問題 15 – Brainf*ck

(25 分)

問題敘述

Brainf*ck 是一個極簡程式語言，他用一個初始值為 0 的陣列來模擬記憶體，陣列每個元素都是一個 0~255 的整數，如同常見的程式語言，255 加上 1 會變成 0，而 0 減掉 1 會變成 255，另外還有一個指標一開始指著陣列的第一個元素。

Brainf*ck 只有八種字元，程式執行的時候，會先執行程式碼第一個字元代表的指令，接著執行第二個字元代表的指令，一直到整個程式碼被執行完，但若遇到代表迴圈的 [] 可能會重複執行指令。各個字元代表的意義如下：

<	將指標往左移一格
>	將指標往右移一格
+	將指標指向的元素加上 1
-	將指標指向的元素減掉 1
[	若當前指標指向的元素為 0，則跳到對應的右括號繼續執行程式
]	跳至對應的左括號繼續執行程式

實際上還有代表輸入輸出的 “.” 和 “,”，但本題並不會使用。若把 brainf*ck 的六個字元轉換成 C 語言可以對應到下面的表格（取自維基百科）

<	`++ ptr;`
>	`--ptr;`
+	`++(*ptr);`
-	`--(*ptr);`
[	`while(*ptr){`
]	`}`

其中，ptr 是一個指向陣列某個元素的指標。例如 [+] 會把當成指標指向的元素不斷加一，直到 overflow 成 0，相當於 C 語言中的

```
while(*ptr){
      ++(*ptr);
}
```

而 `[->+<]` 則相當於

```
while(*ptr){
      --(*ptr);
      ++(*(ptr + 1));
}
```

注意到最後的 `<` 是為了確保每次迴圈開始的時候指標是在 `ptr` 而不是 `ptr + 1`。

本題規定記憶體大小只有 1024，也就是 `brainf*ck` 會在大小 1024的陣列上執行，若指標超出陣列範圍則會產生 `error`。給定正整數 n 和長度為 n 的陣列 $a_1, a_2, \ldots, a_n$，請設計一個長度不超過 18000 的 `brainf*ck` 程式使得程式結束時記憶體的前 n 個數字恰為 $a_1, a_2, \ldots, a_n$ 且記憶體其他位置皆為 0。另外為了避免程式執行過久，這個程式最多只能執行 10^7 個指令（字元）。

輸入格式

輸入共兩行。第一行有一個正整數 n，第二行有 n 個以空白分隔的整數 $a_1, a_2, \ldots, a_n$。

輸出格式

輸出一個字串代表一個 `brainf*ck` 程式碼，這個字串只能有 `+-><[]` 這六種字元，若出現非法字元 （ 包含最後的換行字元 ） 則會判定為 `Wrong Answer`。合法字元的數量最多只能有 18000 個。所有產生正確記憶體數值的程式碼皆可得到 `Accepted`。

資料範圍

$1 \leq n \leq 1000$
$0 \leq a_i \leq 255$

輸入範例 1

```
2
0 16
```

輸出範例 1

```
++++[->++++<]
```

輸入範例 2

```
1
255
```

輸出範例 2

```
-
```

範例說明

範例 1 中的 brainf*ck 程式碼 ++++[->++++<] 相當於以下 c 語言程式碼，所以 brainf*ck 程式執行結果，陣列第一格會是 0, 第二格會是 16

```
// initialize
int memory[1024] = {0};
int ptr = 0;

memory[ptr] += 4;          //++++
while(memory[ptr]){              //[
     --memory[ptr];        //-
     ++ptr;                 //>
     memort[ptr] += 4;     //++++
     --ptr;                //<
}                    //]
```

範例 2 則是把陣列第一格減一，因為初始值為 0，所以結果會是 255。