第四題：綴綴不安 (reconstruction)

問題敘述

踢歐埃王國是一個充滿了神奇魔法的國度。在這個王國內，最常見的魔法形式是以拼出咒語的方式實行的。儘管不同的咒語有強弱之分，只要是由英文字母組成的任意字串，都可以是個咒語。舉例來說，`turnonthelight` 是一個能夠把房間內的燈點亮的簡單咒語；而 `turnonthelightforabitur` 則是為了德國高中學測 (Abitur) 把房間周圍的燈光點到最亮的強力咒語。

身為魔法研究協會重要成員的你，對於咒語結構與其強度的關聯性自然有著濃厚的興趣。在大魔法使弗蘭梅千年前留下來的魔法書[*2]中，記載著關於咒語強度的研究。它的敘述是這樣的：『前綴與後綴**等價**時，愈長者愈強也。全小寫，更強。』

諳熟古文的你自然是懂的：考慮一個長度為 n 的咒語字串 $S[0..n-1]$[*3]。對於整數 $i < n$，字串 S 的第 i 個**前綴 (prefix)** $prefix(S,i)$ 被定義為，將 S 的前 i 個字母依照順序接起來的字串，也就是 $prefix(S,i) = S[0..i-1]$。類似地，字串 S 的第 i 個**後綴 (suffix)** $suffix(S,i)$ 則是被定義為字串 S 的最末 i 個字元形成的字串 $suffix(S,i) = S[n-i..n-1]$。

至於**等價**是什麼意思呢？ 我們說兩個長度相同的字串 A 與 B 等價，若且唯若存在一種將字母一一對應的方式將 A 轉換成 B，記作 $A \simeq B$。等價的例子包括 `blueleg` $\simeq$ `frieren` 以及 `pikachu` $\simeq$ `fireman`；但是 `frieren` $\not\simeq$ `fireman`。

魔法書中的句子便可以翻譯成新世代的語言：給定咒語字串 S，對於任何滿足 $prefix(S,i) \simeq suffix(S,i)$ 的整數 i，當 i 的值越大的時候，整個咒語的威力就越強。這個世代的人們把滿足條件的 i 稱為 S 的**等價前後綴長度**。

時代不同了。隨著人類對於魔法研究的理解越來越深刻，對於咒語結構和強度的關聯性研究也越來越細膩。人類甚至發展出了 1,000,000 種不同的字元，並且把它們融入咒語中！ 你埋首研究一陣以後，意識到：若不只考慮一個咒語的最長等價前後綴長度，還考慮該咒語的所有等價前後綴長度形成的集合 C，說不定能對咒語強度的理解更上一層樓！

為了魔法研究的進行，你希望在考慮各式各樣的集合 C 時，實際構造出一對應的咒語，以便拿來試驗你提出的各種假說。寫個程式來產生**任意**滿足前述條件的咒語吧！ 當然，對於某些集合 C，可能不存在對應的咒語，這時候你的程式也必須發現這點並且回報給你。

[*2] 坊間流傳的大魔法使魔法書有相當多的贋品，閱讀前請務必仔細確認記載之咒語合理性。

[*3] 對於整數 a, b 與字串 S，我們用 $S[a..b]$ 代表由以下字母 $S[a], S[a+1], \ldots, S[b]$ 串接起來的字串。

實作細節

你需要完成以下函式：

```
std::vector<int> construct(int n, const std::vector<int>& C);
```

- 對於每一筆測試資料，正式評分程式會呼叫你實作的 `construct()` 函式 $T \leq 5$ 次。
- n 代表希望構造出的咒語長度。
- C 是一個正整數序列，它描述了所求咒語的所有等價前後綴長度，從小到大排列。對於所有 $i \in C$ 皆有 $1 \leq i \leq n-1$。
- `construct()` 需回傳滿足題目條件的咒語 S、空陣列 $\{\}$、或者是 $\{-1\}$。由於允許的字元種類數為 $1\,000\,000$ 種，為了方便起見，咒語 S 將以一個整數陣列 `std::vector<int>` 的形式回傳。第 i 個整數 $S[i]$ 代表咒語字串的第 i 個字元。
- 若你回傳了一個滿足條件的咒語，那麼回傳的陣列長度必須恰好為 n，而且對於所有 i 皆有 $0 \leq S[i] \leq 999\,999$。
- 若不存在滿足條件的咒語，請回傳一個空的陣列 $\{\}$。
- 倘若你非常確定存在至少一個滿足條件的咒語，可是你卻找不到它，請回傳 $\{-1\}$，在某些情形下（詳見評分說明一節）你將能獲得部分分數。

測資限制

- $1 \leq T \leq 5$。
- $2 \leq n \leq 200\,000$。

範例評分程式

範例評分程式採用以下格式輸入：

$$
\begin{array}{l}
T \\
n_1 \ \ k_1 \\
i_{1,1} \ \ i_{1,2} \ \ \cdots \ \ i_{1,k_1} \\
\vdots \\
n_T \ \ k_T \\
i_{T,1} \ \ i_{T,2} \ \ \cdots \ \ i_{T,k_T}
\end{array}
$$

第一列的 T 代表呼叫 `construct()` 的次數。對於第 t 次的呼叫 $(1 \le t \le T)$，範例評分程式會傳入 $n = n_t$ 以及 $C = C_t$。其中集合 C_t 的大小為 k_t，且 $C_t = \{i_{t,1}, i_{t,2}, \ldots, i_{t,k_t}\}$。在本地測試時，範例評分程式會將 `construct()` 回傳的陣列內容以下列格式輸出：

$$
|S| \ \ S[0] \ \ \ldots \ \ S[|S|-1]
$$

範例測試

Sample Input	Sample Output
4 5 1 1 5 2 1 3 6 3 1 2 4 4 2 1 3	5 0 1 0 0 0 5 2 3 3 4 4 6 5 5 6 6 5 5 0

評分說明

對於每一筆測試資料，若你的函式正確地回傳 {}、或是正確地回傳咒語 S，那麼你將會得到**分數比重** $P = 1.0$；若你的函式回傳了不滿足條件的陣列，那麼你將會得到分數比重 $P = 0.0$；當存在滿足條件的咒語時，若你的函式回傳了 $\{-1\}$，那麼 $P = 0.4$。

本題共有 6 組子任務，條件限制如下所示。每一組可有一或多筆測試資料，你在該子任務的得分為所有組內測試資料的 P 值的最小值，乘以該子任務的總分。在下面的敘述中，我們令 $k = |C|$，且將集合 C 內的數由小到大排列為 $C_0 < C_1 < \cdots < C_{k-1}$。

子任務	分數	額外輸入限制
1	10	$k = 2$。
2	12	$2C_{k-1} \le n$；此外對於所有 i 皆有 $2C_{i-1} \le C_i$。
3	3	$C \cup \{n\}$ 形成一個等差數列。
4	29	$C_0 = 1$、$C_1 = 2$、$2C_{k-1} > n$、對所有 $i \ge 2$ 皆有 $2C_{i-1} > C_i$、且 $n \le 5\,000$。
5	22	$n \le 5\,000$。
6	24	無額外限制。