Algorithm Week 9

17기 정규세션

TOBIG’S 16기 전민진

Contents

17기 정규세션
TOBIG'S 16기 전민진

Unit 01 | 7주차 문제 리뷰

Unit 02 | greedy algorithm

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

Unit 01 | 7주차 문제 리뷰

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 1. 병합 정렬

다음 주어진 코드의 빈칸을 채워 입력으로 주어진 여러 수의 나열 중 작은 수부터 큰 수 순서대로 배열하는 병합 정렬 알고리즘을 만들어 보자.

첫째 줄에 N(1 ≤ N ≤ 1,000,000)개의 수의 나열이 주어진다. 이 수는 절댓값이 1,000,000보다 작거나 같은 정수이다. 수는 중복되지 않는다.

첫째 줄부터 N개의 줄에 오름차순으로 정렬한 결과를 한 줄에 하나씩 출력한다.

```
###input 구현###
lst = list(map(int, input().split()))


### mergeSort 함수의 빈칸 ___를 채워 완성해주세요 ###

def mergeSort(A):
  ### Base case ###
    if len(A) <= 1:
        return ___

    ### A를 반으로 쪼개서 recursive call을 해 주고 정렬된 반쪽짜리들을 받아옵니다.
    left = ___
    right = ___

    ### 여기서부터 두 개의 쪼개졌던 list를 합치는 Merge 부분입니다.
    ### 여기서 포인트는 정렬된 상태로 돌아왔기 때문에 앞에서부터 순차적으로 한번만 돌면 정렬
    i, j, k = 0, 0, 0
    while i<len(left) and j<len(right):
        if ___:
            A[k] = left[i]
            i += 1
        else:
            A[k] = right[j]
            j += 1
        k += 1
    if ___: #만약 left의 원소를 모두 채웠고, right가 남아있을 때.
        while j<len(right):
            A[k] = right[j]
            j += 1
            k += 1
    elif ___: #만약 right의 원소를 모두 채웠고, left가 남아있을 때.
        while i<len(left):
            A[k] = left[i]
            i += 1
            k += 1
    return A #마지막으로 정렬된 list를 리턴합니다.


###output 구현###
ans = mergeSort(lst)
for i in range(len(ans)):
  print(ans[i])
```

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 1. 병합 정렬

```python
lst = list(map(int, input().split()))
def mergeSort(A):
  ### Base case ###
    if len(A) <= 1:
        return A

    ### A를 반으로 쪼개서 recursive call을 해 주고 정렬된 반쪽짜리들을 받아옵니다.
    left = mergeSort(A[:len(A)//2])
    right = mergeSort(A[len(A)//2:])

    ### 여기서부터 두 개의 쪼개졌던 list를 합치는 Merge 부분입니다.
    ### 여기서 포인트는 정렬된 상태로 돌아왔기 때문에 앞에서부터 순차적으로 한번만 돌면 정렬
    i, j, k = 0, 0, 0
    while i<len(left) and j<len(right):
        if left[i] < right[j]:
            A[k] = left[i]
            i += 1
        else:
            A[k] = right[j]
            j += 1
        k += 1
    if i == len(left): #만약 left의 원소를 모두 채웠고, right가 남아있을 때.
        while j<len(right):
            A[k] = right[j]
            j += 1
            k += 1
    elif j == len(right): #만약 right의 원소를 모두 채웠고, left가 남아있을 때.
        while i<len(left):
            A[k] = left[i]
            i += 1
            k += 1
    return A #마지막으로 정렬된 list를 리턴합니다.

ans = mergeSort(lst)
for i in range(len(ans)):
  print(ans[i])
```

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 2. 이상한 나라의 가위바위보

NXN명의 사람이 가위바위보를 한다. 이 때, 다음과 같은 수칙에 따라 가위바위보를 하려고 한다.
(가위, 바위, 보는 시작 시 낸 것이 기준이다.)

1. 만약 모든 사람이 같은 것을 냈다면 하나의 조로 확정된다.
2. 1번이 아닌 경우에 전체 인원을 같은 크기의 9개의 조로 나누고, 각각의 나뉜 조에 대해 1번의 과정을 반복한다.

이와 같이 가위바위보를 할 때, 전체가 가위를 낸 조의 수, 전체가 바위를 낸 조의 수, 전체가 보를 낸 조의 수를 구하는 프로그램을 작성하시오.

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 2. 이상한 나라의 가위바위보

[입력 조건]

- 첫째 줄에 N($1 \le N \le 3^7$, N은 3의 거듭제곱 꼴)이 주어진다. 다음 N개의 줄에는 N개의 정수로 행렬이 이루어진다.(이 때, -1은 가위, 0은 바위, 1은 보를 냈음을 의미한다.)

[출력 조건]

- 첫째 줄에 -1(가위)만 낸 조의 수를, 둘째 줄에 0(바위)만 낸 조의 수를, 셋째 줄에 1(보)만 낸 조의 수를 출력한다.

[입력 예시]

```
9
0 0 0 1 1 1 -1 -1 -1
0 0 0 1 1 1 -1 -1 -1
0 0 0 1 1 1 -1 -1 -1
1 1 1 0 0 0 0 0 0
1 1 1 0 0 0 0 0 0
1 1 1 0 0 0 0 0 0
0 1 -1 0 1 -1 0 1 -1
0 1 -1 0 1 -1 0 1 -1
0 1 -1 0 1 -1 0 1 -1
```

[출력 예시]

```
10
12
11
```

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 2. 이상한 나라의 가위바위보

```python
n = int(input())
graph = [list(map(int, input().split())) for _ in range(n)]

one_count = 0
zero_count = 0
m_one_count = 0

def dnc(x, y, n):
    global one_count, zero_count, m_one_count

    check = graph[x][y]
    for i in range(x, x + n):
        for j in range(y, y + n):
            if check != graph[i][j]:
                check = -2
                break

    if check == -2:
        n = n // 3
        dnc(x, y, n)
        dnc(x, y + n, n)
        dnc(x, y + 2 * n, n)
        dnc(x + n, y, n)
        dnc(x + n, y + n, n)
        dnc(x + n, y + 2 * n, n)
        dnc(x + 2 * n, y, n)
        dnc(x + 2 * n, y + n, n)
        dnc(x + 2 * n, y + 2 * n, n)
    elif check == 1:
        one_count += 1
    elif check == 0:
        zero_count += 1
    else:
        m_one_count += 1

dnc(0, 0, n)
print(m_one_count)
print(zero_count)
print(one_count)
```

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 3. 최후의 1인

22학번인 투빅이는 새터에 가게 되었다. 새터에서 게임을 해, 최종 1인에게 선물을 주려고 한다. 게임의 방식은 다음과 같다.

1. NxN명이 강당에 앉아있고, 4명씩(2x2)로 조를 짜 게임을 한다.
2. 각 조원들은 -10000이상 10000이하의 숫자를 낼 수 있다. 이 때, 각 조에서 숫자가 2번째로 큰 사람만이 남아 게임을 계속한다고 한다.
3. 이를 최후의 1인이 남을 때까지 반복한다.

게임에 NxN명의 학생이 참여했을 때, 최종 1인이 낸 숫자를 산출하시오. (출력의 각 행에 주어진 원소 N개는 그 행의 N명의 학생들이 각각 낸 숫자에 해당한다.)

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 3. 최후의 1인

[입력 조건]

첫째 줄에 N(2<= N <= 1024)이 주어진다. N은 항상 2의 거듭제곱 꼴이다.(N=2^k, 1<=k<=10) 다음 N개의 줄 마다 각 행의 원소 N개가 차례대로 주어진다. 행렬의 모든 성분은 -10000 이상 10000이하의 정수이다.

[출력 조건]

마지막에 남은 수를 출력한다.

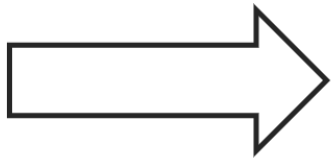
-1	2	14	7	4	-5	8	9
10	6	23	2	-1	-1	7	11
9	3	5	-2	4	4	6	6
7	15	0	8	21	20	6	6
19	8	12	-8	4	5	2	9
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 3. 최후의 1인

-1	2	14	7	4	-5	8	9
10	6	23	2	-1	-1	7	11
9	3	5	-2	4	4	6	6
7	15	0	8	21	20	6	6
19	8	12	-8	4	5	2	9
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24



6	14	-1	9
9	5	20	6
8	4	5	8
17	19	21	23

Unit 01 | 7주차 문제 리뷰

17기 정규세션
TOBIG'S 16기 전민진

문제 3. 최후의 1인

```python
def polling(n,start_x,start_y):
    half = n//2
    if n==2:
        arr = [matrix[start_x][start_y], matrix[start_x+1][start_y], matrix[start_x][start_y+1], matrix[start_x+1][start_y+1]]
        arr.sort()
        return arr[-2]
    left_top = polling(half,start_x,start_y)
    right_top = polling(half,start_x+half,start_y)
    left_bottom = polling(half,start_x,start_y+half)
    right_bottom = polling(half,start_x+half,start_y+half)
    arr = [left_top, right_top, left_bottom, right_bottom]
    arr.sort()
    return arr[-2]

N = int(input())
matrix = [list(map(int, input().split())) for _ in range(N)]
print(polling(N,0,0))
```

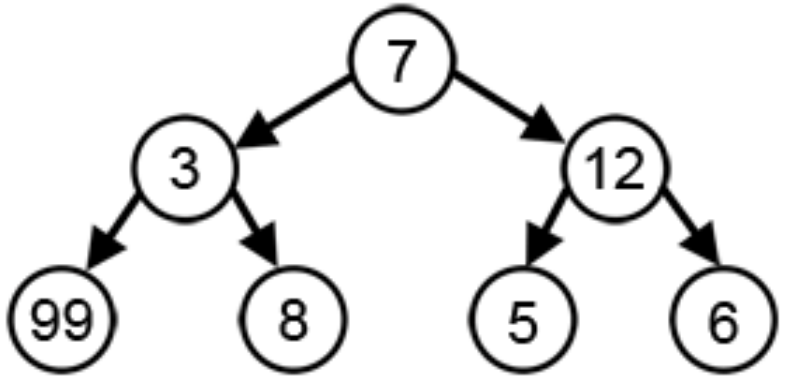
17기 정규세션
TOBIG'S 16기 전민진

Unit 02 | greedy algorithm

Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

- A *greedy algorithm* always makes the choice that looks best at the moment
- It makes a locally optimal choice in the hope that this choice will lead to a globally optimal solution
- So, in many problems, a greedy strategy does not produce an optimal solution.

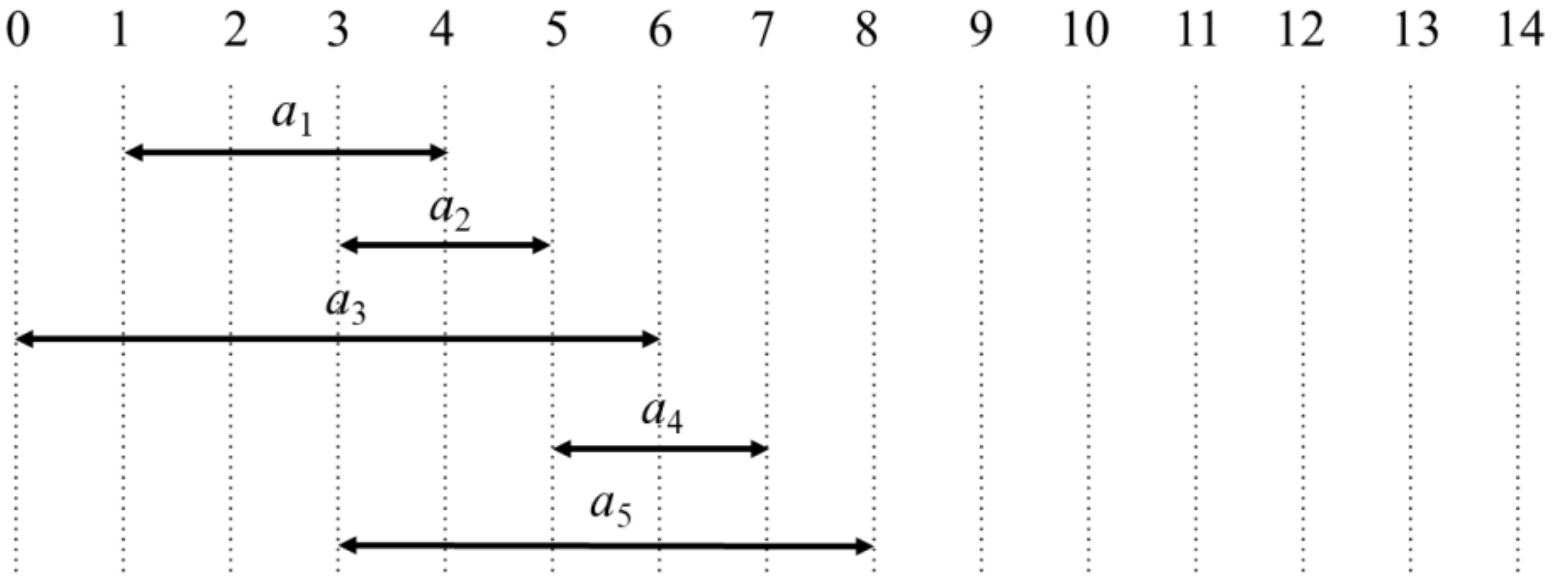


Find the path with the largest sum

Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

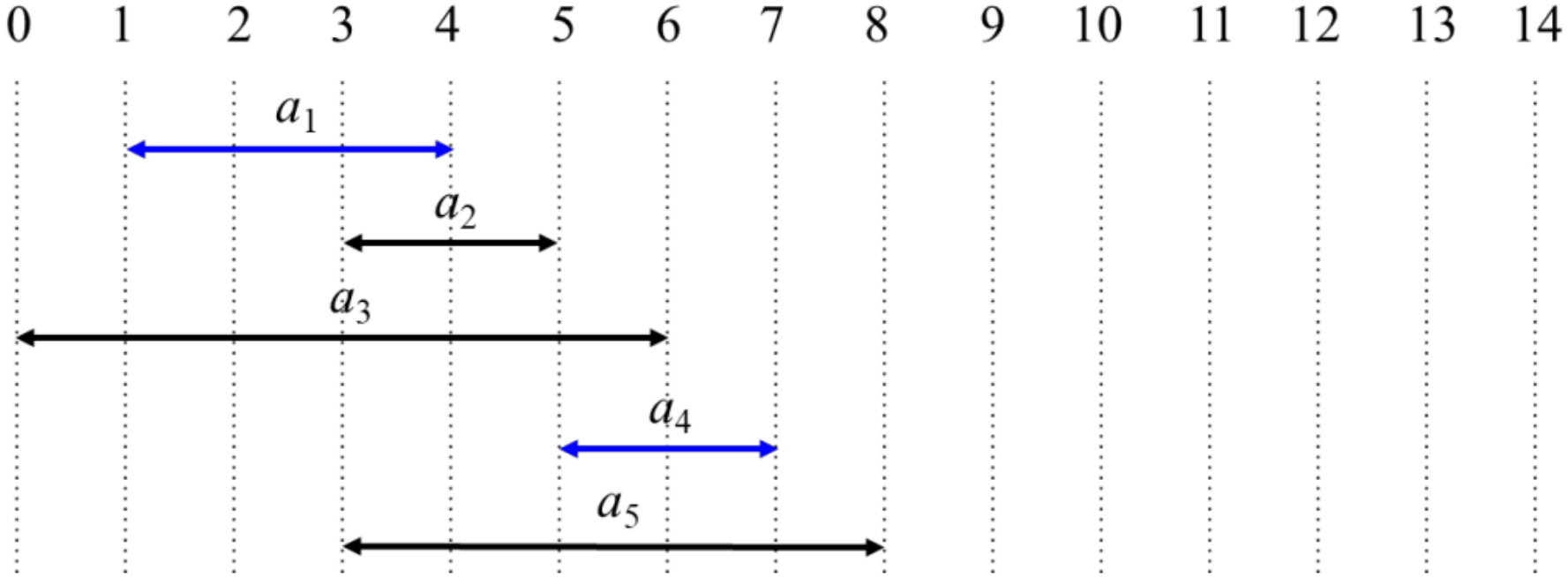
- An activity selection problem
  - To select a maximum-size subset of mutually compatible activities.
  - For example,
    - Given : n activities and only 1 lecture room,
    - Goal : to select the maximum number of activities to be scheduled in 1 lecture room



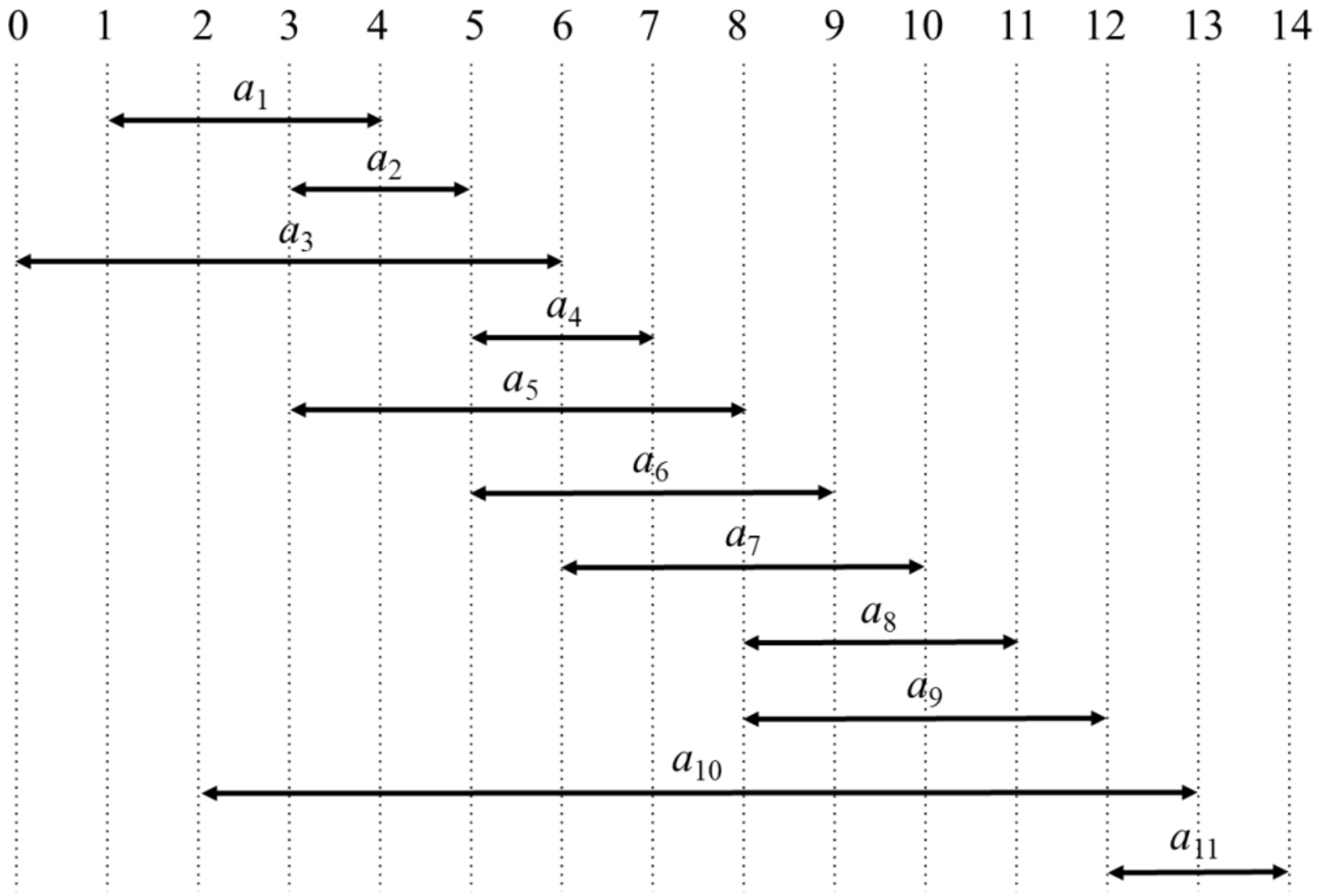
Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

- A set of activities :
- Each activitiy a_i has its start times s_i and finish time f_i
  - $0 \leq s_i < f_i < \infty$
- Activity a_i takes place during $[s_i, f_i)$
- Activities a_i and a_j are compatible if the intervals $[s_i, f_i)$ and $[s_i, f_i)$ do not overlap



Unit 02 | greedy algorithm

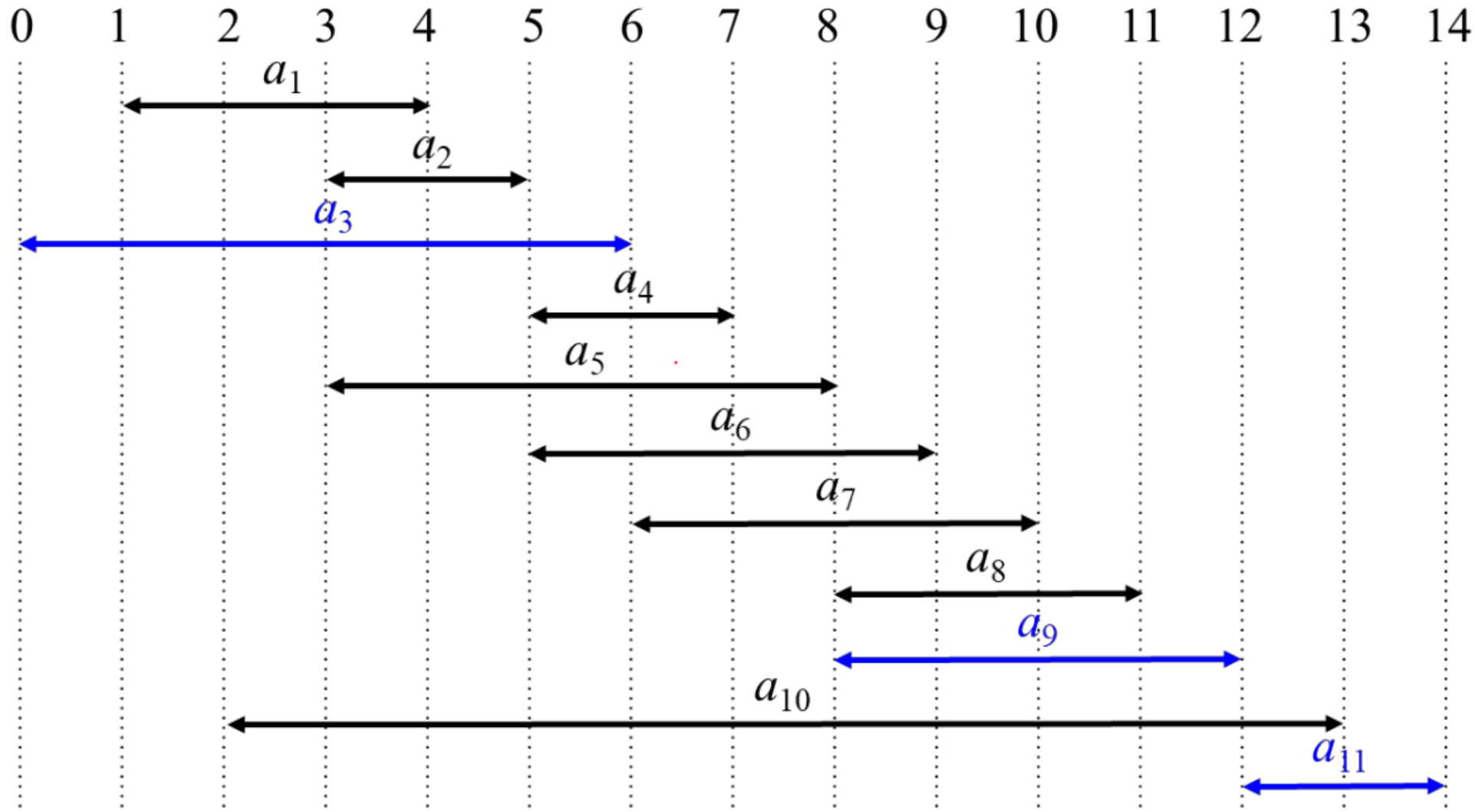


17기 정규세션
TOBIG'S 16기 전민진

Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

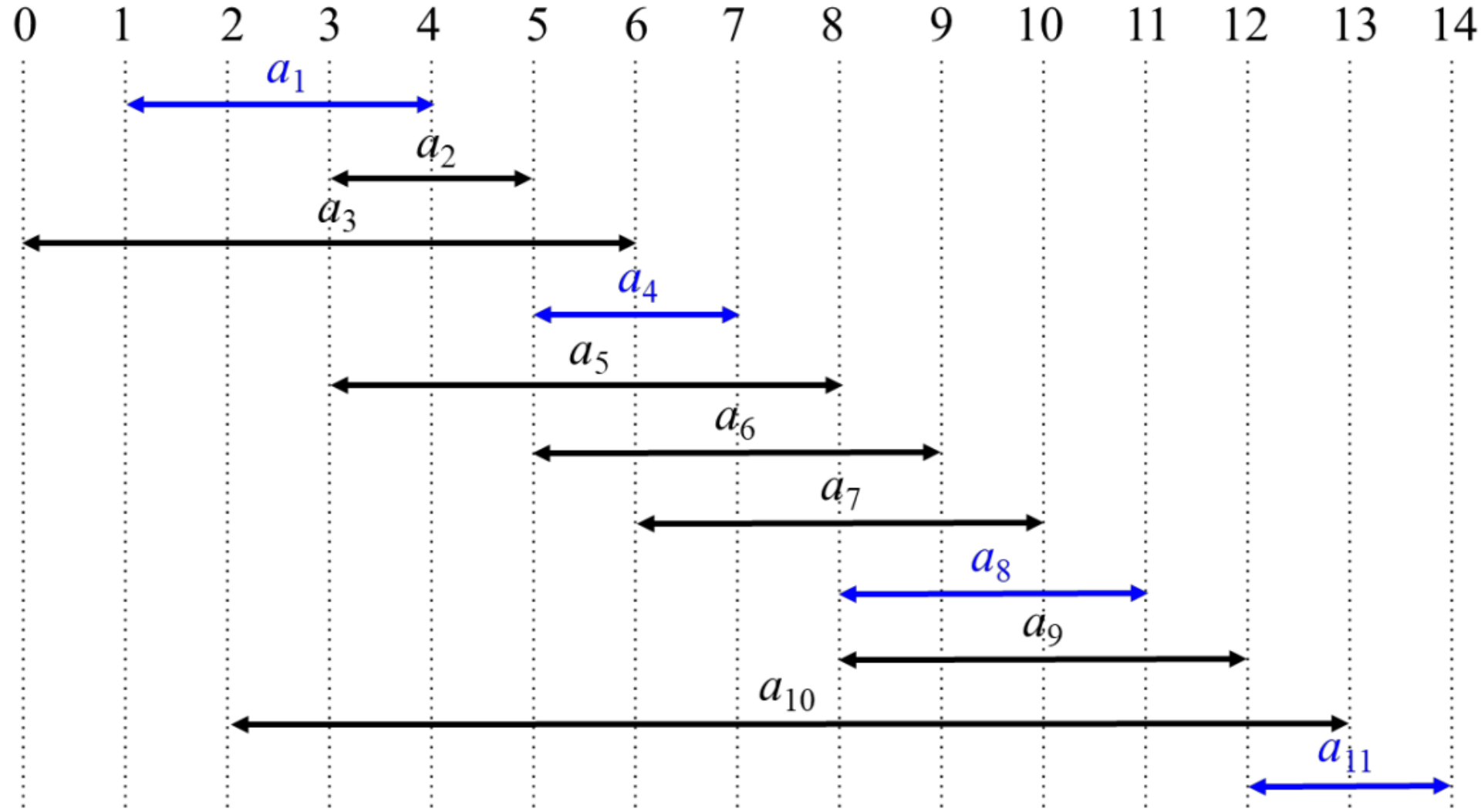
- $\{a_3, a_9, a_{11}\}$: mutually compatible activities, not a largest set



Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

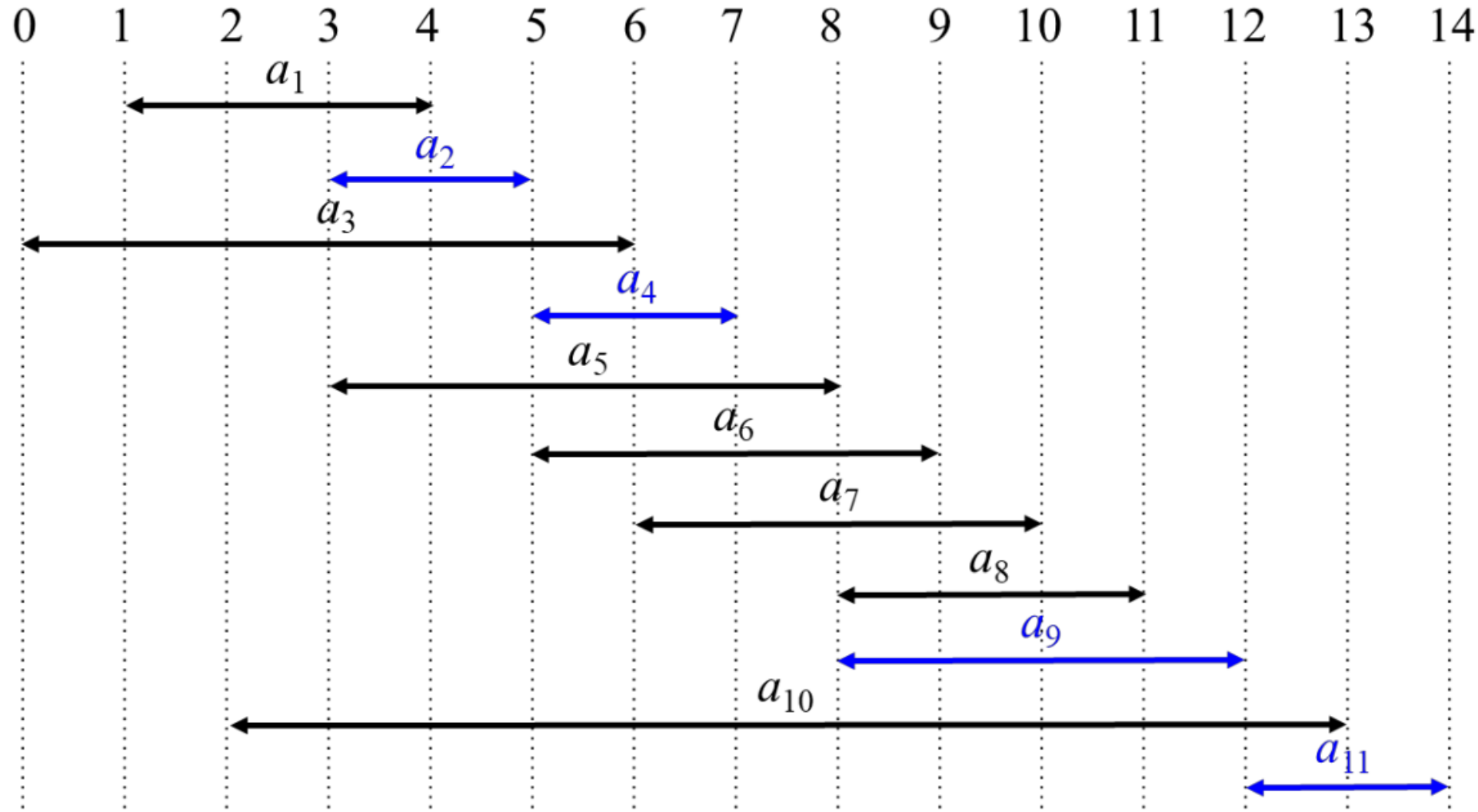
- $\{a_1, a_4, a_8, a_{11}\}$: A largest set of mutually compatible activities



Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

- $\{a_2, a_4, a_9, a_{11}\}$: Another largest subset



Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

- Optimal substructure
  - Assume that activities are sorted in increasing order of finish time.

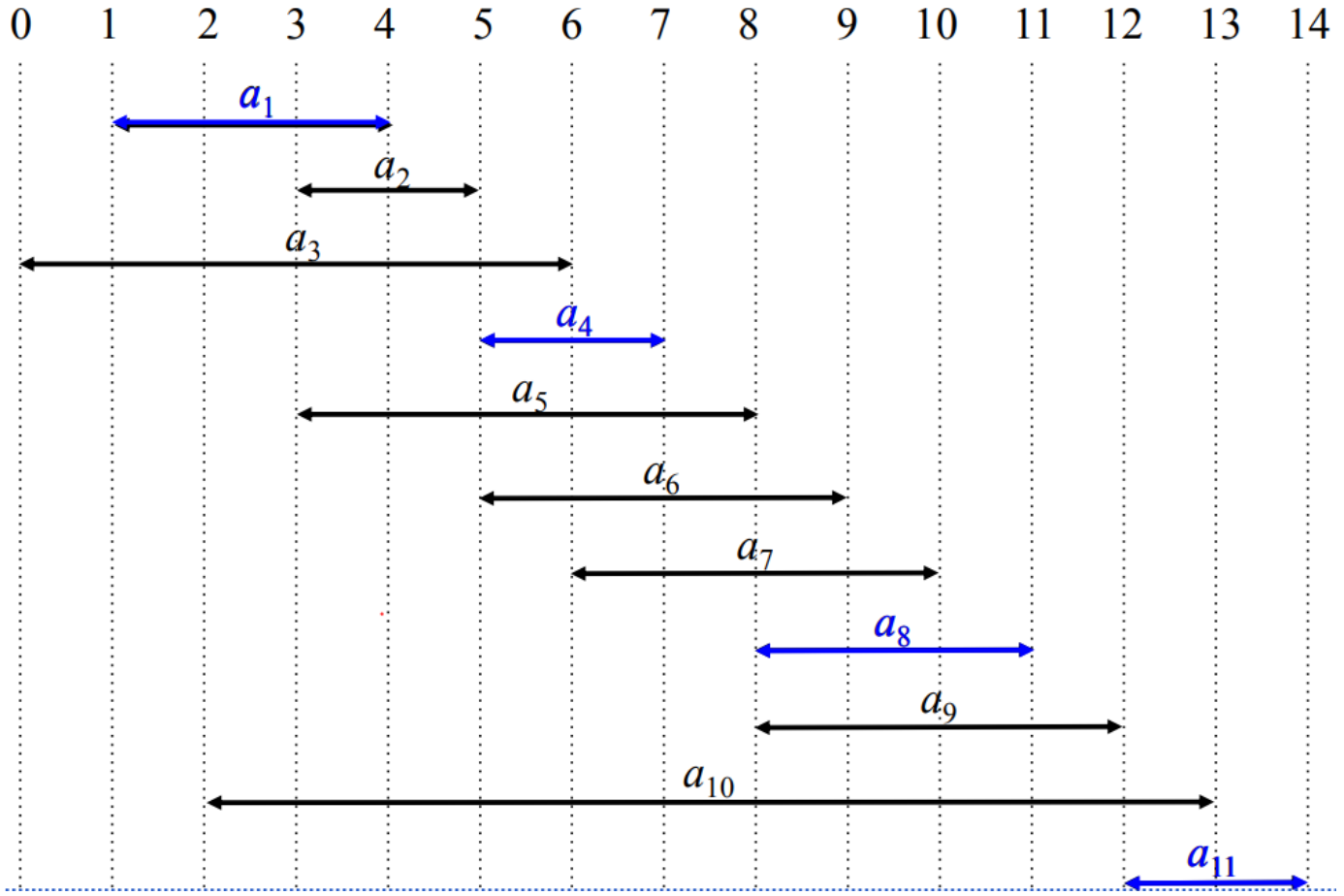
$$f_0 \le f_1 \le f_2 \le \cdots \le f_n < f_{n+1}$$

i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

Unit 02 | greedy algorithm

17기 정규세션
TOBIG'S 16기 전민진

- Greedy algorithm
  - Select the earliest finishing activity one by one



Unit 03 | 9주차 문제 소개

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

문제 1. 저금통 털기

승주는 1년 동안 저금통에 동전을 가득 채웠다. 승주의 저금통에 있는 동전의 종류는 총 N종류이며 각각의 동전의 개수는 충분히 많다. 승주는 저금통에 있는 동전을 적절히 사용해서 K원의 물건을 구매하고자 한다. 단, 최소한의 동전 개수로 계산을 하고 싶다. 승주를 도와 K원을 만드는데 필요한 동전 개수의 최솟값을 찾아보자.

[입력 조건]

첫째 줄에 N과 K가 주어진다. (단, 1<=N<=10, 1<=K<=100,000,000)

둘째 줄부터 N개의 줄에 동전의 가치 A(i)가 오름차순으로 주어진다.

(1 ≤ A(i) ≤ 1,000,000, A(1) = 1, i ≥ 2인 경우에 A(i)는 A(i-1)의 배수)

[출력 조건]

K원을 만드는데 필요한 동전의 최소 개수를 출력한다.

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

문제 1. 저금통 털기

[입력 예시]

10 4790
1
5
10
50
100
500
1000
5000
10000
50000

[출력 예시]

12

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

문제 2. 연수원의 공중 전화

휴대 전화를 사용할 수 없는 투빅스 연수원에는 단 하나의 공중 전화가 설치되어 있다. 연수생 N명은 공중 전화를 사용하여 전화를 하고 싶어 한다. 투빅스 연수원의 담당자인 권호는 연수생 N명에게 전화 시작 시간과 끝나는 시간이 담긴 전화 사용 계획표를 받았다. 권호를 도와 최대한 많은 연수생들이 공중 전화를 이용할 수 있도록 각 전화 시간이 겹치지 않으면서 공중 전화를 사용할 수 있는 최대 인원의 수를 구해보자.

단, 통화는 한번 걸면 중간에 끊을 수 없으며 앞 사람의 통화가 끝나는 것과 동시에 다음 사람의 통화가 시작될 수 있다. 또한, 통화 시작 시간과 끝나는 시간이 같을 수도 있다. 이 경우에는 시작하자마자 끝나는 것으로 생각하면 된다.

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

문제 2. 연수원의 공중 전화

[입력 조건]

첫째 줄에 전화를 걸고 싶어하는 연수생의 수 N이 주어진다. 둘째 줄부터 N+1 줄까지 각 연수생의 전화 사용 계획표가 주어지는데 이것은 공백을 사이에 두고 전화 시작 시간과 끝나는 시간이 주어진다.

[출력 조건]

공중 전화를 사용할 수 있는 최대 인원의 수를 출력한다.

[입력 예시]

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

[출력 예시]

```
4
```

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

문제 3. 신기한 렌트카

투빅이들이 컨퍼런스에 참석하기 위해 렌트카를 빌려 장소로 이동하려고 한다.
그런데 렌트카는 작아서 한 번에 최대 2명씩밖에 탈 수 없고, 무게 제한도 있다.

예를 들어, 투빅이 4명의 몸무게가 [70kg, 50kg, 80kg, 50kg]이고 렌트카의 무게 제한이 100kg이라면 2번째 투빅이와 4번째 투빅이는 같이 탈 수 있지만 1번째 투빅이와 3번째 투빅이의 무게의 합은 150kg이므로 무게 제한을 초과하여 같이 탈 수 없다.

렌트카를 최대한 적게 빌려 모든 투빅이들을 컨퍼런스 장소로 이동시키려고 할 때, 필요한 렌트카 수의 최솟값을 출력하시오.

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

문제 3. 신기한 렌트카

[입력 조건]

- 첫째 줄에는 공백을 기준으로 투빅이들의 몸무게가 한 줄로 입력된다.
- 둘째 줄에는 렌트카의 무게 제한이 주어진다.
- 이때 각 투빅이의 몸무게는 40kg 이상 240kg 이하이며, 렌트카의 무게 제한 역시 동일하다.
- 렌트카의 무게 제한은 항상 투빅이들의 몸무게의 최댓값보다 크게 주어진다.

[출력 조건]

필요한 렌터카 수의 최솟값을 출력한다.

[입력 예시1]

70 50 80 50
100

[출력 예시1]

3

[입력 예시2]

70 80 50
100

[출력 예시2]

3

Unit 03 | 9주차 문제 소개

17기 정규세션
TOBIG'S 16기 전민진

Course Code

87b7b01aa2

Course Link

https://class.mimir.io/courses/87b7b01aa2/registrations/new

질문 / 힌트 / 오류 등 모든 문의는

누구에게? 김권호 / 박한나 / 이승주 / 이예림 / 전민진 혹은 멘토에게!

언제? 24시간 OK

알고리즘 파이팅!