

(b)

有顏色者為目前處理區間，藍色為 *pivot*， $pivotIndex = (n - 1)/2$ ，橘色為已分完堆之數字

[1 8 5 3 2 6 4 7] → [1 8 5 7 2 6 4 3] → [1 2 5 7 8 6 4 3] → [1 2 3 7 8 6 4 5] →
[1 2 3 7 8 6 4 5] → [2 1 3 7 8 6 4 5] → [2 1 3 7 8 6 4 5] → [1 2 3 7 8 6 4 5] →
[1 2 3 7 8 6 4 5] → [1 2 3 7 8 4 5 6] → [1 2 3 4 5 7 8 6] → [1 2 3 4 5 6 8 7] →
[1 2 3 4 5 6 8 7] → [1 2 3 5 4 6 8 7] → [1 2 3 5 4 6 8 7] → [1 2 3 4 5 6 8 7] →
[1 2 3 4 5 6 8 7] → [1 2 3 4 5 6 7 8] → [1 2 3 4 5 6 7 8] → [1 2 3 4 5 6 7 8]

(c)

藍色為目前處理區間，由左至右依序放入 *bucket*，橘色為從 *bucket* 取出後之狀態

[15 27 35 17 36 28 16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [], 6: [26], 7: [], 8: [], 9: []
[27 35 17 36 28 16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15], 6: [26], 7: [], 8: [], 9: []
[35 17 36 28 16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15], 6: [26], 7: [27], 8: [], 9: []
[17 36 28 16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15 35], 6: [26], 7: [27], 8: [], 9: []
[36 28 16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15 35], 6: [26], 7: [27 17], 8: [], 9: []
[28 16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15 35], 6: [26 36], 7: [27 17], 8: [], 9: []
[16] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15 35], 6: [26 36], 7: [27 17], 8: [28], 9: []
[] →
0: [], 1: [], 2: [], 3: [], 4: [], 5: [15 35], 6: [26 36 16], 7: [27 17], 8: [28], 9: []

[15 35 26 36 16 27 17 28]

[35 26 36 16 27 17 28] →
0: [], 1: [15], 2: [], 3: [], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[26 36 16 27 17 28] →
0: [], 1: [15], 2: [], 3: [35], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[36 16 27 17 28] →
0: [], 1: [15], 2: [26], 3: [35], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[16 27 17 28] →
0: [], 1: [15], 2: [26], 3: [35 36], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[27 17 28] →
0: [], 1: [15 16], 2: [26], 3: [35 36], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[17 28] →
0: [], 1: [15 16], 2: [26 27], 3: [35 36], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[28] →
0: [], 1: [15 16 17], 2: [26 27], 3: [35 36], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []
[] →
0: [], 1: [15 16 17], 2: [26 27 28], 3: [35 36], 4: [], 5: [], 6: [], 7: [], 8: [], 9: []

[15 16 17 26 27 28 35 36]

Problem 3

(a)

1. Merge sort

首先考慮合併時，若一子區間中有相同元素，必為連續出現，又因為若左右子區間內有相同元素，會優先放入屬於左子區間的，且同一區間中之元素也是由左而右放入新序列，因此僅考慮合併前、後時，是 **stable**；並且任意長度超過 1 之子區間皆為其子區間合併而成，且長度為 1 之子區間是 **stable**，得任意子區間對於排序前之原序列皆為 **stable**

故 Merge sort 為 **stable**

2. Quick sort

考慮一序列 $[3\ 3'\ 4]$ ，因 *pivot* 之選取為隨機，因此以下排序過程為可能的：

$$[3\ 3'\ 4] \rightarrow [4\ 3'\ 3] \rightarrow [3'\ 4\ 3] \rightarrow [3'\ 3\ 4]$$

得一反例證明 Quick sort 並非 **stable**

3. Radix sort

由於元素由左至右放入其 *bucket* 之末端，因此相同元素放入 *bucket* 之操作為 **stable**；提出來時是由 *bucket* 之前端提出，放入序列末端，因此相同元素提出 *bucket* 之操作也是 **stable**

得 Radix sort 為 **stable**

(b)

在 *Data* 中使用一變數 *pos*，計為該元素在序列中的位置，比較時若值相同則比較 *pos*，這樣相同值之元素間排序前兩元素 *pos* 較小者會位於右側，排序後仍 *pos* 較小者仍會位於右側，故為 **stable**