# Nhóm 11

# Mô tả thuật toán:

Điều kiện đề bài có phần đặc biệt:

$$\text{age}[i] \leq \text{age}[j],\ \text{intelligence}[i] \leq \text{intelligence}[\text{j}]$$

Quan sát điều kiện trên, ta có thể sort lại theo thứ tự tăng dần theo age, nếu age bằng nhau thì tăng dần theo intelligence để đơn giản hóa bài toán và đưa về dạng tìm dãy con không giảm có trọng số lớn nhất trên intelligence.

Từ đó, ta có thể đưa ra thuật toán của 2 subtask.

## Subtask 1:

---

Với subtask 1, giới hạn $n$ là rất nhỏ ($n \leq 20$) nên tả có thể sử dụng brute-force để có thể duyệt qua mọi tập hợp có thể và kiểm tra liệu chúng có phải 1 dãy con không giảm hay không và lưu trọng số vào biến kết quả.

## Subtask 2:

---

Ta rõ ràng có thể thấy được rằng nếu sử dụng lại thuật toán của Subtask 1 thì độ phức tạp sẽ rất lớn, lên tới $O(2^N * N)$. Vì vậy, ta cần một thuật toán tốt hơn.

Từ nhận xét ban đầu, ta thấy, bài toán hiện tại là tìm dãy con tăng có trọng số lớn nhất. Ta gọi $\text{Product}(n)$ là tích lớn nhất có thể đạt được xét tới chiến binh thứ $n$, những chiến binh được chọn tạo thành một dãy thỏa mãn điều kiện đề bài và chiến binh thứ $n$ là người đứng cuối hàng.

Từ đây, ta có thể thấy yêu cầu bài toán là tìm $\text{Product}(n)$ với hàm mục tiêu ta vừa định nghĩa.

Để tính $\text{Product}(n)$ ta cần tìm những chiến binh có age và intelligence nhỏ hơn chiến binh thứ $n$, gọi tập đó là tập chỉ số các chiến binh tìm được $S$. Dựa vào đó, ta có công thức truy hồi như sau:

$$\text{Product}(n) = \underset{j \in S}{Max}(\text{Product}(j) \cdot \text{Power}_i)$$

Để làm cho công thức trên dễ hiểu hơn, hãy tưởng tượng $\text{Product}(j)$ là một dãy chiến binh kết thúc có người cuối cùng là $j$. Vì chiến binh thứ $n$ có thể vào hàng vì thỏa mãn điều kiện rằng age và intelligence đều lớn hơn hoặc bằng mọi người trong hàng nên ta sẽ thêm $n$ vào hàng, tức hàng đó sẽ kết thúc tại chiến binh thứ $n$ và có tích bằng $\text{Product}(n) = \text{Product}(j) \cdot \text{Power}_i$

Tuy nhiên tới đây, ta nhận thấy bài toán có điểm bất thường là tích có thể âm. Điều này dẫn tới việc chỉ lưu Max của tích thôi là chưa đủ vì số âm nhân số âm vẫn thành số dương được.

Từ đó, ta sẽ thay đổi định nghĩa như sau

- $\text{Product}(n, 0)$ là tích lớn nhất có thể đạt được xét tới chiến binh thứ $n$, những chiến binh được chọn tạo thành một dãy thỏa mãn điều kiện đề bài và chiến binh thứ $n$ là người đứng cuối hàng.
- $\text{Product}(n, 1)$ là tích nhỏ nhất có thể đạt được xét tới chiến binh thứ $n$, những chiến binh được chọn tạo thành một dãy thỏa mãn điều kiện đề bài và chiến binh thứ $n$ là người đứng cuối hàng.

Với việc định nghĩa thay đổi, công thức truy hồi của chúng ta cũng thay đổi những ý tưởng vẫn sẽ được giữ nguyên:

- $\text{Product}(n, 0) = \underset{j \in S}{Max}(\text{Product}(j, 0) \cdot \text{Power}_i)$
- $\text{Product}(n, 0) = \underset{j \in S}{Max}(\text{Product}(j, 1) \cdot \text{Power}_i)$
- $\text{Product}(n, 1) = \underset{j \in S}{Min}(\text{Product}(j, 0) \cdot \text{Power}_i)$
- $\text{Product}(n, 1) = \underset{j \in S}{Min}(\text{Product}(j, 1) \cdot \text{Power}_i)$

## Cài đặt thuật toán

PYTHON

```python
import sys
input = sys.stdin.readline

INF = int(1e9)

n = int(input())
soldier = []
Product = [[0 for _ in range(2)] for _ in range(n)]

for i in range(n):
    a , b , c = map(int , input().split())
    soldier.append([a , b , c])

soldier.sort()

res = -INF
```

```
for i in range(n):

    Product[i][0] = soldier[i][2]
    Product[i][1] = soldier[i][2]

    for j in range(i):
        if soldier[j][0] > soldier[i][0] or soldier[j][1] > soldier[i][1]:
            continue

        Product[i][0] = max(Product[i][0] , Product[j][1] * soldier[i][2])
        Product[i][0] = max(Product[i][0] , Product[j][0] * soldier[i][2])

        Product[i][1] = min(Product[i][1] , Product[j][1] * soldier[i][2])
        Product[i][1] = min(Product[i][1] , Product[j][0] * soldier[i][2])

    res = max(res , Product[i][0])
    res = max(res , Product[i][1])

print(res)
```

## Phân tích độ phức tạp:

- Subtask 1:
  - Độ phức tạp thời gian: $O(2^N * N)$
  - Độ phức tạp không gian: $O(N)$
- Subtask 2:

- Độ phức tạp thời gian: $O(N^2)$
- Độ phức tạp không gian: $O(N)$