Thuật toán ứng dụng buổi số 4

Bài 1. **Bài toán tìm cách bán hàng tối ưu**

Trên một trục đường thẳng có n cửa hàng tạp hóa liên tiếp nhau. Mỗi cửa hàng đều có nhu cầu làm đại lý cho sản phẩm mới ra mắt của công ty X. Nhu cầu của cửa hàng thứ i sẽ này được biểu diễn bởi 1 giá trị nguyên $\mathbf{a}_i$.

Với mục đích phân phối sản phẩm rộng nhất, nên công ty đưa ra chính sách là hai đại lý bán sản phẩm này phải cách nhau ít nhất khoảng cách **d**, trong đó **L1<=d<=L2**.

Hãy đưa ra tổng số lượng nhu cầu sản phẩm là lớn nhất với cách chọn đại lý sao KHÔNG vi phạm chính sách đại lý của công ty.

Ví dụ. File đầu vào và đầu ra dạng

Input	**Output**
6 2 3 3 5 9 6 7 4	19

Dòng đầu tiên của file input gồm

- N, L1, L2 lần lượt là số lượng cửa hàng liên tiếp nhau, L1 và L2 là 2 ngưỡng giới hạn trên và dưới của hai đại lý gần nhau
- Dòng tiếp theo lần lượt là nhu cầu của các cửa hàng $\mathbf{a}_i$ về sản phẩm mới

Trong ví dụ trên thì 2 đại lý gần nhau nhất phải cách nhau 2-3 vị trí, vậy phương án tốt nhất sẽ là chọn cửa hàng 1, 3 và 5 với tổng nhu cầu sẽ là 3+9+7 = 19

Gợi ý

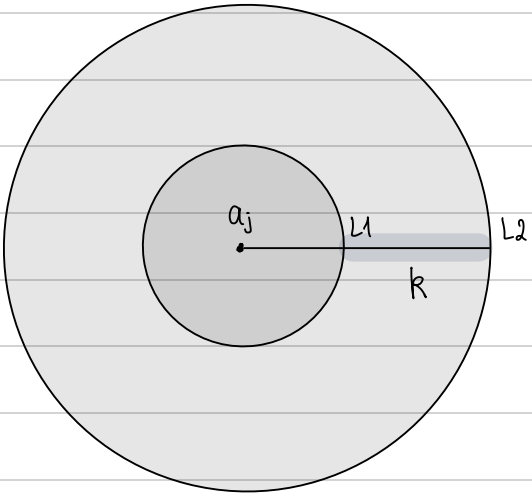
- Thuật toán vét cạn sẽ không khả thi vì số trường hợp rất nhiều
- Xây dựng thuật toán theo hướng quy hoạch động

Thuật toán quy hoạch động

- Cửa hàng được chọn/không chọn sẽ được biểu diễn dưới dạng chuỗi nhị phân độ dài n
- Của hàng nào được chọn sẽ là bit 1 và không chọn sẽ là bit 0
- Duyệt lần lượt từ trái qua phải, tìm cách chọn nào sẽ cho tổng nhu cầu lớn nhất tới vị trí hiện tại
- Vì nhu cầu là số >=0 nên nếu ở vị trí cửa hàng thứ i ta có thể có các phương án sau
    - Nếu i<L1 thì ta chỉ có thể chọn giá trị là MAX nhu cầu các cửa hàng với thứ tự từ 1 -> i
    - Nếu L1<=i<=L2 thì ta có thể chọn
        - MAX giá trị nhu cầu từ của hàng thứ tự i-L1 tới i
        - hoặc, nhu cầu cửa hàng thứ i + MAX nhu cầu trong khoảng (i-L2) tới (i-L1)

Ví dụ với đầu vào ở trên, 2<=d<=3

a_1 a_2 a_3 ... $\boxed{a_j}$ ... a_n

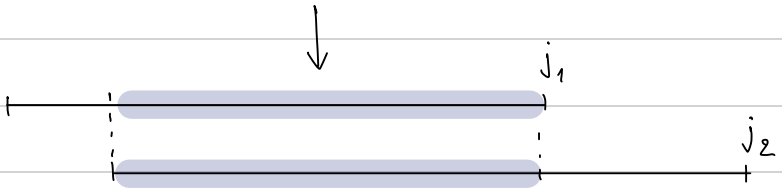


S_j : dãy con, lấy a_j là ptử cuối cùng

$$S_j = a_j + \max S_{j-k} \qquad k = L_1 \to L_2$$

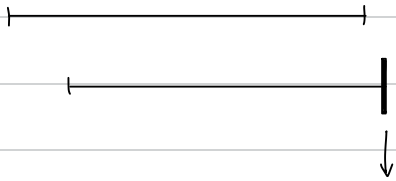
$O(n)$ $O(n)$ $\to O(n^2)$

Vấn đề: Lặp tính toán



↳ Giải quyết: Monotonic Stack!

Gsử xét 1 khoảng

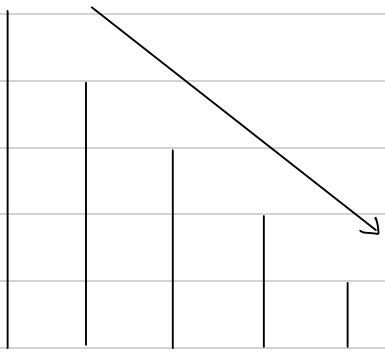


Có lấy ptử này?

Xét ⊙ khoảng trc ptử đó:

1) Dịch sang phải 1
2) Nếu bên trái ptử ms thêm có ptử nào < ptử mới đó → Loại bỏ

Kết quả



Tại ô đầu tiên thì cách chọn lớn nhất sẽ là chọn cửa hàng 1 với tổng nhu cầu là 3

3					

Tại cửa hàng thứ 2 tiếp theo, cách chọn tới vị trí này sẽ là :

- hoặc chọn cửa hàng 1,
- hoặc chọn cửa hàng 2,

Không thể chọn cả 2 vì khoảng cách d mới bằng 2-1=1

3 (chọn 1)	5(chọn 2)				

Tại cửa hàng thứ 3 tiếp theo, với 3 > giá trị tối thiểu của L1 nên cách chọn sẽ là

- Chọn nhu cầu tại cửa hàng 3 + MAX(ô trong khoảng 0-1)
- Hoặc bỏ qua cửa hàng 3, nên nhu cầu hiện tại sẽ bằng cách chọn tại cửa hàng thứ 2
- Ta sẽ điền vào cách chọn nào cho giá trị lớn hơn

3 (chọn 1)	5(chọn 2)	11 (chọn 1 và 3)			

Tại cửa hàng thứ 4 tiếp theo, với L2<4 nên cách chọn sẽ là

- Chọn 4 + MAX(cửa hàng trong khoảng vị trí 1 đến 2)
- Hoặc bỏ qua 4 và chọn giá trị của cửa hàng 3
- Ta sẽ điền vào cách chọn nào cho giá trị lớn hơn
- Trong trường hợp bằng nhau, bạn điền vào cách nào cũng được

3 (chọn 1)	5(chọn 2)	11 (chọn 1 và 3)	11 (chọn 2 và 4)		

Tại cửa hàng thứ 5 tiếp theo cách chọn sẽ là

- Chọn cửa hàng 5 + MAX(cửa hàng trong khoảng 2 đến 3)
- Hoặc bỏ qua cửa hàng 5, giữ giá trị của cửa hàng 4
- Ta sẽ điền vào cách chọn nào cho giá trị lớn hơn

3 (chọn 1)	5(chọn 2)	11 (chọn 1 và 3)	11 (chọn 2 và 4)	19 (chọn 1,3 và 5)	

Tại cửa hàng thứ 6 tiếp theo cách chọn sẽ là

- Chọn 6 + MAX(cửa hàng trong khoảng 3-4)
- Hoặc bỏ qua cửa hàng thứ 6, giữ lại cách chọn của cửa hàng 5

- Ta sẽ điền vào cách chọn nào cho giá trị lớn hơn

3 (chọn 1)	5(chọn 2)	11 (chọn 1 và 3)	11 (chọn 2 và 4)	19 (chọn 1,3 và 5)	19 (chọn 1,3 và 5)

Nhận xét: Nếu chỉ cần tìm max ta không cần tổ chức thành bảng để tiết kiệm bộ nhớ, chỉ cần lưu các biến tại vi trí cách tối đa l2 vị trí

Câu hỏi

Câu 1. Độ phức tạp theo O-lớn của thuật toán trên trong trường hợp tồi nhất

Câu 2. Cách điền bảng như vậy trong kỹ thuật quy hoạch động được gọi là cách nào? Memoization hay Tabulation?

Câu hỏi 3. Nếu muốn in ra cách chọn, ta cần tổ chức dữ liệu phụ như thế nào?

==========

Bài 2. Max even subsequence : Tìm đoạn con chẵn lớn nhất – các phần tử liên tiếp với tổng giá trị các phần tử trong dãy là số chẵn có tổng giá trị lớn nhất

Đầu vào và đầu ra có dạng như sau

Input	**Output**
8 4 -5 2 4 -8 2 3 1	6

Đầu vào được diễn giải như sau

* Bài này tương tự với maxsub

a_1 a_2 a_3 ... a_n

max

Xét dãy con S_j : a_i ... a_{j-1} $\boxed{a_j}$

Lấy phần tử a_j

S_j liên kết với S_v $(v<j)$ hay S_{j-1} ntn?

$$S_j = \begin{cases} a_j + S_{j-1} & \text{nếu lấy } a_{j-1} \\ a_j & \text{nếu không lấy } a_{j-1} \end{cases}$$

$$\rightarrow S_j = \max(a_j + S_{j-1}, a_j)$$

Sau đó $\rightarrow$ $ans = \max(ans, S[j])$; $j : 1 \rightarrow n$

* Với bài này ta có

S_j = a_j + S_{j-1}

S_j: Chẵn (vì ta muốn nó thế)

a_j: chẵn, lẻ

S_{j-1}: chẵn, $lẻ_{max}$

↳ Sử dụng 2 mảng // S_i^0 → tổng đến i, chẵn

S_i^1 → " lẻ

$$S_j^0 = \begin{cases} a_j + S_{j-1}^0 & , a_j \text{ chẵn} \\ a_j + S_{j-1}^1 & , a_j \text{ lẻ} \end{cases}$$

$$S_j^1 = \begin{cases} a_j + S_{j-1}^1 & , a_j \text{ chẵn} \\ a_j + S_{j-1}^0 & , a_j \text{ lẻ} \end{cases}$$

Vấn đề: Liệu các bài toán con S_{j-1}^1, S_{j-1}^0 ... có tồn tại?

↳ Mảng đánh dấu (2)

$S_{j-1}^1 \rightarrow b^1[j]$ $S_{j-1}^0 \rightarrow b^0[j]$

- Dòng 1 là số lượng phần tử trong dãy ban đầu n, $n<10^6$
- Dòng 2 lần lượt là giá trị các phần tử trong dãy, với giá trị trong khoảng $[-10^6,10^6]$

Tìm và in ra màn hình tổng đoạn con chẵn lớn nhất (nếu có) trong dãy n phần tử trên. Trong trường hợp KHÔNG tồn tại đoạn con chẵn thì in ra là NOT_FOUND (VD. Dãy chỉ gồm 1 số)

Gợi ý:

- Đoạn con trong khoảng i tới j sẽ là các phần tử nằm liên tiếp nhau: $a_i,a_{i+1}, a_{i+2},..,a_j$
- Tổng giá trị của đoạn con là tổng giá trị các phần tử $a_i+a_{i+1}+ a_{i+2}+..+a_j$
- Nếu vét cạn số lượng đoạn con chẵn trong dãy n phần tử thì số lượng phải xét sẽ là $O(n^2)$, nếu n tới cỡ 1tr thì vẫn có thể vét được, nhưng mà thời gian sẽ khá lâu, tính tới phút

Câu hỏi 1. Muốn liệt kê hết tổng giá trị các đoạn con độ dài k trong dãy n phần tử thì thời gian tồi nhất là bao nhiêu theo O-lớn?

VD. Dãy 2,4,-5,6,7 sẽ có các dãy con độ dài 3 là (2,4,-5), (4,-5,6) và (-5,6,7)

Phương án nhanh hơn?

- Nếu thuật toán chỉ yêu cầu tìm đoạn con có tổng lớn nhất thì thuật toán sẽ là :
    - duyệt tuần tự các phần tử từ trái qua phải
    - Đoạn con tổng lớn nhất tới phần tử ai sẽ là
        - Đoạn con mới xuất phát từ a_i
        - Tiếp tục đoạn con từ a_{i-1} trước
    - Vậy ta sẽ chọn là MAX(a_i hoặc a_i+tổng dãy tới a_{i-1})

VD với dãy 2,4,-5,6,-7

Vị trí 1 sẽ là chọn phần tử đầu tiên vì trước đó chưa có giá trị

2				

Vị trí 2 sẽ là MAX(dãy xuất phát từ 2, và dãy từ a1 chứa thêm a2) = MAX(4, 2+4)

2	6			

Vị trí 3 sẽ là MAX(dãy xuất phát từ 3, và dãy từ a2 chứa thêm a3) = MAX(-5, 6+(-5))

2	6	1		

Vị trí 4 sẽ là MAX(dãy xuất phát từ 4, và dãy từ a3 chứa thêm a4) = MAX(6, 6+1)

2	6	1	7	0

Vị trí 5 sẽ là MAX(dãy xuất phát từ 5, và dãy từ a4 chứa thêm a5) = MAX(-7, -7+7)

Sau đó ta duyệt lại dãy lần nữa để tìm max, trong trường hợp này max là 7

Thời gian xử lý là O(n)

Bổ sung thêm yêu cầu tổng số lượng phần tử phải là chẵn?

- Bổ sung thêm 1 bảng tra nữa với giá trị max sẽ là $-\infty$ nếu dãy đó không phải chẵn liệu có được?
- Bổ sung thêm bảng chỉ chứa tổng lẻ lớn nhất tới tổng dãy tới a_{i-1} ? Khi đó đoạn con chẵn tổng lớn nhất tới phần tử a_i sẽ là
    - $-\infty$ vì đoạn con mới xuất phát từ a_i là dãy lẻ
    - Tiếp tục đoạn con lẻ từ a_{i-1} trước
    - Vậy ta sẽ chọn là MAX($-\infty$ hoặc a_i+tổng đoạn lẻ tới a_{i-1})
- Còn đoạn lẻ sẽ được cập nhật thế nào?
    - Là đoạn con xuất phát từ a_i
    - Hoặc tiếp đoạn con chẵn lớn nhất từ a_{i-1}
    - ta chọn MAX của 2 giá trị này

Ví dụ với dãy 4 -5 2 4 -8 2 3 1

Dòng 1 là tổng chẵn lớn nhất

Dòng 2 là tổng lẻ lớn nhất

Với vị trí 1, sẽ ko có tổng chẵn

$-\infty$							
4							

Với vị trí 2

$-\infty$	-1						
4	-5						

Với vị trí 3

$-\infty$	-1	-3					
4	-5	2					

Với vị trí 4

$-\infty$	-1	-3	6				
4	-5	2	4				

Với vị trí 5

−∞	-1	-3	6	-4			
4	-5	2	4	-2			

Với vị trí 6

−∞	-1	-3	6	-4	0		
4	-5	2	4	-2	2		

Với vị trí 7

−∞	-1	-3	6	-4	0	5	
4	-5	2	4	-2	2	3	

Với vị trí 8

−∞	-1	-3	6	-4	0	5	4
4	-5	2	4	-2	2	3	6

Duyệt lại 1 lần hàng 1 để tìm max, trong trường hợp này là 6

Câu hỏi 2. Nếu chỉ cần tìm max ta có thể cắt giảm thao tác hoặc dữ liệu nào để tiết kiệm bộ nhớ

Câu hỏi 3. Nếu cần đưa ra danh sách các phần tử của dãy con, bạn sẽ cần thêm dữ liệu phụ nào?

==========

Bài 3. Nurse – Đếm cách xếp lịch

Trong bệnh viện, mỗi y tá cần làm việc x ngày liên tục rồi mới được nghỉ 1 ngày. Số ngày làm việc x thỏa mãn (K1 <= x <= K2).

- Y tá KHÔNG thể làm việc liên tục quá K2 ngày liên tiếp!
- Y tá cũng không thể làm việc ít hơn K1 ngày liên tiếp

Giả sử cần xếp lịch cho N ngày, hỏi có bao nhiêu cách có thể xếp lịch thảo mãn ràng buộc trên?

Ngày 1 2 3 4 ... n

S_j : số cách chọn nếu j là ngày cuối cùng

2TH — j là ngày nghỉ (O)
\ j là ngày làm việc (W)

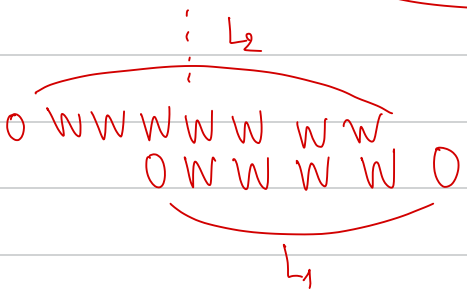
$S_j^O = S_{j-1}^W$ (nếu j là ngày nghỉ, ngày j-1 phải làm việc)

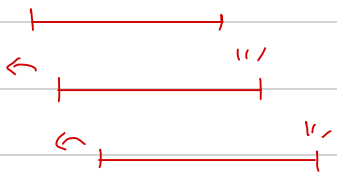
$S_j^W = S_{j-k}^W$, $k: K_1 \to K_2$

↳ viết ntn thì ko tính đc!

$= \sum_{j=L_1}^{L_2} S_{i-j}^O$ (khi tính) O(1)

thực chất là dịch tổng sang phải → 1





(Chỉ cần trừ đi phần đầu ; thêm phần cuối)

Cách khác (Đỗ Hải)

Cần phải nêu đc thành CT truy vấn thì mới được!

↳ Gọi dp [i] [j]

↓ số ngày đã làm việc tính đến ngày i

↓ tính đến ngày i

$dp[i][0] = \sum dp[i-1][K_1 \dots K_2]$

$dp[i][j] = dp[i-1][j-1]$

đến ngày i, đã làm j ngày

đến ngày i-1, đã làm

Đầu vào: Gồm 3 số nguyên dương lần lượt là N, K1, K2 (N <= 1000, K1 < K2 < 400)

Đầu ra: Số cách xếp lịch khác nhau: M % (109+7)

Input	**Output**
6 2 3	4

Với 6 ngày và đợt làm việc phải kéo dài ít nhất 2 ngày và tối đa 3 ngày thì các cách xếp có thể là

Cách 1: mỗi đợt 2 ngày						
Cách 2. mỗi đợt 2 ngày						
Cách 3: Đợt 2 và 3 ngày						
Cách 4: Đợt 2 và 3 ngày						

Màu xanh là ngày làm việc và màu trắng là ngày nghỉ

Gợi ý:

- Nếu N=1 thì chỉ có 1 cách bố trí là ngày đó là ngày nghỉ
- Nếu 1<N<K1 thì KHÔNG có cách bố trí hợp lệ
- Nếu N=K1 thì chỉ có 1 cách bố trí
- Nếu N=K+1 thì có 2 cách bố trí
- KHÔNG thể có 2 ngày nghỉ liên tiếp

Nếu N>K+1 thì đếm số cách thế nào?

- Nếu ngày trước là cuối cùng của đợt làm việc thì ngày tiếp theo bắt buộc phải là 1 ngày nghỉ
- Ngày hiện tại thứ i sẽ có 2 phương án
    - Nó là ngày nghỉ hoặc
    - Nó là ngày cuối cùng của đợt làm việc

(1) Nếu ngày thứ i là ngày nghỉ thì số phương án sẽ là số cách trong trường hợp ngày i-1 là ngày làm việc cuối cùng của đợt làm việc

(2) Còn nếu ngày thứ i là ngày làm việc cuối cùng? Thì nó sẽ phải được nối tiếp từ các cách trong đó ngày i-x là ngày nghỉ trước đó (với K1<=x<=K2)

Số phương án sắp xếp sẽ là tổng số cách của (1) + (2)

Lấy ví dụ với đầu vào N = 10 và K1=2, K2=4

Loại ngày	i=1	i=2	3	4	5	6	7	8	9	10
Ngày nghỉ (1)	1	0	1	2	2					
Ngày làm việc cuối cùng (2)	0	1	1+1=2	1+1=2	2					

Cột với i=1 thì sẽ là

- 1 phương án nếu ngày đó là ngày nghỉ và
- 0 phương án nếu ngày đó là ngày cuối của đợt làm việc

Cột với i=2 sẽ là

- 0 phương án nếu là ngày nghỉ vì KHÔNG thể nghỉ 2 ngày liên tiếp được, và cũng KHÔNG tồn tại cách mà ngày i=2-1 là ngày làm việc cuối cùng
- 1 phương án Nếu đây là ngày cuối cùng của đợt làm việc (đợt 2 ngày)

Cột với i=3 sẽ là

- 1 phương án ngày nghỉ thứ 3 vì trước đó có 1 phương án cho i=3-1 = 2 là ngày làm việc cuối cùng
- 1 phương án ngày làm việc (sau đợt nghỉ của ngày thứ 3-2) + 1 phương án cho chuỗi 3 ngày làm việc liên tiếp

Cột với i=4 sẽ là

- 2 phương án cho ngày thứ 4 là ngày nghỉ vì ta có 2 phương án cho ngày 4-1=3 là ngày làm việc cuối cùng
- 1 Phương án cho chuỗi 4 ngày làm việc liên tục + 1 phương án cho 3 ngày làm việc và ngày 4-3=1 là ngày nghỉ

Vậy cách check số phương án trong trường hợp ngày i là ngày cuối cùng của đợt làm việc sẽ là tổng số cách mà ngày i-x là ngày nghỉ với K1<=x<=K2

Với i=4 thì i-x sẽ là

- 4-4=0: KHÔNG tồn tại
- 4-3 = 1: có 1 phương án
- 4-2=2: Không có phương án

Cột với i=5 sẽ là

- 2 Phương án nếu ngày thứ 5 là ngày nghỉ
- Nếu là ngày cuối của đợt làm việc thì cần check với
    - 5-4 = 1 Có 1 phương án
    - 5-3=2 Không có phương án
    - 5-2=3 Có 1 phương án
- Tổng là 2

Tương tự cho các cột tiếp theo

Câu hỏi 1. Độ phức tạp tính toán của phương án trên?

Câu hỏi 2. Nếu cột thứ i = x (K1<x<K2) và ngày đó là ngày cuối của đợt làm việc thì công thức tìm số cách sẽ là

==========

Bài 4. Warehouse (PP 2 chiều)

Có N kho hàng chứa loại hàng hóa giống nhau nằm trên một đường thẳng. Một xe tải vận chuyển muốn lấy hàng ở một số nhà kho (trong tổng số N nhà kho) trên. Để cho đơn giản ta giả sử

- Vị trí x_i của mỗi nhà kho lần lượt sẽ là 1,2,3,4,…,N
- Mỗi nhà kho có chứa khối lượng hàng hóa là a_i ($1 <= a_i <= 10$)
- Thời gian để chất hết hàng lên xe của nhà kho thứ i sẽ là t_i ($1 <= t_i <= 10$)

Giả sử trọng lượng có thể chở của xe là vô hạn, tuy nhiên xe sẽ bị giới hạn bởi:

- Tổng thời gian lấy hàng T, và
- Khoảng cách hàng tiếp theo x_{i+1} từ kho hiện tại x_i KHÔNG vượt quá D ($x_{i+1} - x_i <= D$)
- Xe tải có thể bắt đầu lấy hàng từ 1 kho bất kỳ, mỗi lần lấy sẽ lấy hết số hàng trong kho lên xe.
- Kho đã lấy hàng rồi sẽ KHÔNG được quay lại.

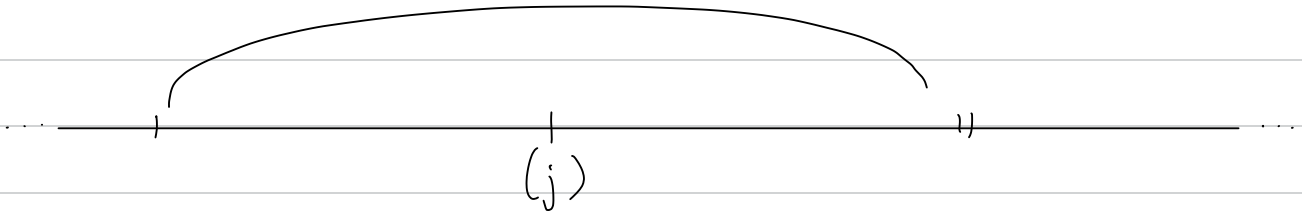
Hãy tìm tổng trọng lượng lớn nhất lấy được từ chuỗi kho hàng ($x_1<x_2<x_3<..<x_k$) đã cho ở trên.

Input:	**Output:**
6 6 2 6 8 5 10 11 6 1 2 2 3 3 2	24

Mô tả:

- Dòng đầu tiên sẽ là N, T, D (1<=N<=1000, 1<=T<=100, 1<=D<=10)
- Dòng tiếp là khối lượng hàng hóa trong các kho a_i
- Dòng cuối là thời gian để bốc hàng tại mỗi kho t_i

Gợi ý.



Xét bài toán con, ở ràng buộc T

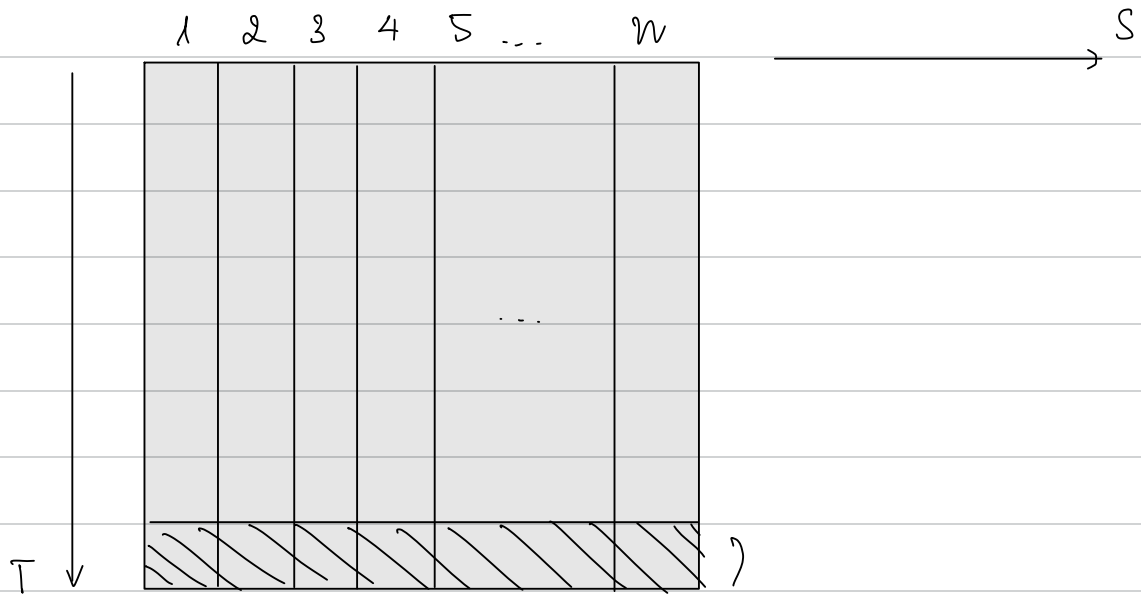
$$S[j] = a_j + \max\left(S_{j-D}^{j-1}\right) = a_j + \max_{i=j-D..j-1}(S_i)$$

$T = 10$; $t_j = 2$; 8

Thêm thời gian T

$$S_{j,T} = a_j + \max\left(S_{j-D}^{j-1}{}_{i,\,T-T_j}\right)$$

$$= a_j + \max_{(i=j-D\,\ldots\,j-1)}\left(S_{i,\,T-T_j}\right)$$



- Bài này có thể sử dụng ý tưởng quy hoạch động khá giống của bài số 3 – Nurse
- Nếu x_{i+1} là nhà kho hiện tại thì nó có thể được tiếp tục từ 1 trong các nhà kho ở vị trí $x_i+1<=x_{i+1}=<=x_i+D$
- Vì ưu tiên tổng trọng lượng lớn nhất, và khối lượng hàng hóa đều là số dương nên ta sẽ chọn nhà kho trước đó cũng là có tổng lớn nhất

Ví dụ. D=3 và tổng khối lượng lấy được từ các nhà kho trước đó như sau

Vị trí nhà kho	1	2	3	4	$X_{i+1}=5$
Tổng khối lượng hàng hóa	12	15	8	7	

Nếu ta dừng ở nhà kho số 5 thì ta có thể đến đó từ nhà kho 2,3,4 và ta sẽ chọn chuỗi trước số 5 là nhà kho số 2

- Ta sẽ xây dựng thuật toán ngược lần lượt tìm nhà kho trước nhà kho hiện tại trong chuỗi kiểu như trên

Với ví dụ đầu vào là

6 6 2
6 8 5 10 11 6
1 2 2 3 3 2

N = 6, T = 6, D = 2

Hàng ngang sẽ là thời gian T và cột lần lượt sẽ là trường hợp nhà kho thứ i được chọn tại thời điểm hiện tại

Ta sẽ xây dựng bảng 6x6 dạng

Thời gian	$X_i=1$	$X_i=2$	$X_i=3$	$X_i=4$	$X_i=5$	$X_i=6$
T=1						
T=2						
T=3						
T=4						
T=5						
T=6						

Với thời gian T=1 thì chỉ có nhà kho nào có thời gian bốc xếp là 1 ta mới điền vào khối lượng, còn ngược lại sẽ điền vào là 0 (vì KHÔNG đủ thời gian)

Thời gian	$X_i=1$	$X_i=2$	$X_i=3$	$X_i=4$	$X_i=5$	$X_i=6$
T=1	6	0	0	0	0	0

Với thời gian T>1 thì ta sẽ tìm và điền vào tổng giá trị lớn nhất giữa của

- a_i nếu $T>=t_i$
- a_i + MAX tổng kho thứ j trong các kho hàng trong khoảng ($x_{i-1}+1<=x_i=<=x_{i-1}+D$) nếu $T>=t_i + t_j$
- Tra kho thứ j trong cột $T' = T - t_i$

Với thời gian T=2

Thời gian	X_i=1	X_i=2	X_i=3	X_i=4	X_i=5	X_i=6
T=1	6	0	0	0	0	0
T=2	6	8	5	0	0	6

Ô tương ứng với xi=1 là 6 vì KHÔNG có kho hàng nào trước 1
Ô tương ứng với xi=2,3,6 sẽ là tổng hàng hóa tại kho đó vì các kho này có thời gian bốc hàng là 2 (vừa đủ bốc, KHÔNG thể lấy thêm ở kho nào trước đó)

Ô tương ứng xi=3,4 sẽ là 0 vì chưa đủ thời gian bốc hàng

Với thời gian T=3

Thời gian	X_i=1	X_i=2	X_i=3	X_i=4	X_i=5	X_i=6
T=1	6	0	0	0	0	0
T=2	6	8	5	0	0	6
T=3	6	14	11	10	11	6

Ô tương ứng với xi=1 là 6 vì KHÔNG có kho hàng nào trước 1

Ô tương ứng với xi=4,5 sẽ là tổng hàng hóa tại kho đó vì các kho này có thời gian bốc hàng là 3 (vừa đủ bốc, KHÔNG thể lấy thêm ở kho nào trước đó)

Ô tương ứng với xi=2 sẽ là 8 + 6 = tổng hàng hóa tại kho 1 vì T=3 đủ bốc tại kho 2 và kho 1. Bạn tra lại hàng T’ = T-2 = 1.

Ô tương ứng với xi=3 sẽ là 5 + 6 = tổng hàng hóa tại kho 1 vì T=3 đủ bốc tại kho 2 và kho 1 (kho 3 và kho 1 cách nhau 2=D), Bạn tra lại hàng T’ = T-2 = 1.

Ô tương ứng với xi=6 vẫn là 6 vì KHÔNG có lựa chọn bốc tại kho trước. Bạn tra lại hàng T’ = T-2 = 1 thì các kho lân cận có thể của ô này đều là 0

Với thời gian T=4

Thời gian	X_i=1	X_i=2	X_i=3	X_i=4	X_i=5	X_i=6
T=1	6	0	0	0	0	0
T=2	6	8	5	0	0	6
T=3	6	14	11	10	11	6
T=4	6	14	13	10	11	6

Ô tương ứng với xi=1 là 6 vì KHÔNG có kho hàng nào trước 1

Ô tương ứng với xi=4,5 sẽ là tổng hàng hóa tại kho đó vì các kho này có thời gian bốc hàng là 3 và KHÔNG thể lấy thêm ở kho nào trước đó do thời gian còn lại là 4-3=1 không đủ. Tra lại hàng T’ = 4-3=1 thì các kho lân cận có thể đều là 0

Ô tương ứng với xi=2 sẽ là 8 + 6 = lấy từ kho 1 (vì KHÔNG còn kho nào nữa để chọn)

Ô tương ứng với xi=3 sẽ là 5 + MAX(6 = tổng hàng hóa tại kho 1, 8 = tổng hàng lấy từ kho 2). Tra hàng tương ứng T' = 4-2=2.

Ô tương ứng với xi=6 vẫn là 6 vì KHÔNG có lựa chọn bốc tại kho trước. T' = 2 và các ô có teher của nó đều là 0

Với thời gian T=5

Thời gian	X_i=1	X_i=2	X_i=3	X_i=4	X_i=5	X_i=6
T=1	6	0	0	0	0	0
T=2	6	8	5	0	0	6
T=3	6	14	11	10	11	6
T=4	6	14	13	10	11	6
T=5	6	14	19	18	16	17

Ô tương ứng với xi=1 là 6 vì KHÔNG có kho hàng nào trước 1

Ô tương ứng với xi=2 sẽ là 8 + 6 = lấy từ kho 1 (vì KHÔNG còn kho nào nữa để chọn)

Ô tương ứng với xi=3 sẽ là 5 + MAX(6 = tổng hàng hóa tại kho 1, 14 = tổng hàng lấy từ kho 2). Tra hàng tương ứng T' = 5-2=3.

Ô tương ứng với xi=4 sẽ là 10 + MAX(8 = tổng hàng hóa tại kho 2, 5 = tổng hàng lấy từ kho 3). Tra hàng tương ứng T' = 5-3=2.

Ô tương ứng với xi=5 sẽ là 11 + MAX(5 = tổng hàng hóa tại kho 3). Tra hàng tương ứng T' = 5-3=2.

Ô tương ứng với xi=6 sẽ là 6 + MAX(10 = tổng hàng hóa tại kho 4, 11 = tổng hàng lấy từ kho 5). Tra hàng tương ứng T' = 5-2=3.

Với thời gian T=6

Thời gian	X_i=1	X_i=2	X_i=3	X_i=4	X_i=5	X_i=6
T=1	6	0	0	0	0	0
T=2	6	8	5	0	0	6
T=3	6	14	11	10	11	6
T=4	6	14	13	10	11	6
T=5	6	14	19	18	16	17
T=6	6	14	19	24	22	17

Câu hỏi 1. Hãy thử lại với đầu vào sau

6 8 3
6 8 5 10 11 6
1 2 2 3 3 2

Thời gian	X_i=1	X_i=2	X_i=3	X_i=4	X_i=5	X_i=6
T=1						
2						
3						
4						
5						
6						
7						
8						

Câu hỏi 2. Độ phức tạp tính toán theo thời gian của thuật toán trên?

Câu hỏi 3. Nếu muốn in ra cả chuỗi nhà kho, bạn sẽ làm thêm thế nào?