第四題：提款機問題 (ATM) [此題為互動題 Interactive]

問題敘述

踢歐埃共和國長久以來只有一種貨幣面額：M 元。所有的交易額度都以 M 的整數倍處理，相當地不方便。於是踢歐埃共和國的中央銀行決定要進行一些全國性的實驗，除了面值為 M 的貨幣以外，嘗試發行各種面額較小的貨幣。在實驗的過程中，央行可能會隨時發行面額為 x 的貨幣、或是回收所有面額為 x 的貨幣。

貨幣的設計本身是相當特別的，把它拿在手上其實不具有任何意義。只有在金流移動的時候（比方說付款、或是提款等等）該貨幣才會在兩方的戶頭產生轉帳的功效。而事實上，面額為 x 的一張貨幣，根據給出這張貨幣的人的選擇，它可以發揮出 $+x$ 或 $-x$ 其中之一的效果。

踢歐埃共和國有一款非常神奇的提款機。這些提款機跟自動販賣機長得很像，只不過，它的每一個品項對應到的是一綑鈔票。更仔細地說，提款機上有一些兩個一組的按鈕，每一組按鈕對應到了兩個參數 (v, q)，當你選擇了這組按鈕，並按下其中一個按鈕時，提款機會吐出恰好 q 張面額為 v 元並且發揮了 $+v$ 功效的貨幣。相反地，如果你按下了同組的另一個按鈕，提款機會吐出恰好 q 張面額為 v 元並且發揮了 $-v$ 功效的貨幣。此外，這個提款機有個安全限制：對於任何一次交易，每一組按鈕只能擇一按鈕按下至多一次。同一個按鈕按兩次以上、或是同一組的兩個按鈕都按，都會引起保全人員的注意。隨便亂按參數是 $(0, 0)$ 的按鈕或按到不存在的面額也是不被允許的。

目前踢歐埃共和國裡面的每一台提款機，都支援同時存在 65536 組不同的按鈕參數，每一組按鈕參數編號為 $1, 2, \ldots, 65536$。目前只有前 60 組有在使用：對於 $1 \le i \le 60$，第 i 組的按鈕參數是 $(v_i = M, q_i = 2^{i-1})$。對於其他的 $i > 60$，皆有 $(v_i = 0, q_i = 0)$。

身為維護該神奇提款機的工程師，央行要求你動態修改提款機提供的參數，並且協助進行一些實驗：央行可能會隨時發行面額為 x 的貨幣、回收面額為 x 的貨幣、或是詢問你某些金額要怎麼湊才能**用最少張數**湊出來。當一種新的貨幣開始發行、或一種現有貨幣被回收的時候，你可以更動提款機的按鈕參數至多 60 次。**目標是讓使用到的最大按鈕參數編號越小越好。** 當然，更動後必須確保按下這個按鈕的時候，會有鈔票跑出來，也就是說回收一種貨幣時，需要把該面額貨幣使用到的參數歸零。

實作細節

你需要完成以下四個函式：

```
void initialize(int M);
void make_available(int v);
void make_unavailable(int v);
vector<set<int>> calculate_optimal_bundle_set(vector<long long> values);
```

- 評分程式一開始會呼叫 `initialize` 並傳入 M 值。
- 傳入的 M 值滿足 $1 \le M \le 500$。
- `make_available` 函式被呼叫的時候，代表央行決定增加一種貨幣面額 v。輸入保證 $1 \le v < M$ 而且是目前沒有發行的面額。
- `make_unavailable` 函式被呼叫的時候，代表央行決定回收並取消面額為 v 的貨幣。輸入保證 $1 \le v < M$ 而且是目前已發行的面額。
- `calculate_optimal_bundle_set` 函式被呼叫的時候，代表央行想要知道你調整後的提款機，要如何湊出指定數值。
- 對於每一個 `values`[i]，`calculate_optimal_bundle_set` 回傳的第 i 個集合 S 包含一些整數。若 $j \in S$ 則代表按鈕 j 被按下，且提款機吐出恰好 q_j 張面額為 v_j、發揮 $+v_j$ 功效的貨幣。若 $-j \in S$ 則代表按鈕 j 被按下，且提款機吐出恰好為 q_j 張面額為 v_j、發揮 $-v_j$ 功效的貨幣。這些紙鈔的總張數必須是所有能夠湊出 `values`[i] 元最少的湊法。如果有多種最小張數湊法，回傳其中的任何一種都可以。
- 對於每一個 `values`[i]，你可以假設必定存在一種使用當前央行發行的貨幣，湊出恰好 `values`[i] 元的方法。
- 輸入保證 $0 \le$ `values`[i] $\le 10^{18}$。
- 對於每一筆測試資料，央行會呼叫 `make_(un)available` 函式合計至多 10^5 次。
- 對於每一筆測試資料，央行至多會呼叫 30 次 `calculate_optimal_bundle_set` 函式，且所有 `values` 的長度加起來不超過 10^5。

你的程式可以呼叫以下函式：

```
void set_bundle(int id, int v, long long q);
```

- `set_bundle` 函式會設定提款機編號為 id 的按鈕參數組，將這組按鈕參數改為 (v, q)。
- 傳入之參數 id 必須滿足 $1 \le id \le 65536$。
- 傳入之參數 v 必須是目前央行發行中的貨幣面額，或是 0。
- 傳入之參數 q 必須滿足 $0 \le q < 2^{60}$，且 $v = 0$ 時 q 必須也等於 0、$v > 0$ 時 q 必須也大於 0。
- 在 `calculate_optimal_bundle_set` 函式執行的時候，不得呼叫 `set_bundle` 函式；在每次 `make_(un)available` 函式執行時，允許呼叫 `set_bundle` 函式至多 60 次。

如果不符合上述條件限制，你的程式會被判為 **Wrong Answer**；否則你的程式會被判斷為 **Accepted**。

互動範例

考慮以下的測試資料：$M = 10$。

一個被評分程式判斷為 **Accepted** 的互動例子顯示如下：

評分程式端	參賽者端
呼叫 `initialize`(10)。	
呼叫 `make_available`(3)。	
	呼叫 `set_bundle`$(61, 3, 1)$。
	呼叫 `set_bundle`$(62, 3, 2)$。
呼叫 `calculate_optimal_bundle_set`$([6, 7, 11])$。	
	回傳 $[\{62\}, \{-61, 1\}, \{-62, -61, 2\}]$。

評分說明

對於一筆測試資料，假設你呼叫 `set_bundle` 的過程中使用的最大參數 id 值為 x，則得到的**分數比重** S 值如下：

- 若 $x > 60M$，則 $S = 0.0$。
- 若 $x \leq 60 + \lceil \frac{4}{3} M \rceil - \lfloor \log_2 M \rfloor$，則 $S = 1.0$。
- 若為其他情形，得分比重為：

$$S = 1 - 0.18 \left(-1 + \sqrt{\log_{\frac{4}{3}} \left(\frac{x - 60 + \lfloor \log_2 M \rfloor}{M} \right)} \right)$$

本題共有 5 組測試題組，條件限制如下所示。每一組可有一或多筆測試資料，你的得分是該組所有測試資料之得分比重 S 中最低者，乘以該子任務的分數。

子任務	分數	額外輸入限制
1	29	$M \leq 250$，且所有 `values` 內數值都不超過 255。
2	23	$M \leq 250$，且評分程式只會呼叫一次 `calculate_optimal_bundle_set`。
3	14	$M \leq 10$。
4	31	$M \leq 250$。
5	3	無額外限制。

範例評分程式

範例評分程式以下列格式讀取輸入：

- 第 1 列：M, Q
- 第 $2 \sim Q+1$ 列：每一列一開始有一個 op，根據 op 決定評分程式的動作。
  - $op = 1$：後面緊接著 x，此時評分程式呼叫 `make_available`(x)
  - $op = 2$：後面緊接著 x，此時評分程式呼叫 `make_unavailable`(x)
  - $op = 3$：後面接著 $k, values_0, values_1, \ldots, values_{k-1}$ 代表一組詢問。

請注意：使用自己上傳的測試資料進行測試時，沒有下列 MSG 描述的情形時你總會得到 Accepted。 如果你的程式被評為 **Accepted**，範例評分程式輸出 Accepted: $MaxId$，其中 $MaxId$ 表示你使用到的最大按鈕參數編號。如果你的程式被評為 **Wrong Answer**，範例評分程式輸出 Wrong Answer: MSG，其中 MSG 格式與意義如下：

- no quota to edit: 呼叫 `set_bundle` 次數超過限制。
- button id out of range: 呼叫 `set_bundle` 的時候按鈕參數 id 超過範圍。
- invalid v: 呼叫 `set_bundle` 的時候按鈕參數 v 不滿足題目條件。
- invalid q: 呼叫 `set_bundle` 的時候按鈕參數 q 不滿足題目條件。
- (v, q) must be both zero or both non-zero: 呼叫 `set_bundle` 的時候按鈕參數不滿足題目要求。
- face value is unavailable: 呼叫 `set_bundle` 的時候按鈕參數 v 不是央行正在發行的面額。
- invalid length of returned vector: 呼叫 `calculate_optimal_bundle_set` 回傳的陣列長度不符合規定。
- button id out of range: 呼叫 `calculate_optimal_bundle_set` 回傳的某個集合使用了不合法的按鈕編號。
- id and -id cannot coexist: 呼叫 `calculate_optimal_bundle_set` 回傳的某個集合同時按下了屬於同一組的兩個按鈕。
- answer contains invalid button: 呼叫 `calculate_optimal_bundle_set` 回傳的某個集合內包含不能按的按鈕編號。
- returned set does not result in correct value: 呼叫 `calculate_optimal_bundle_set` 回傳的某個集合湊不出指定金額。
- exists unavailable face value on ATM: 呼叫 `make_unavailable` 結束以後，有一些該面額的按鈕參數沒有清除。