 7일 때의 예시를 확인해 봅시다.

무게	가치
6	13
4	8
3	6
5	12

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
0	0	0	0	0	0	13	13
0	0	0	0	8	8	13	13
0	0	0	6	8	8	13	14
0							

평범한 배낭: https://www.acmicpc.net/problem/12865

I 문제 풀이 핵심 아이디어

- 예를 들어 N = 4, K = 7일 때의 예시를 확인해 봅시다.

무게	가치
6	13
4	8
3	6
5	12

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
0	0	0	0	0	0	13	13
0	0	0	0	8	8	13	13
0	0	0	6	8	8	13	14
0	0	0	6	8	12	13	14

평범한 배낭: https://www.acmicpc.net/problem/12865

소스코드

```python
n, k = map(int, input().split())
dp = [[0] * (k + 1) for _ in range(n + 1)]

for i in range(1, n + 1):
    weight, value = map(int, input().split())
    for j in range(1, k + 1):
        if j < weight:
            dp[i][j] = dp[i - 1][j]
        else:
            dp[i][j] = max(dp[i - 1][j], dp[i - 1][j - weight] + value)

print(dp[n][k])
```

I 혼자 힘으로 풀어 보기

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

문제 제목: 가장 긴 증가하는 부분 수열

문제 난이도: 하(Easy)

문제 유형: 동적 프로그래밍, LIS

추천 풀이 시간: 30분

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

I 문제 풀이 핵심 아이디어

- 가장 긴 증가하는 부분 수열(LIS) 문제는, 전형적인 동적 프로그래밍 문제입니다.
- 수열의 크기가 N일 때, 기본적인 동적 프로그래밍 알고리즘으로 $O(N^2)$에 해결할 수 있습니다.

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

문제 풀이 핵심 아이디어

- $D[i] = array[i]$를 마지막 원소로 가지는 부분 수열의 최대 길이
- 모든 $0 \leq j < i$ 에 대하여, $D[i] = \max(D[i], D[j] + 1)$ if $array[j] < array[i]$

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

문제 풀이 핵심 아이디어

- $N = 6$일 때, 예시 수열에 대하여 다음과 같이 계산할 수 있습니다.

	10	20	10	30	20	50
[초기 상태]	1	1	1	1	1	1

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

l 문제 풀이 핵심 아이디어

- $N = 6$일 때, 예시 수열에 대하여 다음과 같이 계산할 수 있습니다.

	10	20	10	30	20	50
[초기 상태]	1	1	1	1	1	1
i = 1	1	2	1	1	1	1

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

I 문제 풀이 핵심 아이디어

- $N = 6$일 때, 예시 수열에 대하여 다음과 같이 계산할 수 있습니다.

	10	20	10	30	20	50
[초기 상태]	1	1	1	1	1	1
i = 1	1	2	1	1	1	1
i = 2	1	2	1	1	1	1

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

l 문제 풀이 핵심 아이디어

- $N = 6$일 때, 예시 수열에 대하여 다음과 같이 계산할 수 있습니다.

	10	20	10	30	20	50
[초기 상태]	1	1	1	1	1	1
i = 1	1	2	1	1	1	1
i = 2	1	2	1	1	1	1
i = 3	1	2	1	3	1	1

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

l 문제 풀이 핵심 아이디어

- $N = 6$일 때, 예시 수열에 대하여 다음과 같이 계산할 수 있습니다.

	10	20	10	30	20	50
[초기 상태]	1	1	1	1	1	1
i = 1	1	2	1	1	1	1
i = 2	1	2	1	1	1	1
i = 3	1	2	1	3	1	1
i = 4	1	2	1	3	2	1

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

l 문제 풀이 핵심 아이디어

- $N = 6$일 때, 예시 수열에 대하여 다음과 같이 계산할 수 있습니다.

	10	20	10	30	20	50
[초기 상태]	1	1	1	1	1	1
i = 1	1	2	1	1	1	1
i = 2	1	2	1	1	1	1
i = 3	1	2	1	3	1	1
i = 4	1	2	1	3	2	1
i = 5	1	2	1	3	2	4

가장 긴 증가하는 부분 수열: https://www.acmicpc.net/problem/11053

소스코드

```python
n = int(input())
array = list(map(int, input().split()))
dp = [1] * n

for i in range(1, n):
    for j in range(0, i):
        if array[j] < array[i]:
            dp[i] = max(dp[i], dp[j] + 1)

print(max(dp))
```