알튜비튜
그리디

오늘은 '탐욕법'이라고도 불리는 그리디 알고리즘에 대해 배울 것입니다.
가장 직관적인 알고리즘 중 하나죠.

그리디 특징

그리디

- 욕심쟁이 방법, 탐욕적 방법, 탐욕 알고리즘 등으로 불리기도 한다.
- 우리가 원하는 답을 여러 개의 조각으로 쪼개고, 각 단계마다 답의 한 부분을 만들어간다.
- 모든 단계의 선택지를 고려해보지는 않는다.
- 각 단계마다 지금 당장 가장 좋은 방법을 선택 (근시안적인 선택)
- 계산속도가 빠르다는 장점

-> 많은 경우 최적해를 찾지 못한다. 특정 조건 하에서만 적용 가능

그리디 알고리즘의 사용 조건

- 시간적, 공간적 제약으로 최적해를 구하지 못해, 근사해를 구해야 하는 문제
  - 대개 출제되지 않는다.

- 탐욕법을 사용해도 항상 최적해를 구할 수 있는 문제
  - 탐욕 선택 속성: 탐욕적으로 선택하더라도 문제의 최적해가 보장될 때 (손해X)
  - 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있을 때

  ⇒ 알고리즘의 정당성을 증명하는 과정을 연습

#알튜비튜

/<> 11047번 : 동전 0 – Silver 4

문제

- 가지고 있는 동전: 총 N종류
- 각각의 동전을 매우 많이 가지고 있다 → 동전이 부족하지 않다
- 동전을 적절히 사용해 가치의 합이 K가 되게 한다
- 동전 개수의 최솟값을 구하는 문제

제한 사항

- n의 범위는 $1 \le n \le 10$
- K의 범위는 $1 \le K \le 100,000,000$
- 동전의 가치(오름차순) $1 \le A_i \le 1,000,000$
- $A_1 = 1$, $i \ge 2$인 경우에 A_i 는 A_{i-1} 의 (양의) 배수 → 모든 K값을 만들 수 있다.

/<> 11047번 : 동전 0 – Silver 4

알고리즘의 정당성 증명

- 탐욕 선택 속성: 탐욕적으로 선택하더라도 문제의 최적해가 보장되어야 한다
  ➔ 큰 값의 동전을 이용하는 것이 최적해를 보장한다.

  ➔ 최적해 = 동전 개수의 최솟값

  e.g. 100원 짜리와 500원짜리 동전 2가지만 있는 상황에서 500원을 표현해야 한다
  → 500원짜리 한 개로 나타내는 것이 100원짜리 5개로 나타내는 것보다 적은 개수의 동전을 사용한다 (필요한 동전의 개수는 금액에 반비례)

#알튜비튜

/<> 11047번 : 동전 0 – Silver 4

알고리즘의 정당성 증명

- 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있어야 한다
  ➔ 큰 가치의 동전부터 최대한 이용하는 것이 전체 동전 개수의 최소화로 확장된다.

  e.g. 2700원을 1000원, 500원, 100원, 50원짜리 동전으로 나타내자
  2700원 = 1000원*2 + 700원 ➔ 1000원짜리를 덜 사용한다면 필요한 동전 개수가 늘어난다.

11047번 : 동전 0 – Silver 4

알고리즘의 정당성 증명

- 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있어야 한다
  ➔ 큰 가치의 동전부터 최대한 이용하는 것이 전체 동전 개수의 최소화로 확장된다.

  e.g. 2700원을 1000원, 500원, 100원, 50원짜리 동전으로 나타내자
  2700원 = 1000원*2 + 700원 ➔ 1000원짜리를 덜 사용한다면 필요한 동전 개수가 늘어난다.
  = 1000원*2 + 500원*1 + 200원

#알튜비튜

11047번 : 동전 0 – Silver 4

알고리즘의 정당성 증명

- 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있어야 한다
  ➔ 큰 가치의 동전부터 최대한 이용하는 것이 전체 동전 개수의 최소화로 확장된다.

  e.g. 2700원을 1000원, 500원, 100원, 50원짜리 동전으로 나타내자
  2700원 = 1000원*2 + 700원 ➔ 1000원짜리를 덜 사용한다면 필요한 동전 개수가 늘어난다.
  = 1000원*2 + 500원*1 + 200원
  = 1000원*2 + 500원*1 + 100원*2 ➔ 100원짜리를 덜 사용한다면 필요한 동전 개수가 늘어난다.

⇒ 탐욕 선택 속성과 최적 부분 구조를 만족하므로, 그리디 알고리즘을 이용할 수 있다.

/<> 11047번 : 동전 0 – Silver 4

직관적 이해 (증명할 시간이 부족한 경우)

- 각 구조가 규칙성을 보이며 반복되는 경우 의심해보자

k = 1200 , 보유한 동전이 1원, 100원, 500원일 때

500원짜리 1개를 덜 사용하면 100원짜리 5개를 사용해야 한다 → 4개씩 동전 개수 증가

<u>큰 가치의 동전을 많이 사용할수록 동전의 총 개수가 줄어든다!</u>

예제

#알튜비튜

예제 입력1

```
10 4200
1
5
10
50
100
500
1000
5000
10000
50000
```

예제 출력1

```
6
```

4200 = 1000*4 + 100*2

11047번 : 동전 0 – Silver 4

(Optional) 만약 동전끼리 배수가 아니라면?

- 우리나라에 160원 동전이 추가된 경우를 생각해보자.
- 거스름돈이 200원이라면

우리의 알고리즘은 160원 동전 1개와 10원 동전 4개

총 5개를 return한다.

그러나 최적해는 100원 동전 2개이다.

1931번: 회의실 배정 – Silver 1

문제

- 한 개의 회의실, 이를 사용하고자 하는 N개의 회의 존재
- 각 회의의 시작 시간, 끝나는 시간 주어짐
- 회의끼리는 겹쳐서는 안 되며, 중간에 중단될 수 없다. (끝나자마자 바로 시작하는 건 가능)
- 회의실을 사용할 수 있는 회의의 최대 개수 구하는 문제

제한 사항

- n의 범위는 1 <= n <= 100,000
- 시작 시간 & 끝나는 시간은 2^{31}-1보다 작거나 같은 자연수 또는 0
  → int 범위 내에 있다.

1931번: 회의실 배정 – Silver 1

알고리즘의 정당성 증명

- 탐욕 선택 속성: 탐욕적으로 선택하더라도 문제의 최적해가 보장되어야 한다
  - 가장 빨리 끝나는 회의를 선택하면, 다른 회의에 사용할 수 있는 남은 시간이 커진다.

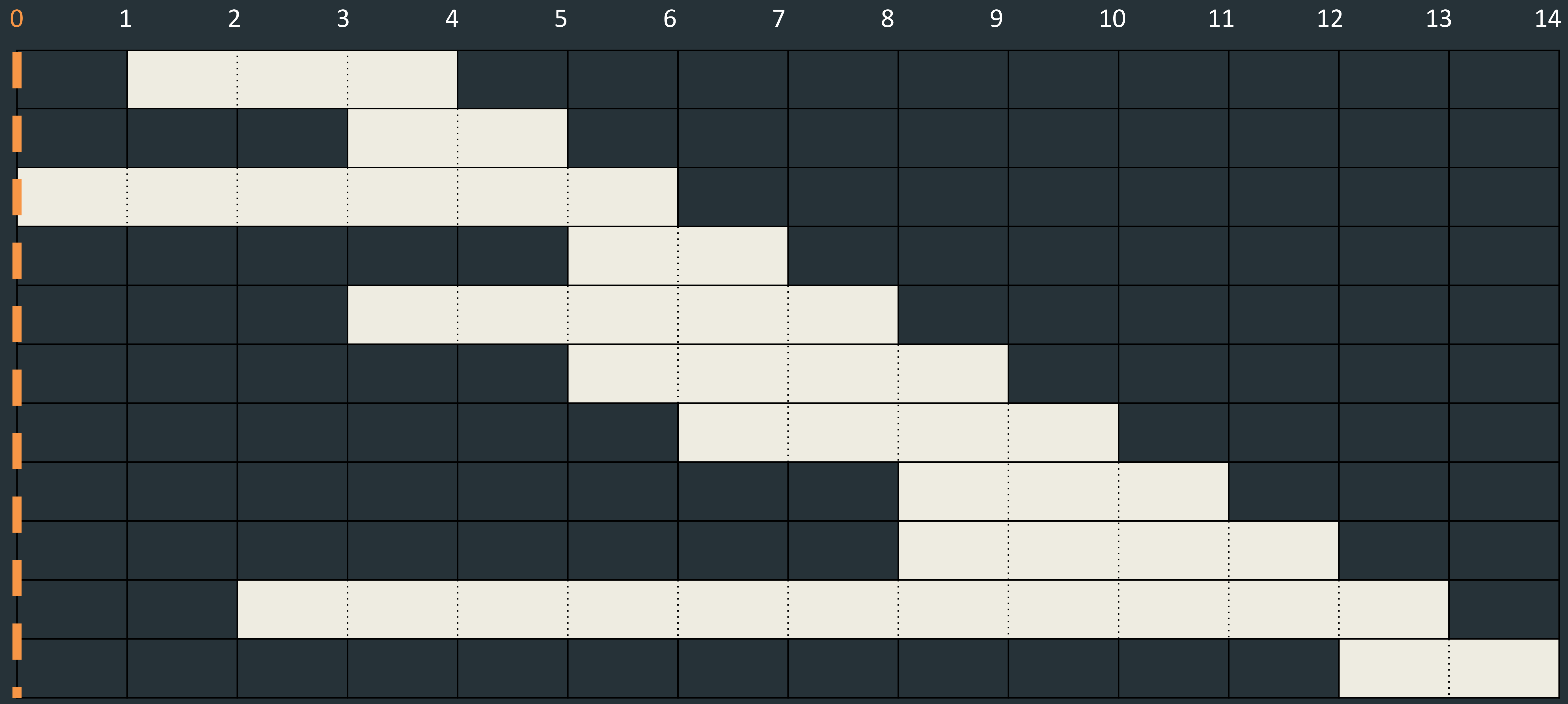
- 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있을 때
  - 매번 가장 빨리 끝나는 회의를 선택한다
  - 이후, 남은 회의 시간에 대하여 똑같이 가장 빨리 끝나는 회의를 선택할 수 있다.

⇒ 그리디 알고리즘을 이용할 수 있다.

예제

예제 입력1

```
11
1 4
3 5
0 6
5 7
3 8
5 9
6 10
8 11
8 12
2 13
12 14
```



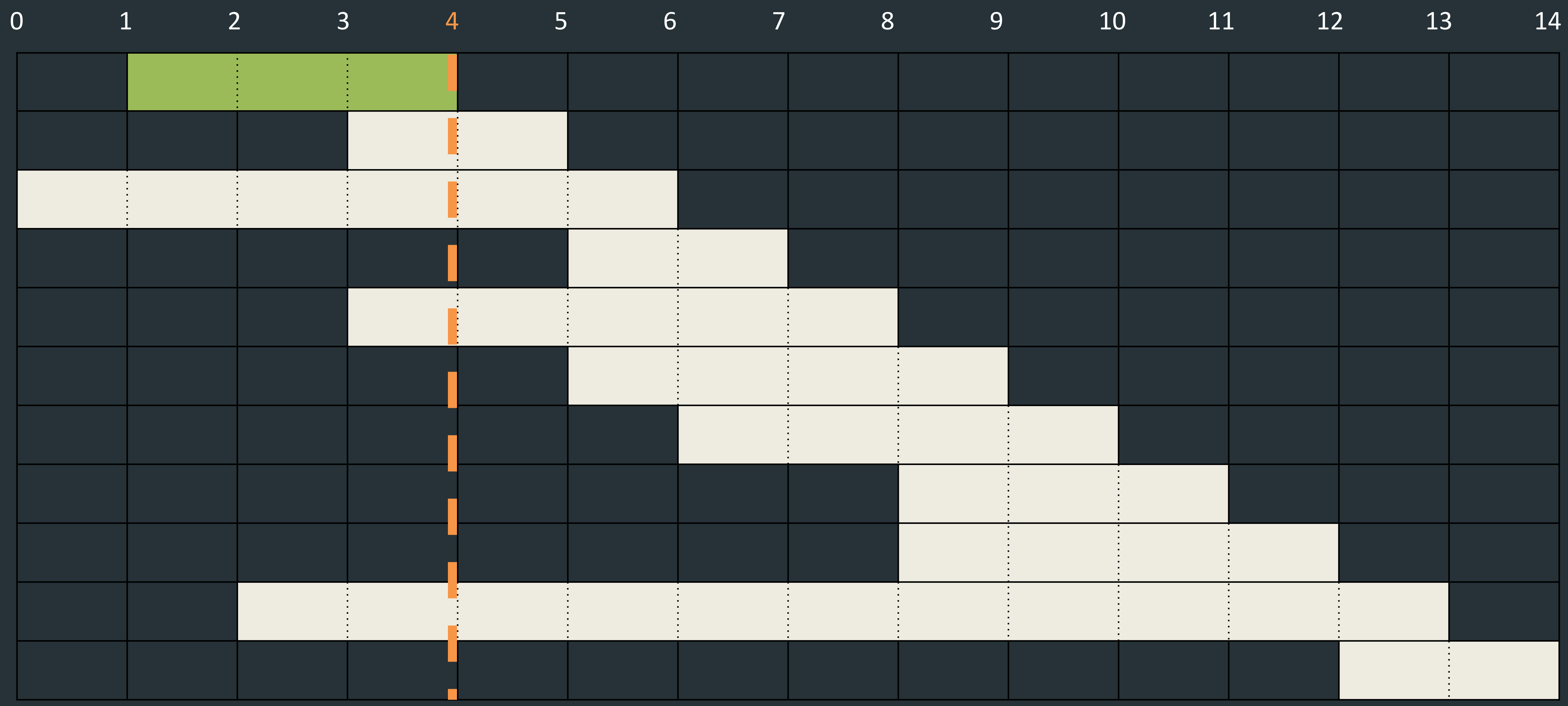
예제 출력1

```
4
```

예제

예제 입력1

```
11
1 4
3 5
0 6
5 7
3 8
5 9
6 10
8 11
8 12
2 13
12 14
```



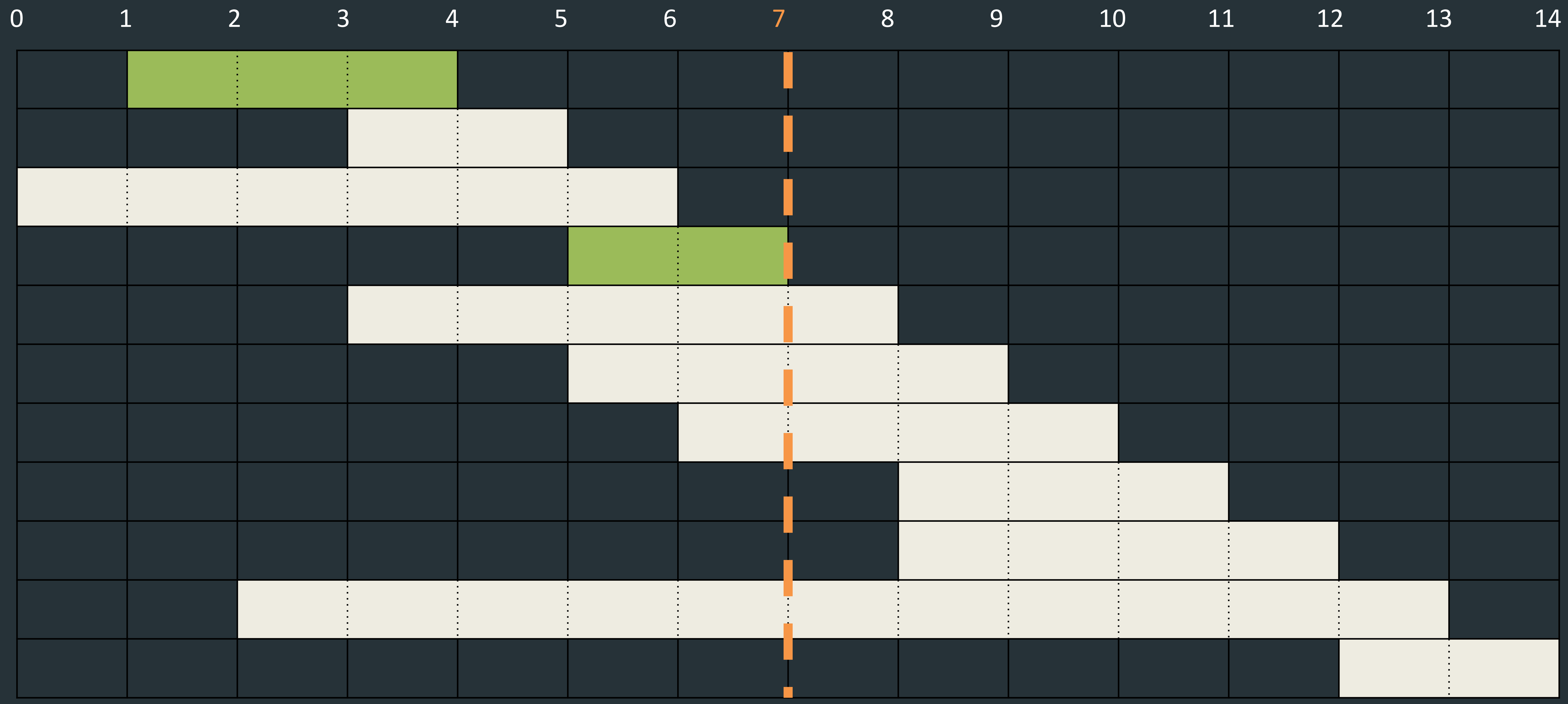
예제 출력1

```
4
```

예제

예제 입력1

```
11
1 4
3 5
0 6
5 7
3 8
5 9
6 10
8 11
8 12
2 13
12 14
```



예제 출력1

```
4
```

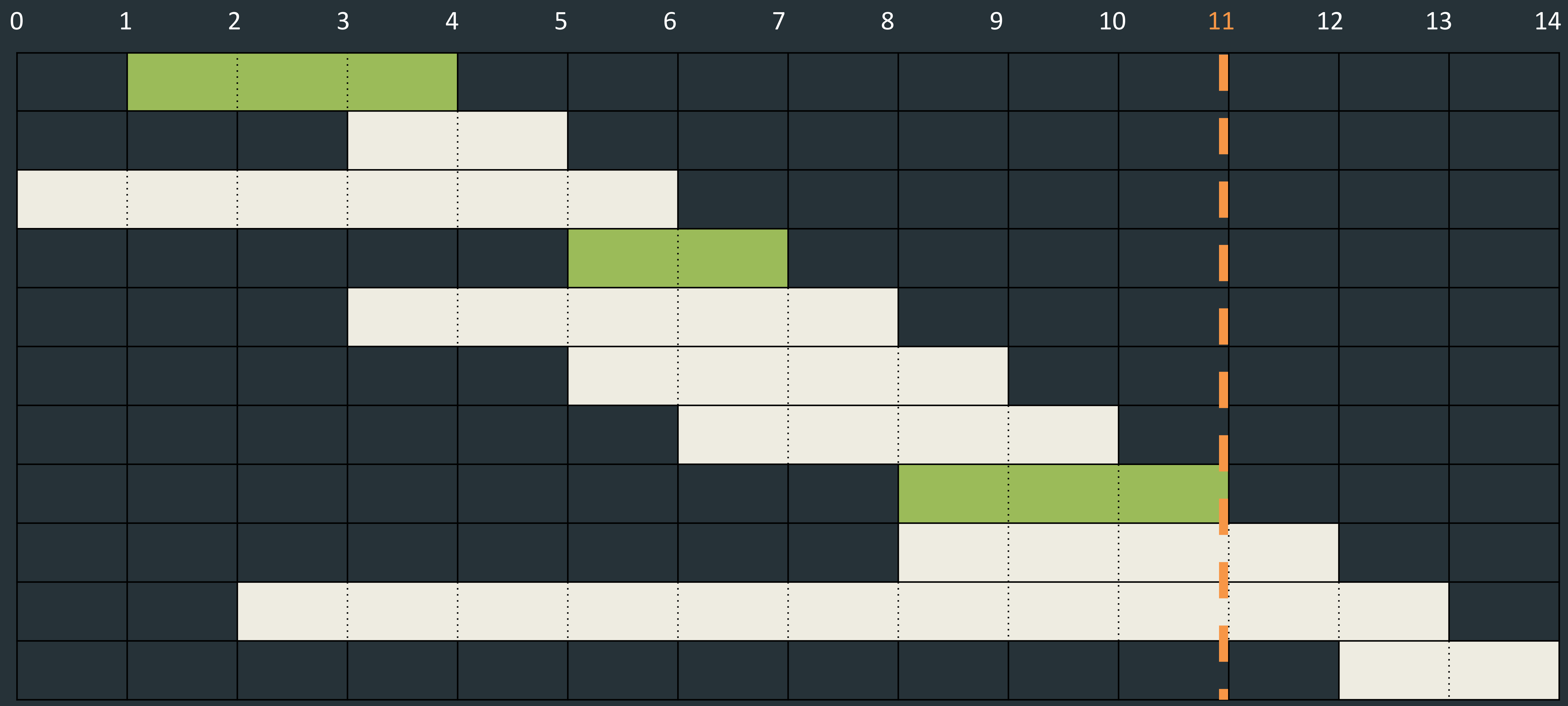
예제

예제 입력1

```
11
1 4
3 5
0 6
5 7
3 8
5 9
6 10
8 11
8 12
2 13
12 14
```

예제 출력1

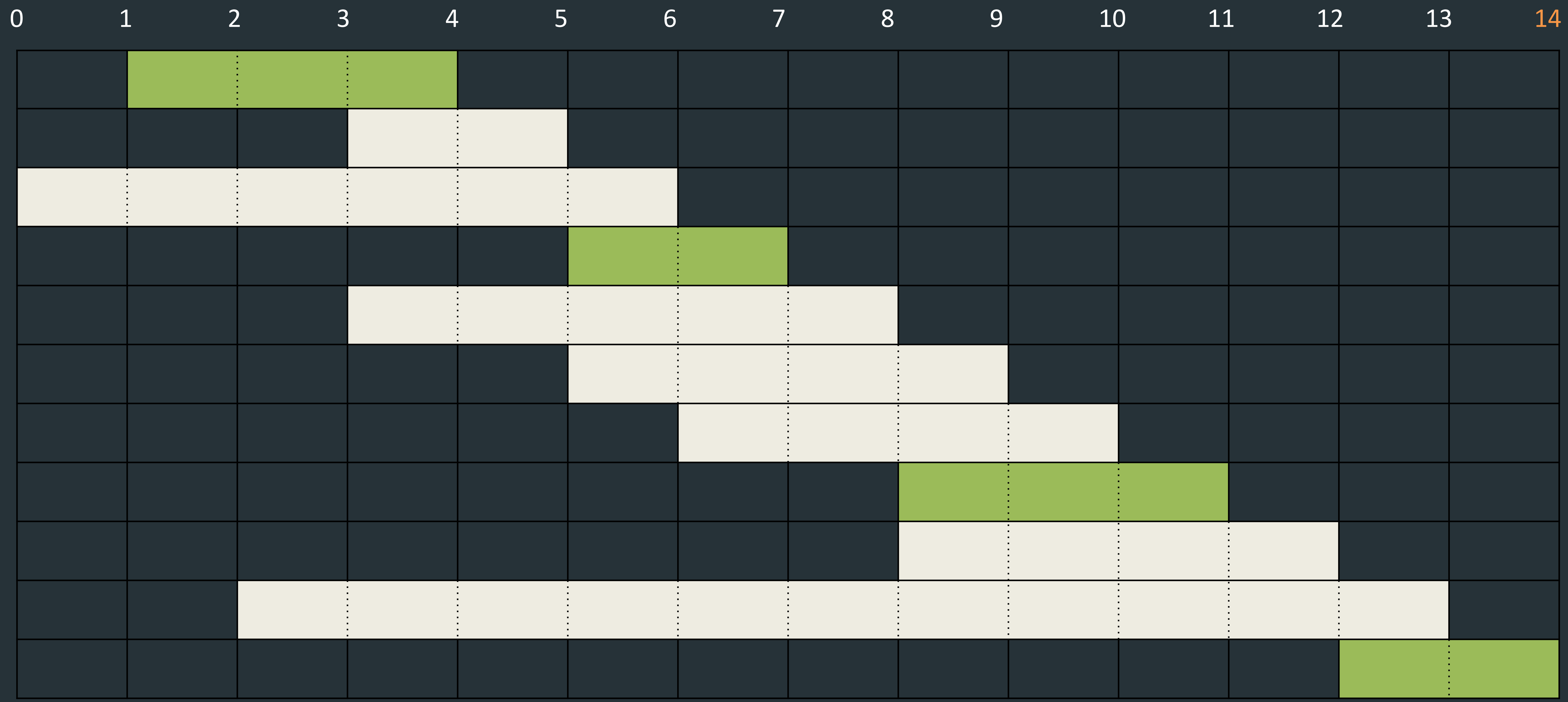
```
4
```



예제

예제 입력1

```
11
1 4
3 5
0 6
5 7
3 8
5 9
6 10
8 11
8 12
2 13
12 14
```



예제 출력1

```
4
```

프로그래머스 : 체육복 – Lv.1

문제

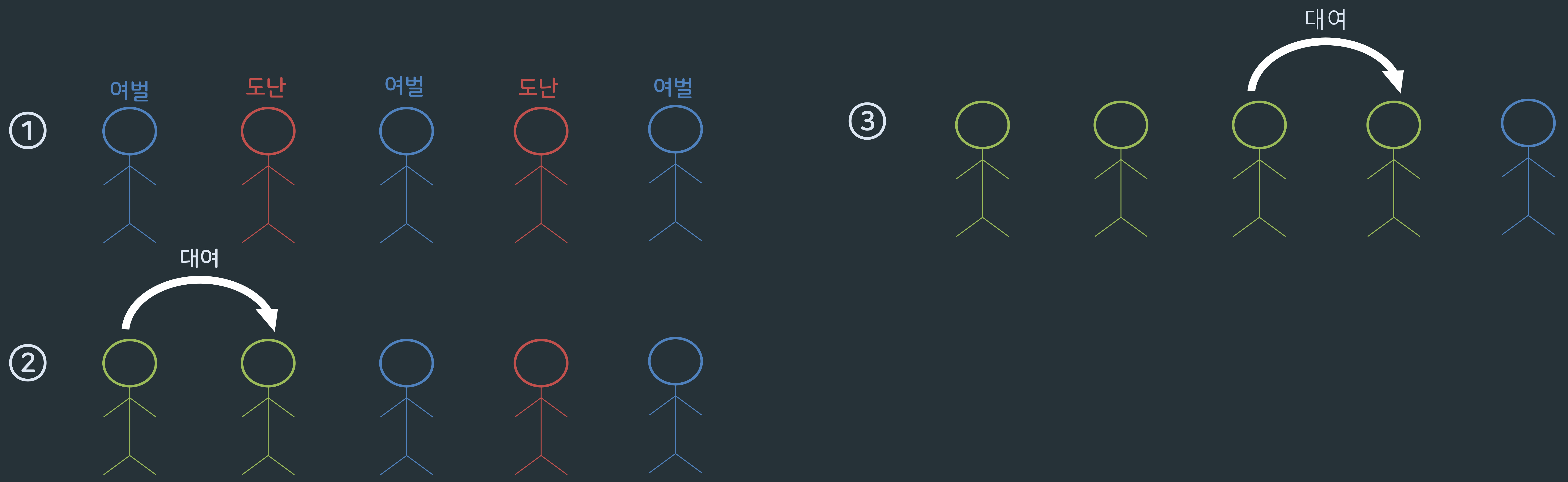
- n명의 학생들 중 일부가 체육복을 도난당했다.
- 여벌 옷이 있는 학생은 바로 앞번호 또는 뒷번호 학생에게 체육복을 빌려줄 수 있다.
- 전체 학생의 수 n, 체육복을 도난당한 학생들의 번호가 담긴 배열 lost, 여벌의 체육복을 가져온 학생들의 번호가 담긴 배열 reserve가 매개변수로 주어진다.
- 체육 수업을 들을 수 있는 학생의 최댓값 구하는 문제

제한 사항

- 전체 학생의 수 n의 범위는 2 <= n <= 30
- 체육복을 도난당한 학생의 수는 1명 이상, n명 이하 (중복X)
- 여벌 체육복이 있는 학생의 수는 1명 이상, n명 이하 (중복X)
- 여벌 체육복을 가져온 학생이 체육복을 도난당했다면 <u>체육복을 빌려줄 수 없다.</u>

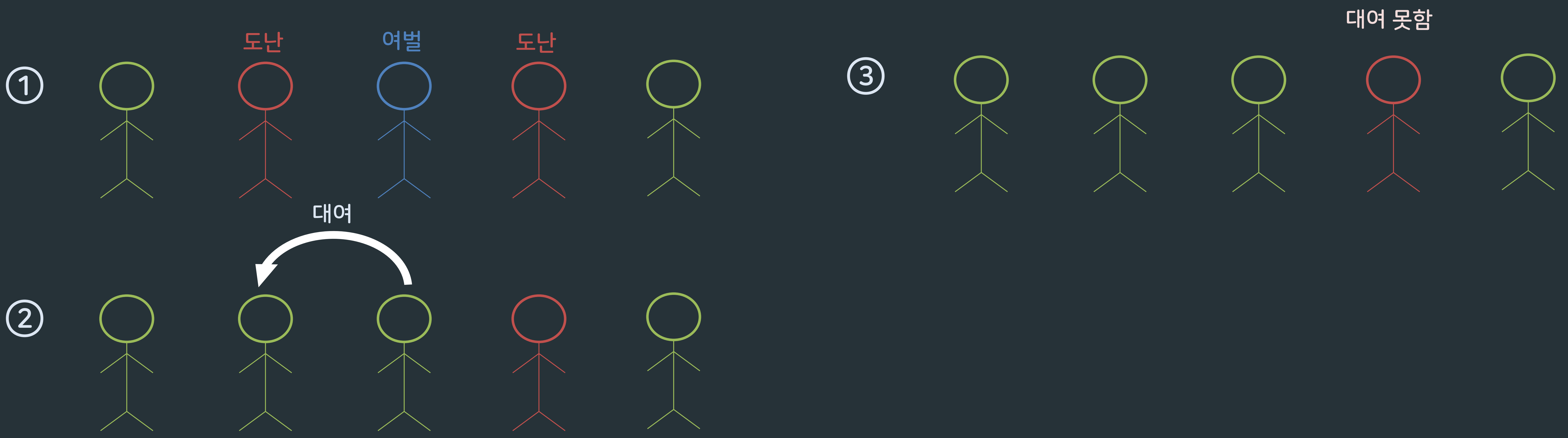
예제

n	lost	reserve	return
5	[2, 4]	[1, 3, 5]	5
5	[2, 4]	[3]	4
3	[3]	[1]	2



예제

n	lost	reserve	return
5	[2, 4]	[1, 3, 5]	5
5	[2, 4]	[3]	4
3	[3]	[1]	2



#알튜비튜

접근

앞 번호부터 체육복을 대여할 수 있는지 여부를 차례대로 확인한다.

- 탐욕 선택 속성: 탐욕적으로 선택하더라도 문제의 최적해가 보장되어야 한다
  - k번째 사람이 빌릴 수 있는 상황에서 양보한다면? (-1)
    - k+1번째 사람에게 체육복이 있는 경우, k+2번째 사람이 빌릴 수 있다 (-1 +1)
    - k-1번째 사람에게 체육복이 있는 경우, k+2번째 사람은 빌릴 수 없게 된다.(-1)
    - 따라서, 빌릴 수 있는 상황에서는 무조건 빌려야 손해가 발생하지 않는다.

- 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있을 때
  - 앞에서부터 체크하는 경우, 중간에 누락되지 않으므로 부분 문제가 성립한다.

마무리

정리

- 우리가 원하는 답을 여러 개의 조각으로 쪼개고, 각 단계마다 최적해를 구한다.
- 그리디 알고리즘의 사용 조건
  - 탐욕 선택 속성: 탐욕적으로 선택하더라도 문제의 최적해가 보장될 때 (손해X)
  - 최적 부분 구조: 부분 문제의 최적해가 전체 문제의 최적해로 확장될 수 있을 때

이것도 알아보세요!

- 다른 알고리즘과의 차이점(동적계획법 – 다음 주차 강의를 듣고 비교해보아요~)
- 알고리즘의 정당성을 증명하는 과정

과제

필수

- 15662번 : 톱니바퀴 (2) – Gold 5
- 19941번 : 햄버거 분배 – Silver 3
- 17451번 : 평행 우주 – Silver 3

도전

- 큰 수 만들기 - Lv.2
- 19539번 : 사과나무 – Gold 5