建議對演算法與數據結構已經有足夠熟悉的學員跳過此章節

主要原因是我們課程的設計核心為實際去模擬當你真實遇到該問題時會有什麼樣的想法，但講義內容其實已經有部分暗示解法，但實際遇到問題時並不會知道接下來的題目會用到解法的方向。

# Sliding window

Sliding window 靠兩個 pointers 去維護一段資訊，通常是靠移動 left pointer 跟 right pointer 去擴展或縮減去更新資訊，並找出所有符合條件的元素或元素集合，並藉由這些元素集合去計算出答案，這邊的元素集合大部分情況下都是指 subarray。判斷一個題目是否能使用 Sliding window 實作，首先要思考如果原本的 subarray 不符合條件（這邊條件指符合題目的答案），那包含這個 subarray 的其他 subarray 也不會符合條件，反之如果原本的subarray符合條件，那所有被這個 subarray 包含的 subarray 也都會符合條件。

## 分辨是否使用

以下兩個題目：

1. 給定一串數字陣列，找出最長沒重複數字的子陣列有多長？
2. 給定一串由 01 組成的陣列，找出一段最長 0 和 1 數量相等的子陣列有多長？

第一題能夠用 sliding window 實作因為如果某段陣列 `1,3,6,2,5` 符合條件，那他的子陣列也都不會有重複數字（符合條件）。反之如果某段陣列 `1,3,3,5,2` 不符合條件，那包含他的其他陣列也都會有重複數字（不符合條件）。

第二題不能使用 sliding window 實作因為某段陣列 `0,1,0,1` 符合條件，但他的子陣列 `0,1,0` 的 0 跟 1 數量並不相等（不符合條件）。反之某段陣列 `0,1,0` 不符合條件，但包含他的陣列 `0,1,0,1` 的 0 跟 1 數量卻相等（符合條件）。因此並不符合使用 sliding window 的前提。

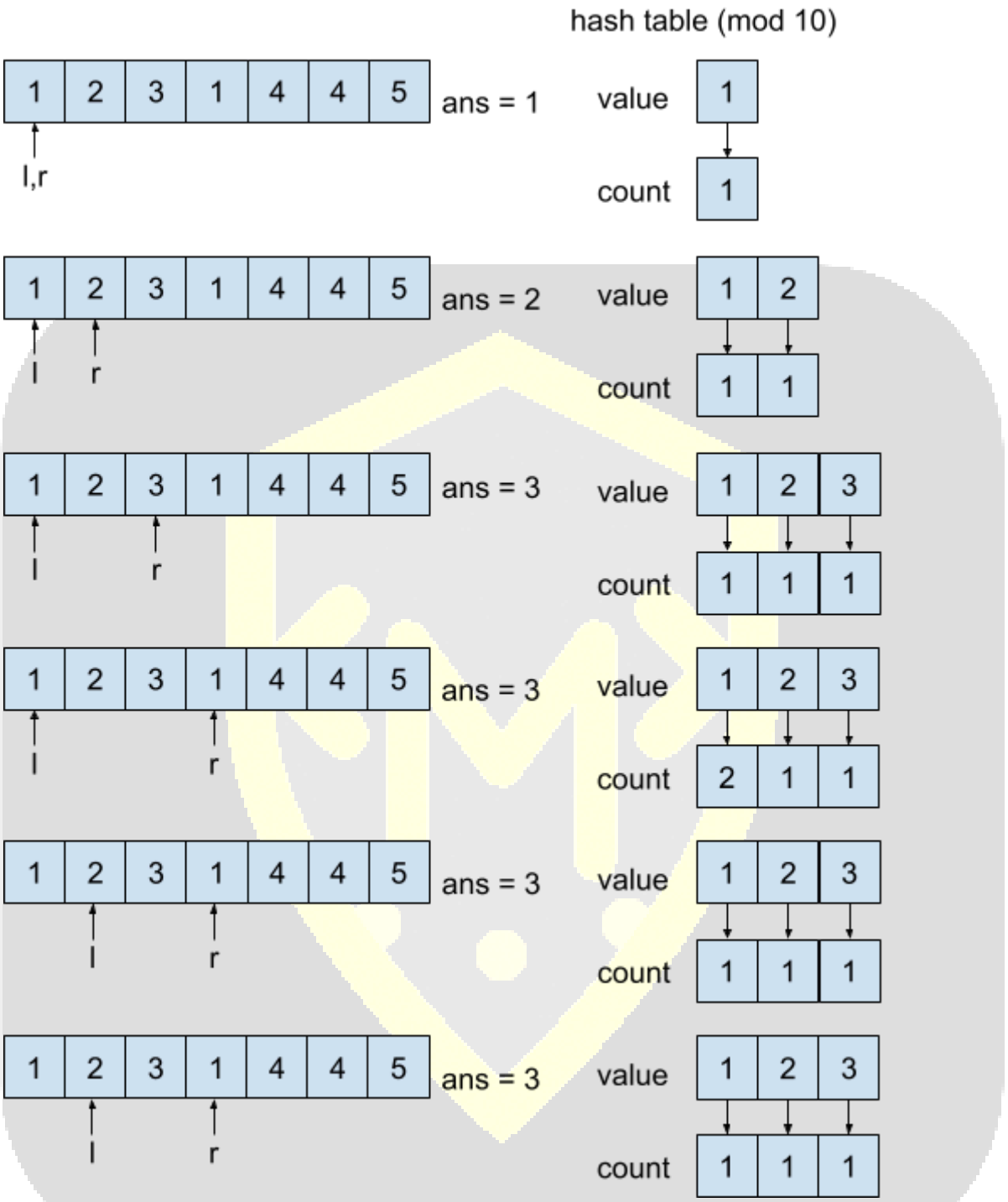
## Sliding window 主要分成四部分

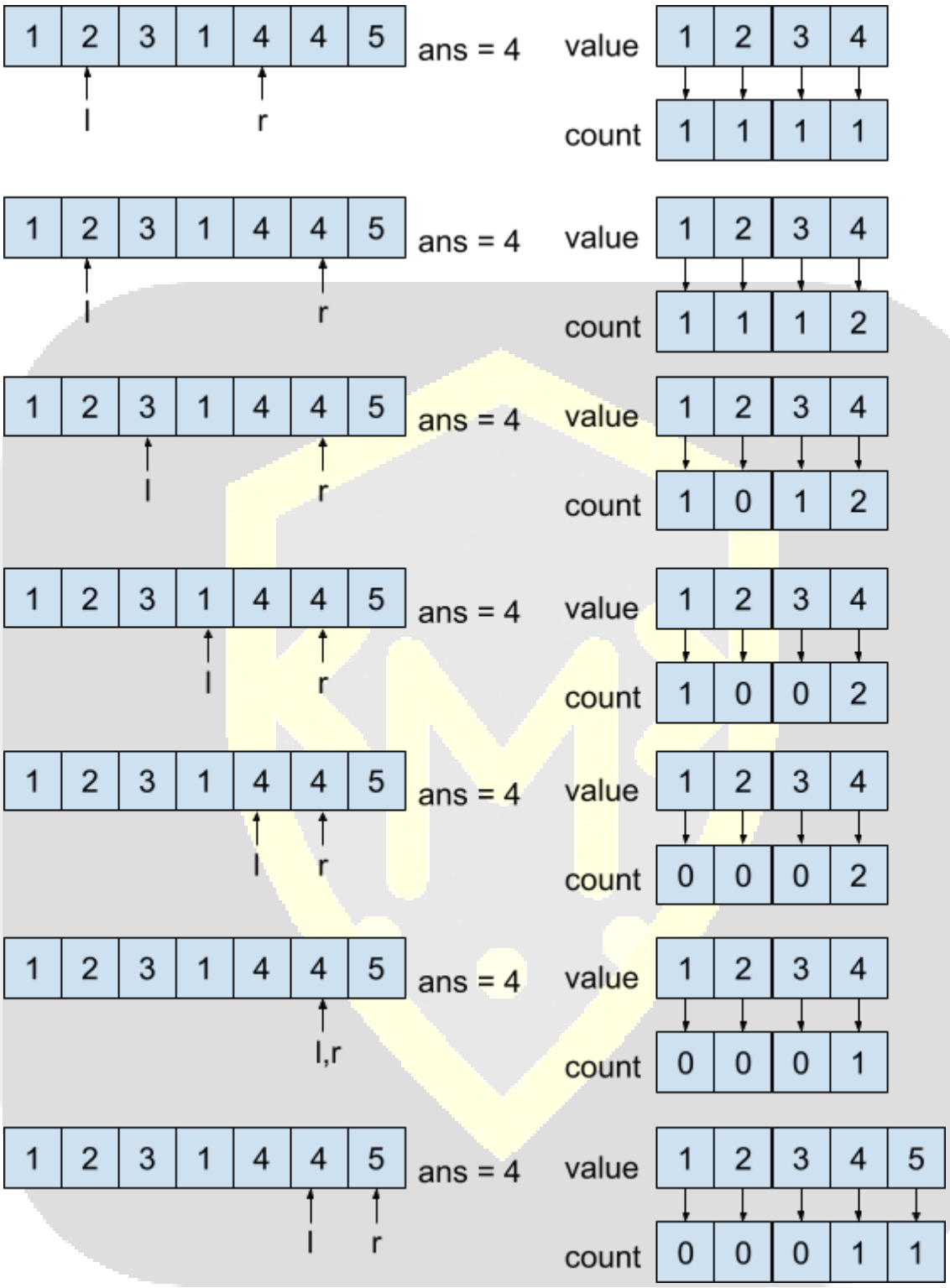
1. 決定要儲存的資訊，且該資訊可以幫助我們判斷當前元素集合是否符合條件
2. 當加入一個元素時如何更新資訊
3. 當刪除一個元素時如何更新資訊
4. 更新答案

## 範例

以下範例會使用上述的四個部分來執行。

通常在做 sliding window 的時候會使用 hash table 來儲存資訊（hash table 的用法請參考對應章節）。以下用『給定一串數字陣列，找出最長沒重複數字的子陣列有多長？』這個題目舉例。l 和 r 分別代表子陣列中最左邊的 index 和最右邊的 index，我們要找到對於每個 r 他對應的 l 最遠可以到哪裡。以下範例陣列為 `1,2,3,1,4,4,5`，其答案為 4。






○