1 RMQ 問題

取極值在數學中跟演算法中都是很重要的操作，現在考慮這樣的一個問題：給定一個一維的 n 個元素的陣列 A，接著有多筆詢問，每組詢問都為一數對 (x, y)，其中 $x \leq y$，代表詢問 $A[x], A[x+1], \ldots, A[y]$ 中最大的元素為何。這樣子的問題稱為「區間極值查詢」(Range Minimum/Maximum Query, RMQ) 問題。

人們對於 RMQ 問題已經有許多的研究，底下簡單介紹幾種常見的 RMQ 問題的形式：

1. 無修改/離線 (off-line) 查詢

   這是最簡單的形式，給定陣列 A 後，使用者不但不會去更動 A 的內容，還可以預先知道所有的詢問 (或者可以想成可以一口氣讀入所有的詢問)。其中後面的這個性質我們通常稱為「離線查詢 (off-line query)」，只能解決離線查詢的演算法通常稱為「離線算法 (off-line algorithm)」。

2. 無修改/在線 (on-line) 查詢

   相較於離線查詢，支援動態詢問的「在線查詢 (on-line query)」是更符合現實需求的問題類型。可以解決在線查詢的演算法通常稱為「在線算法 (on-line algorithm)」。注意到一個在線算法一定也能解決離線查詢的問題，所以我們不會用離線算法來稱呼一個在線算法。

3. 可修改/離線查詢

   相較於無修改的類型，可修改的類型要求使用者必須可以修改 A 的內容。許多無修改的好演算法都仰賴於先對 A 進行一些預先的處理好回答之後的問題，因此可修改的問題往往難度上升很多。當我們在討論可修改的離線查詢問題時，通常意指修改的指令和查詢的指令都能夠預先知道。當然，修改的形式也有很多種，本文中不詳細討論。

4. 可修改/在線查詢

   四種類型中毋庸置疑的最困難的形式。事實上，很多實際應用上的問題，人類至今還沒有太好的支援修改的在線算法。

目前針對 RMQ 問題，若修改的方式不要太奇怪，人類多數已經發現很不錯的方法，其中無修改類型的問題甚至已經有 $O(n+q)$ (q 是詢問的次數) 的理論最佳算法。我們會在往後的課程中提到一些 RMQ 問題的經典解法，而在這次作業中，我們先介紹一種對於可以「單點修改，在線查詢」的 RMQ 問題，用「分組」的方式優化時間複雜度的方法。

首先，最大值 (最小值) 有一個特性，就是

$$\max(a, b, c) = \max(a, \max(b, c))$$

意即，要對很多數字取極值，可以先對其中一些數字取極值，再把結果和其他數字取極值。將這個性質套用到查詢的操作上，可以得到一個特性：假設一次查詢的區間為 $[x, y]$，如果我們可以找到一些小區間 $[x_1, y_1], [x_2, y_2] \cdots [x_k, y_k]$，並且這些小區間的聯集恰好是 $[x, y]$，那麼先對於每個小區間分別求極值，之後再取所有小區間極值之中的極值即是 $[x, y]$ 的極值。如果對於不同的詢問區間，都可以分割成盡量相同的小區間的話，我們就可以在一開始就先預先算好這些小區間的極值，之後在詢問的時候就可以直接使用這些結果，節省時間。

基於這個想法，我們把 A 陣列的元素按照編號分組，從 0 號開始每連續的 k 個分成一組 (如：$n = 8, k = 3$ 就分成兩組，第一組為 $A[0 \cdots 2]$，第二組為 $A[3 \cdots 5]$，第三組為 $A[6 \cdots 7]$)，並且在開始回答詢問前先預處理好每一組內的極值。對於一個詢問區間 $[x, y]$，它必定跨越若干個完整的組，而前後可能會包含某個組的一部分，如下圖中，詢問的區間包含兩個完整的組 (綠色)，區間兩端則分別是兩個組的一部分 (紅色)：

對於中間橫跨的完整的組的部份，我們直接取一開始預處理後的結果得到該組的極值；對於兩端點覆蓋不完整組的部份，則是暴力的一個一個比較，如此一來在中間完整成組的部份就可以省去許多時間。

單點修改一個值時，除了修改該點之外，該點所屬的組的極值也可能改變，因此還需要重新計算該組的極值。更新的方法就是將組別內的所有元素一一看過，並找出最大值/最小值。

該做法的時間複雜度將在習題中探討。

習題

1. 我們使用「分組」算法處理一個 RMQ 問題，支援

   - 將陣列其中一個元素修改為 val
   - 詢問陣列中一段區間的最大值

   已知 n 為陣列大小，k 為每組大小，請回答下列問題：

   (a) (10 pts) 請問預處理的時間複雜度為何？

   (b) (15 pts) 請問單點修改的時間複雜度為何？

   (c) (15 pts) 請問回答詢問的時間複雜度為何？

   (d) (10 pts) 根據 (a),(b),(c)，若詢問和修改一樣重要，則 k 應該取多少可以讓算法有最低的時間複雜度呢？為什麼？此時時間複雜度又是多少？

2. 我們使用「分組」算法處理一個比較複雜的 RMQ 問題，支援

   - 將陣列中一段區間中的值都修改為 val
   - 詢問陣列中一段區間的最大值

   已知 n 為陣列大小，k 為每組大小，請回答下列問題：

   (a) (10 pts) 假設某一組的值全部都被修改為 val，請問這一組的最大值為多少？

   (b) (20 pts) 請設計修改和查詢的算法，使得這兩種操作的複雜度皆為 $O(k + \frac{n}{k})$。

3. 我們使用「分組」算法處理一個比較複雜的 RMQ 問題，支援

   - 將陣列中一段區間中小於 val 的值都修改為 val
   - 詢問陣列中一段區間的最大值

   已知 n 為陣列大小，k 為每組大小，請回答下列問題：

   (a) (20 pts) 請設計修改和查詢的算法，使得這兩種操作的複雜度皆為 $O(k + \frac{n}{k})$。