113 學年度資訊學科能力競賽臺南一中校內複選試題本

競賽規則

1. 競賽時間：2024/09/27 13:00 ~ 17:00，共 4 小時。
2. 本次競賽試題共 6 題，每題皆有子任務。
3. 為了愛護地球，本次競賽題本僅提供電子檔，不提供紙本。
4. 每題的分數為該題所有子任務得分數加總；單筆子任務得分數為各筆繳交在該筆得到的最大分數。
5. 本次初選比照南區賽提供記分板，複選比照全國賽不提供記分板。
6. 全部題目的輸入皆為標準輸入。
7. 全部題目的輸出皆為標準輸出。
8. 所有輸入輸出請嚴格遵守題目要求，多或少的換行及空格皆有可能造成裁判系統判斷為答案錯誤。
9. 每題每次上傳間隔為 120 秒，裁判得視情況調整。
10. 所有試題相關問題請於競賽系統中提問，題目相關公告也會公告於競賽系統，請密切注意。
11. 如有電腦問題，請舉手向監考人員反映。
12. 如有如廁需求，須經過監考人員同意方可離場。
13. 不得攜帶任何參考資料，但競賽系統上的參考資料可自行閱讀。
14. 不得自行攜帶隨身碟，如需備份資料，請將資料儲存於電腦 D 槽。
15. 競賽中請勿交談。請勿做出任何會干擾競賽的行為。
16. 如需使用 C++ 的 `std::cin` 或 `std::cout` 可將以下程式碼插入 main function 以及將 `endl` 取代為 `'\n'` 來優化輸入輸出速度。唯須注意不可與 `cstdio` 混用。

```
std::ios::sync_with_stdio(false);
std::cin.tie(nullptr);
```

A. 水題

Problem ID: Water
Time Limit: 5.0s
Memory Limit: 512MiB

首先我們知道，死庫水跟水庫彼此只差了一個字。再者，Samuel 非常喜歡死庫水，想當然爾，他也相當喜歡水庫。

在某個遙遠的國度，存在著彼此相通的 n 個水庫，而相鄰的兩個水庫間有著一個閘門，因此共有 $n-1$ 個閘門。第 1 個閘門連接著第 1 個水庫和第 2 個水庫，第 2 個閘門連接著第 2 個水庫和第 3 個水庫，以此類推，第 i 個閘門連接著第 i 個水庫和第 $i+1$ 個水庫，而所有的閘門在一開始都是關閉的。這 n 水庫的高度是由 1 到 n 遞減的，即對於所有 i，都有第 i 個水庫的高度高於第 $i+1$ 個水庫。這表示若第 i 個閘門是開啟的，則第 i 個水庫中的水會全數流向第 $i+1$ 個水庫，且不會有任何的水留在第 i 個水庫中，而每個水庫的容量都可視為無限大，並且每個水庫的初始水量都為 0 單位。

Samuel 是這 n 個水庫的管理員，他喜歡水庫卻相當懶惰，因此他決定把水庫交給你來管理，他會告訴你共 q 個發生的事件，要請你幫他處理，而一共有以下三種事件：
$1\ l\ r\ k$：第 l 個水庫到第 r 個水庫下了場大雨，使得這 $r-l+1$ 個水庫各增加了 k 單位的水
$2\ x$：打開第 x 個閘門
$3\ y$：領導來視察，並詢問你第 y 個水庫中有多少單位的水

最後，在 q 個事件都結束後，請你告訴 Samuel 這 n 個水庫中分別含有多少單位的水。

– 輸入 –

第一行包含兩個正整數 n, q，分別代表水庫和事件的數量。
接下來有 q 行，每行代表一個事件，而每一個事件都會以下列三種格式之一呈現：
$1\ l\ r\ k$
$2\ x$
$3\ y$

– 輸出 –

若有詢問某水庫中含有多少單位的水，則應輸出一個整數並換行。
最後一行輸出 n 個整數，代表 n 個水庫中各有多少單位的水。

– 輸入限制 –

- $1 \leq n, q \leq 10^6$
- $1 \leq l \leq r \leq n$
- $1 \leq k \leq 10^5$
- $1 \leq x \leq n - 1$
- $1 \leq y \leq n$

– 子任務 –

編號	分數	額外限制
1	0	範例輸入輸出
2	7	$1 \leq n,\ q \leq 500$
3	19	只有第 1 種事件
4	13	只有第 1 和第 3 種事件
5	29	只有第 1 和第 2 種事件
6	21	$1 \leq n,\ q \leq 5 \times 10^5$
7	11	無額外限制

– 範例輸入 1 –

```
5 5
1 2 4 3
2 3
2 2
1 1 3 2
2 4
```

– 範例輸出 1 –

```
2 0 0 0 13
```

– 範例輸入 2 –

```
3 4
1 2 3 2
2 2
3 3
1 1 2 6
```

– 範例輸出 2 –

```
4
6 0 10
```

– 範例說明 –

範例測資 1 說明:
範例測資 1 中，第 2, 3 和 4 個閘門都被打開了，因此第 2, 3 和 4 個水庫的水都會流向第 5 個水庫，因此第 5 個水庫中最後共有 13 單位的水。
同時，由於第 1 個閘門並未打開，因此第 1 個水庫在大雨中獲得的水會留在裡面，最後共有 2 單位的水。

範例測資 2 說明:
範例測資 2 中，詢問第 3 個水庫的水量時，第 2 個閘門已經打開，因此第 2 個水庫的水會流向第 3 個水庫，使得第 3 個水庫共有 4 單位的水。
當 4 個事件都結束後，第 1 個閘門並未打開，因此第 1 個水庫在大雨中獲得的水會留在裡面，最後共有 6 單位的水，而第 3 個水庫中則有 10 單位的水。

B. 你好我的世界

Problem ID: HelloMinecraft
Time Limit: 1.0s
Memory Limit: 128MiB

今年正好是「我的世界 (Minecraft)」從開始開發至今的 15 周年，身為「我的世界」的忠實老玩家與一名新手的遊戲開發者，Samuel 開發的第一款遊戲名為「你好我的世界」，靈感正是來自於原版的「我的世界」。

這個遊戲的主要目標就是在世界中合成各種不同的物品。
當然了，還要有蓋房子、打怪、PVP、跑酷等等，只不過對於 Samuel 這個遊戲開發新手，這些功能還太複雜了，可能在看不見的將來才會加入遊戲中。

回到正題，在這款遊戲中有 n 種不同物品可以合成，編號為 $1, 2, ..., n$，而除此之外還有 m 種不同的功能方塊可以製造，編號為 $1, 2, ..., m$。
所謂功能方塊是指用於輔助合成物品，但不會被消耗的道具，也就是說取得一次就可以無限次使用了（對應原版我的世界遊戲中的合成台、熔爐等）。

為了合成各種不同的物品，你好我的世界遊戲中對於每種物品 i 皆有 way_i 種的合成方式，對於物品 i 的第 j 種合成方式會需要使用 $a_{i,j}$ 種物品與 $b_{i,j}$ 種功能方塊並花費 $c_{i,j}$ 秒。更嚴謹的說有 $a_{i,j}$ 組需求 $(x_{i,j,k}, y_{i,j,k})$，代表會消耗 $y_{i,j,1}$ 個物品 $x_{i,j,1}$、$y_{i,j,2}$ 個物品 $x_{i,j,2}$、$\ldots$、$y_{i,j,a_{i,j}}$ 個物品 $x_{i,j,a_{i,j}}$ 來製造 1 個物品 i，並且同時需要有 $b_{i,j}$ 種指定的功能方塊 $z_{i,j,1}, z_{i,j,2}, \ldots, z_{i,j,b_{i,j}}$。
再次強調功能方塊並不會消耗。
請注意每種合成方式的 $a_{i,j}, b_{i,j}, c_{i,j}$ 皆可能為 0，$a = 0$ 代表不會消耗其他物品，$b = 0$ 代表不需要任何功能方塊，$c = 0$ 代表該次合成不需要時間。也特別提醒可能出現「需要物品 i 來合成物品 i」的合成方式。

另外功能方塊的取得方法，雖然正常來說也必須透過合成，但為了簡化遊戲，在早期版本直接設定可以花費 t_i 即可取得功能方塊 i。
除此之外，請注意在遊戲中同時只能做一件事情，也就是說不論是合成物品或者取得功能方塊，都必須在經過其所需時間之後才能進行下一次的合成物品或取得功能方塊。

現在，身為 speedrunner 的你想要挑戰用最快的速度達到遊戲中的里程碑，因此你想知道對於每一種物品，理論上最快可以取得一件該物品的時間為何？
而如果該物品無論如何都無法合成的話，請輸出 -1。另外如果最小取得時間大於等於 10^{15} 秒的話，由於沒有那麼多時間玩遊戲，因此也視為無法拿到，輸出 -1。

– 輸入 –

輸入的第一行包含兩個正整數 n, m，分別代表物品的種類數、功能方塊的種類數。
第二行包含 n 個正整數 $way_1, way_2, ..., way_n$，代表每種物品的合成方法數。
第三行包含 m 個非負整數 $t_1, t_2, ..., t_m$，代表每種功能方塊的取得所需時間。注意由於 m 可能為 0，因此該行可能為空。
接下來對於每種物品 i 有 way_i 種物品的合成方式，每種合成方式包含三行。其中的第一行的開頭有一個非負整數 $a_{i,j}$，接著有 $a_{i,j}$ 組正整數 $x_{i,j,k}$ $y_{i,j,k}$，代表消耗的物品種類編號、數量；其中的第二行的開頭有一個非負整數 $b_{i,j}$，接著有 $b_{i,j}$ 個正整數 $z_{i,j,k}$，代表需要的功能方塊編號；其中的第三行有一個非負整數 $c_{i,j}$，代表此種合成方式需要的時間。

簡而言之，輸入格式如下。

$n\ m$
$way_1\ way_2\ \ldots\ way_n$
$t_1\ t_2\ \ldots\ t_m$
$a_{1,1}\ x_{1,1,1}\ y_{1,1,1}\ x_{1,1,2}\ y_{1,1,2}\ \cdots\ x_{1,1,a_{1,1}}\ y_{1,1,a_{1,1}}$
$b_{1,1}\ z_{1,1,1}\ z_{1,1,2}\ \cdots\ z_{1,1,b_{1,1}}$
$c_{1,1}$
$\ldots$
$a_{1,way_1}\ x_{1,way_1,1}\ y_{1,way_1,1}\ x_{1,way_1,2}\ y_{1,way_1,2}\ \cdots\ x_{1,way_1,a_{1,way_1}}\ y_{1,way_1,a_{1,way_1}}$
$b_{1,way_1}\ z_{1,way_1,1}\ z_{1,way_1,2}\ \cdots\ z_{1,way_1,b_{1,way_1}}$
c_{1,way_1}
$a_{2,1}\ x_{2,1,1}\ y_{2,1,1}\ x_{2,1,2}\ y_{2,1,2}\ \cdots\ x_{2,1,a_{2,1}}\ y_{2,1,a_{2,1}}$
$b_{2,1}\ z_{2,1,1}\ z_{2,1,2}\ \cdots\ z_{2,1,b_{2,1}}$
$c_{2,1}$
$\ldots$
$a_{2,way_2}\ x_{2,way_2,1}\ y_{2,way_2,1}\ x_{2,way_2,2}\ y_{2,way_2,2}\ \cdots\ x_{2,way_2,a_{2,way_2}}\ y_{2,way_2,a_{2,way_2}}$
$b_{2,way_2}\ z_{2,way_2,1}\ z_{2,way_2,2}\ \cdots\ z_{2,way_2,b_{2,way_2}}$
c_{2,way_2}
$\ldots$
$a_{n,1}\ x_{n,1,1}\ y_{n,1,1}\ x_{n,1,2}\ y_{n,1,2}\ \cdots\ x_{n,1,a_{n,1}}\ y_{n,1,a_{n,1}}$
$b_{n,1}\ z_{n,1,1}\ z_{n,1,2}\ \cdots\ z_{n,1,b_{n,1}}$
$c_{n,1}$
$\ldots$
$a_{n,way_n}\ x_{n,way_n,1}\ y_{n,way_n,1}\ x_{n,way_n,2}\ y_{n,way_n,2}\ \cdots\ x_{n,way_n,a_{n,way_n}}\ y_{n,way_n,a_{n,way_n}}$
$b_{n,way_n}\ z_{n,way_n,1}\ z_{n,way_n,2}\ \cdots\ z_{n,way_n,b_{n,way_n}}$
c_{n,way_n}

– 輸出 –

輸出 n 行，每行包含一個非負整數，第 i 行代表對於物品 i，取得 1 個所需的最短時間，對於無法合成或所需最短時間大於等於 10^{15} 的物品輸出 -1。

– 輸入限制 –

- $1 \leq n \leq 128$
- $0 \leq m \leq 12$
- $1 \leq way_i \leq 10$
- $0 \leq t_i \leq 10^9$
- $0 \leq a_{i,j} \leq min(n, 10)$
- $0 \leq b_{i,j} \leq m$
- $0 \leq c_{i,j} \leq 100$
- $1 \leq x_{i,j,k} \leq n$
- $1 \leq y_{i,j,k} \leq 10$
- $1 \leq z_{i,j,k} \leq m$
- 對於任意 $d \neq e$ 有 $x_{i,j,d} \neq x_{i,j,e}$ 與 $z_{i,j,d} \neq z_{i,j,e}$

– 子任務 –

編號	分數	額外限制
1	0	範例輸入輸出
2	10	所有 $a_{i,j} = b_{i,j} = 0$ 且 $m \leq 10$
3	14	所有 $a_{i,j} = 0$ 且 $m \leq 10$
4	14	所有 $a_{i,j} = 1, b_{i,j} = 0$ 且 $m \leq 10$
5	18	所有 $a_{i,j} = 1$ 且 $m \leq 10$
6	20	所有 $way_i = 1$ 且 $m \leq 10$
7	12	$m \leq 10$
8	12	無額外限制

– 範例輸入 1 –

```
3 0
2 2 1

0
0
5
1 2 1
0
1
0
0
2
1 1 2
0
0
2 1 2 2 2
0
10
```

– 範例輸出 1 –

```
3
2
20
```

– 範例輸入 2 –

```
3 2
2 2 1
15 20
0
2 1 2
5
1 2 1
0
1
0
0
2
1 1 2
1 2
0
2 1 2 2 2
2 2 1
10
```

– 範例輸出 2 –

```
3
2
55
```

– 範例輸入 3 –

```
2 0
1 1

1 2 1
0
0
1 1 1
0
0
```

– 範例輸出 3 –

```
-1
-1
```

– 範例說明 –

範例測資 1 說明:
範例測資 1 中有 3 種物品、0 種功能方塊，其中物品 1 包含 2 種合成方式、物品 2 包含 2 種合成方式、物品 3 包含 1 種合成方式，以下為範例測資 1 中之資訊所整理之表格。

物品1	合成方式1	所需物品種類數	0	物品編號與數量		
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	5			
	合成方式2	所需物品種類數	1	物品編號與數量	物品2x1個	
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	1			
物品2	合成方式1	所需物品種類數	0	物品編號與數量		
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	2			
	合成方式2	所需物品種類數	1	物品編號與數量	物品1x2個	
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	0			
物品3	合成方式1	所需物品種類數	2	物品編號與數量	物品1x2個	物品2x2個
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	10			

最快合成出物品 3 的方法如下。
首先花費 8 秒執行 4 次物品 2 的合成方式 1，取得 4 個物品 2。
再來花費 2 秒執行 2 次物品 1 的合成方式 2，消耗 2 個物品 2 取得 2 個物品 1，此時有 2 個物品 1、2 個物品 2。
最後花費 10 秒執行 1 次物品 3 的合成方式 1，消耗 2 個物品 1、2 個物品 2 取得 1 個物品 3。
總共花費 $8 + 2 + 10 = 20$ 秒，可以證明沒有更快取得物品 3 的方法。

範例測資 2 說明:

範例測資 2 中有 3 種物品、2 種功能方塊，其中物品 1 包含 2 種合成方式、物品 2 包含 2 種合成方式、物品 3 包含 1 種合成方式，另外功能方塊 1 需要花費 15 秒取得、功能方塊 2 需要花費 20 秒取得，以下為範例測資 2 中之資訊所整理之表格。

物品1	合成方式1	所需物品種類數	0	物品編號與數量		
		所需功能方塊數	2	功能方塊編號	1	2
		合成所需時間	5			
	合成方式2	所需物品種類數	1	物品編號與數量	物品2×1個	
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	1			
物品2	合成方式1	所需物品種類數	0	物品編號與數量		
		所需功能方塊數	0	功能方塊編號		
		合成所需時間	2			
	合成方式2	所需物品種類數	1	物品編號與數量	物品1×2個	
		所需功能方塊數	1	功能方塊編號	2	
		合成所需時間	0			
物品3	合成方式1	所需物品種類數	2	物品編號與數量	物品1×2個	物品2×2個
		所需功能方塊數	2	功能方塊編號	2	1
		合成所需時間	10			

最快合成出物品 3 的方法如下。

首先花費 8 秒執行 4 次物品 2 的合成方式 1，取得 4 個物品 2。

再來花費 2 秒執行 2 次物品 1 的合成方式 2，消耗 2 個物品 2 取得 2 個物品 1，此時有 2 個物品 1、2 個物品 2。

接著花費 20 秒取得功能方塊 2，再花費 15 秒取得功能方塊 1。最後花費 10 秒執行 1 次物品 3 的合成方式 1，在已經取得功能方塊 1 與功能方塊 2 的情況下，消耗 2 個物品 1、2 個物品 2 取得 1 個物品 3。

總共花費 $8 + 2 + 20 + 15 + 10 = 55$ 秒，可以證明沒有更快取得物品 3 的方法。

範例測資 3 說明:

範例測資 3 中有 2 種物品、0 種功能方塊，其中物品 1 包含 1 種合成方式、物品 1 包含 1 種合成方式。

可以證明無法合成出物品 1 與物品 2 的任一種。

C. 神魔之塔

Problem ID: Tower
Time Limit: 1.0s
Memory Limit: 128MiB

眾所周知，神魔之塔是一款發行已久的遊戲，而角色的強度越來越誇張，在遊戲剛推出時，毒龍還需要靠伊登隊慢慢磨，而現在新角色隨便手轉傷害便會破兆，但這都不是重點，重點是在某片遙遠的大陸上，也存在著這麼一座神魔之塔。

這座神魔之塔其實早期也不能稱之為「塔」，因為他最初只有一層，但第一層的神魔覺得這樣太廢了，因此決定擴建神魔之塔到 n 層。每層神魔之塔在蓋好後，其中會孕育一顆魔胎，而這顆魔胎的初始戰力為 1，但這麼弱的神魔實在是太遜咖了，因此在魔胎即將孵化之時，在他樓下的所有神魔都會幫他進行洗禮以提升他的戰力，而由於住的越高獲得的視野越好，因此住在較高樓層的神魔需要幫忙洗禮較多次。我們稱一次強度為 k 的洗禮，會使得受洗神魔的戰力變為原本的 k 倍，則住在第 i 層的神魔幫第 j 層的神魔洗禮時，需要進行 i 次強度為 $gcd(i,j)$ 的洗禮。

值得注意的是，由於第一層的神魔誕生時，沒有樓下的其他神魔幫忙洗禮，因此他的戰力只有 1。同時，我們也可以預見，位於質數層的神魔即使有其他神魔的洗禮，戰力也只會有 1，但正如那句老話：「人生全靠投胎，失敗只能重開。」想來對於神魔而言，這句話也是成立的。

然而，富有正義感的城之內覺得這樣會讓神魔之塔的整體戰力太過強大，因此為了限制神魔之塔，他決定翻開覆蓋的陷阱卡「六芒星的束縛」，該卡的作用會讓過去進行的某些特定強度的洗禮無效，只留下其中 m 種強度分別為 $c_1, c_2, ..., c_m$ 的洗禮仍然有效。也就是說，若過去某神魔受到了這 m 種強度以外的洗禮，則該洗禮對神魔戰力的影響將會消失。

神魔之塔的眾神魔在受到「六芒星的束縛」影響後，都感受到了自身戰力或多或少的降低，因此，為了反抗，他們決定融合第 a 樓到第 b 樓的神魔，以產生一個戰力超群的大神魔 Ex，而他的戰力會是被融合的所有神魔戰力的乘積。但由於事關重大，因此神魔們希望進行 q 次模擬融合，第 i 次模擬會融合第 a_i 樓到第 b_i 樓的所有神魔，請你輸出該次融合產生的大神魔 Ex 的戰力會有多少。若答案過大，則對 998244353 取模。

– 輸入 –

輸入有多行。
第一行包含三個正整數 n, m, q ，意義如題目所述。
第二行包含 m 個正整數 $c_1, c_2, ..., c_m$ ，意義如題目所述。
接下來一共 q 行，每行包含兩個正整數，其中第 i 行包含 a_i, b_i ，意義如題目所述。

– 輸出 –

輸出一共 q 行，其中第 i 行輸出第 i 次模擬產生的大神魔 Ex 戰力為多少。
請注意，答案需對 998244353 取模。

– 輸入限制 –

- $1 \leq n, q \leq 10^5$
- $1 \leq m \leq 100$
- $1 \leq a_i \leq b_i \leq n$
- $1 \leq c_i \leq n$

– 子任務 –

編號	分數	額外限制
1	0	範例輸入輸出
2	17	$1 \leq n \leq 500$
3	29	保證所有 c_i 滿足 $\frac{n}{2} < c_i \leq n$
4	54	無額外限制

– 範例輸入 1 –

```
10 2 1
2 5
10 10
```

– 範例輸出 1 –

```
282066941
```

– 範例輸入 2 –

```
10 5 1
2 3 4 5 8
6 8
```

– 範例輸出 2 –

```
113246208
```

– 範測說明 –

範例測資 1 中，第 10 層的神魔會受到的洗禮如下：
第 1 層的神魔進行 1 次強度為 $gcd(1,10)=1$ 的洗禮，但該強度的洗禮無效。
第 2 層的神魔進行 2 次強度為 $gcd(2,10)=2$ 的洗禮。
第 3 層的神魔進行 3 次強度為 $gcd(3,10)=1$ 的洗禮，但該強度的洗禮無效。
第 4 層的神魔進行 4 次強度為 $gcd(4,10)=2$ 的洗禮。
第 5 層的神魔進行 5 次強度為 $gcd(5,10)=5$ 的洗禮。
第 6 層的神魔進行 6 次強度為 $gcd(6,10)=2$ 的洗禮。
第 7 層的神魔進行 7 次強度為 $gcd(7,10)=1$ 的洗禮，但該強度的洗禮無效。
第 8 層的神魔進行 8 次強度為 $gcd(8,10)=2$ 的洗禮。
第 9 層的神魔進行 9 次強度為 $gcd(9,10)=1$ 的洗禮，但該強度的洗禮無效。
因此，第 10 層的神魔的戰力變為 $2^{2+4+6+8}\cdot 5^5 = 3276800000 = 282066941 \pmod{998244353}$。

D. Angus 與菜寮

Problem ID: AngusFarm
Time Limit: 1.0s
Memory Limit: 128MiB

帥氣又富有的 Angus 在菜寮擁有一大片土地，而他的土地都拿來幹嘛呢？當然是種菜！

Figure 1: 菜寮

Angus 在菜寮種的菜可以平均劃分為 n 個區域種植不同的蔬菜，編號為 $1, 2, ..., n$，每個區域 i 有土地富饒程度 a_i，以及種出來的蔬菜量 b_i，最初所有 b_i 皆為 0。
接著每天會依序發生以下兩件事：
1. 首先是 Angus 可以選擇 k 塊不同的區域，花費 k 元為這些區域施肥，接著這些區域的土地富饒程度 a_i 都會增加 1。注意他無法在同一天中施更多肥來使富饒程度增加更多，因為如此一來肥料的濃度太高會有反效果甚至導致作物枯萎。
2. 再來每塊區域會長出與其土地富饒程度相等的蔬菜量，即對於所有 i 有 b_i 會增加 a_i。

現在 Angus 有 c 元的預算，並且他希望最終每塊地達到至少 v_i 的蔬菜量，他想知道最早在第幾天結束時他可以達到這個目標。
身為程式大師的 Angus 寫了一個程式來計算這個結果，但由於他有點久沒寫程式了所以可能連 Atcoder Beginner Contest 的前四題都做不出來，所以想請你也寫一個程式來確保他寫得沒錯。(當然了，因為他是程式大師，所以他寫的程式是正確的，如果你得到 WA 代表你的輸出結果與他的不一樣)

– 輸入 –

第一行包含兩個正整數 n, c，分別代表 Angus 的菜寮有幾個區域、他的預算。
第二行包含 n 個正整數 $a_1, a_2, ..., a_n$，代表每個區域初始的土地富饒程度。
第三行包含 n 個正整數 $v_1, v_2, ..., v_n$，代表每個區域需要達到的蔬菜量。

– 輸出 –

輸出最早在第幾天結束時每個區域的蔬菜量都可以超過預期。

– 輸入限制 –

- $1 \leq n \leq 8 \times 10^5$
- $0 \leq c \leq 10^{18}$
- $1 \leq a_i \leq 10^9$
- $1 \leq v_i \leq 10^9$

– 子任務 –

編號	分數	額外限制
1	0	範例輸入輸出
2	18	$c = 0$ 且 $n \leq 10^5$
3	24	$c \leq 10^6$ 且 $n \leq 10^5$
4	12	$c = 10^{18}$ 且所有 $a_i = 1$ 且 $n \leq 10^5$
5	16	$c = 10^{18}$ 且 $n \leq 10^5$
6	10	$n \leq 3000$
7	20	無額外限制

– 範例輸入 1 –

```
5 0
1 2 3 4 5
7 25 14 33 57
```

– 範例輸出 1 –

```
13
```

– 範例輸入 2 –

```
5 8
5 2 6 3 7
39 15 109 2 66
```

– 範例輸出 2 –

```
10
```

– 範例說明 –

範例測資 1 說明:
因為沒有預算施肥，因此各區域的土地富饒程度保持 $1, 2, 3, 4, 5$。
第 1 天結束時各區域的蔬菜量變為 $1, 2, 3, 4, 5$，第 2 天結束時各區域的蔬菜量變為 $2, 4, 6, 8, 10$，以此類推。最終在第 13 天結束時各區域的蔬菜量達到 $13, 26, 39, 52, 65$，皆達到各區域期望的蔬菜量。
可以證明沒有更早達到目標的辦法。

範例測資 2 說明:
第 1 天花費 2 元對區域 $2, 3$ 施肥，各區域的土地富饒程度變為 $5, 3, 7, 3, 7$，第 1 天結束時各區域的蔬菜量變為 $5, 3, 7, 3, 7$。
第 2 天花費 1 元對區域 3 施肥，各區域的土地富饒程度變為 $5, 3, 8, 3, 7$，第 1 天結束時各區域的蔬菜量變為 $10, 6, 15, 6, 14$。
而後在第 3 到第 7 天都花費 1 元對區域 3 施肥，第 7 天施肥後各區域的土地富饒程度變為 $5, 3, 13, 3, 7$，而第 7 天結束時各區域的蔬菜量變為 $40, 24, 70, 24, 56$。在這之後沒有預算施肥因此各區域的土地富饒程度保持 $5, 3, 13, 3, 7$，然後在第 10 天結束時各區域的蔬菜量達到 $50, 30, 109, 30, 70$，皆達到各區域期望的蔬菜量。
可以證明沒有更早達到目標的辦法。

E. 反毒

Problem ID: 114514
Time Limit: 3.0s
Memory Limit: 256MiB

書接上回，自從神魔之塔靠著大魔神 Ex 的幫助，打破了「六芒星的束縛」的禁制之後，神魔們的戰力恢復至了鼎盛時期，而憤怒的他們決定大舉進攻城之內的家鄉「玩具凡斗城」。然而，習慣了高高在上的神魔們已經懶得親身下場戰鬥。因此，他們合力創造了一種全新的生物，名為「毒龍」，並派遣了一支含有 n 隻毒龍的毒龍大軍前去攻打玩具凡斗城。

毒龍是一種非常邪門的生物，他的身軀極其的龐大且堅硬，任何擋在他面前的建築物都會被他撞為廢墟，因此若被毒龍大軍成功入侵玩具凡斗城，那將會是一場大災難。同時，這種生物又極難對付，因為他唯有在血量剛好降為 0 時才會完全被消滅。若是對毒龍造成的傷害使其血量降為負值，他便會轉為一種名為「反毒龍」的靈魂體，並繼續對玩具凡斗城造成傷害，而玩具反斗城的所有法師都沒有可以對付靈魂體的魔法，因此若有任何反毒龍產生，便諭示著玩具凡斗城的滅亡。在城之內發現這點後，心如死灰，心想敗局已定之時，平時只能躲在一眾坦克戰士後的伊登站了出來。伊登是一名職業法師，同時也是玩具凡斗城內最資深的法師，毒龍大軍交給她來處理再合適不過了。伊登的攻擊方式有以下三種：

1. 花費 a 的魔力，並對一個敵方目標造成 11 的傷害
2. 花費 b 的魔力，並對一個敵方目標造成 451 的傷害
3. 花費 c 的魔力，並對一個敵方目標造成 11451 的傷害

然而，伊登發現她的攻擊方式有可能無法消滅所有的毒龍，因此她不得不使出自己研發多年的黑魔法。之所以稱之為黑魔法，是因為該種魔法的功能為治療敵方目標，使得敵方目標血量上升，因此在平時的戰鬥中毫無用武之地，然而，現在拿來消滅毒龍大軍卻正好派得上用場。該種魔法會消耗 d 的魔力，讓伊登從 1 到 114514 的這些數字中，選一個數字 k 並對恢復一個敵方目標 k 的血量，而這項黑魔法又極其邪門，可以使目標的血量恢復至超過滿血。例如，若一個敵方目標當前的血量為 a ，並且其滿血時的血量為 b ，伊登使用黑魔法為其恢復 c 的血量時，該目標的血量會無條件變為 $a+c$ ，無論 $a+c$ 是否大於 b。

現已知毒龍大軍中共有 n 隻毒龍，且每隻的血量分別為 $h_1, h_2, ..., h_n$ ，試問在花費最少魔力的情況下，伊登需要多少魔力才能完全消滅毒龍大軍。若伊登無法透過他現有的魔法消滅毒龍大軍，則輸出 -1。

– 輸入 –

輸入共兩行。
第一行包含 5 個正整數，分別為 n, a, b, c, d，意義如題目所述。
第二行包含 n 個正整數，分別為 h_1, h_2, ..., h_n，意義如題目所述。

– 輸出 –

輸出只有一行。
請輸出一個正整數，代表伊登最少需要多少魔力才能消滅毒龍大軍。

– 輸入限制 –

- $2 \leq n \leq 10^5$
- $1 \leq a, b, c, d \leq 10^7$
- 對於所有 $1 \leq i \leq n$，都有 $1 \leq h_i \leq 10^9$

– 子任務 –

編號	分數	額外限制
1	0	範例輸入輸出
2	23	對於所有 $1 \leq i \leq n$ 都有 $1 \leq h_i \leq 10^6$
3	31	所有的 h_i 相等
4	46	無額外限制

– 範例輸入 1 –

```
2 1 2 3 2
462 11445
```

– 範例輸出 1 –

```
8
```

– 範例輸入 2 –

```
3 3 17 9 4
1 1 1
```

– 範例輸出 2 –

```
21
```

– 範例說明 –

範例測資 1 說明:
範例測資 1 中，第一隻毒龍可以透過各造成一次 11 和 451 的傷害量消滅，共花費 $1+2=3$ 的魔力。
第二隻毒龍可以在以黑魔法使其恢復 6 的血量後，進行一次 11451 的傷害以消滅，共花費 $3+2=5$ 的魔力。
因此，消滅這隻毒龍大軍最少需要 $3+5=8$ 的魔力。

範例測資 2 說明:
範例測資 2 中，第一隻毒龍可以透過使用黑魔法，使其恢復 10 的血量後，造成一次 11 的傷害來消滅，共花費 $4+3=7$ 的魔力。
第二隻和第三隻毒龍的血量也為 1，故消滅這隻毒龍大軍最少需 $7\times 3=21$ 的魔力。

F. 又一個簡短的問題

Problem ID: ShortProblemAgain
Time Limit: 2.0s
Memory Limit: 512MiB

輸入兩個正整數 n, m，請輸出第 m 小的非負整數 k 滿足 2^k 除以 $10^{12} + 39$ 的餘數為 n。若找不到滿足條件的第 m 小的非負整數 k 請輸出 -1。
(提示:C++ 有內建型別「__int128_t」可以儲存 128-bit 整數，惟無法直接進行輸出，但在本題若出現 long long int 相乘導致的溢位，可以使用「(__int128_t)x*y%(long long int)(1e12+39)」(其中 x,y 是 long long int)，來將運算結果暫時轉為 __int128_t 避免溢位，再取模後即可存入 long long int 變數內。另外 $10^{12} + 39$ 是質數。)

– 輸入 –

輸入只有一行，包含兩個正整數 n, m，意義如題目所述。

– 輸出 –

輸出第 m 小的非負整數 k 滿足 2^k 除以 $10^{12} + 39$ 的餘數為 n。若找不到滿足條件的第 m 小的非負整數 k 請輸出 -1。

– 輸入限制 –

- $1 \leq n < 10^{12} + 39$
- $1 \leq m \leq 10^{10000000}$

– 子任務 –

編號	分數	額外限制
1	0	範例輸入輸出
2	15	保證存在 $k \leq 10^7$ 滿足 2^k 除以 $10^{12} + 39$ 的餘數為 n 且 $m = 1$
3	20	保證存在 $k \leq 10^7$ 滿足 2^k 除以 $10^{12} + 39$ 的餘數為 n 且 $m \leq 10^6$
4	25	$m = 1$
5	30	$m \leq 10^6$
6	10	無額外限制

– 範例輸入 1 –

```
1024 1
```

– 範例輸出 1 –

```
10
```

– 範例輸入 2 –

```
99511627737 1
```

– 範例輸出 2 –

```
40
```

– 範例說明 –

範例測資 1 說明:
$2^{10} = 1024$，可以證明找不到比 10 更小的非負整數 k 使得 2^k 除以 $10^{12} + 39$ 的餘數為 1024。

範例測資 2 說明:
$2^{40} = 1099511627776$，除以 $10^{12} + 39$ 的餘數為 99511627737，可以證明找不到比 40 更小的非負整數 k 使得 2^k 除以 $10^{12} + 39$ 的餘數為 99511627737。