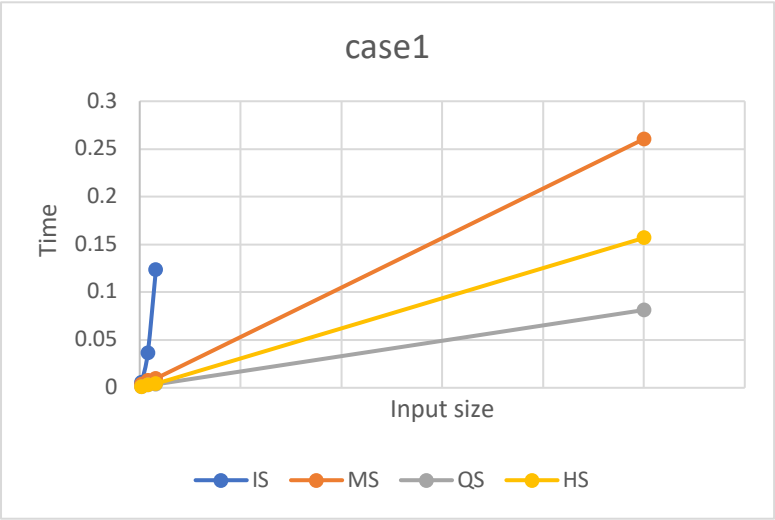
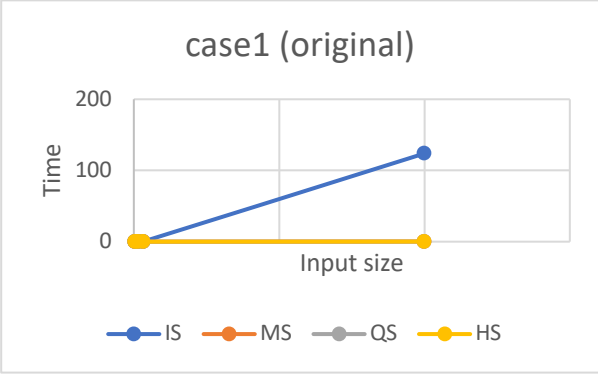
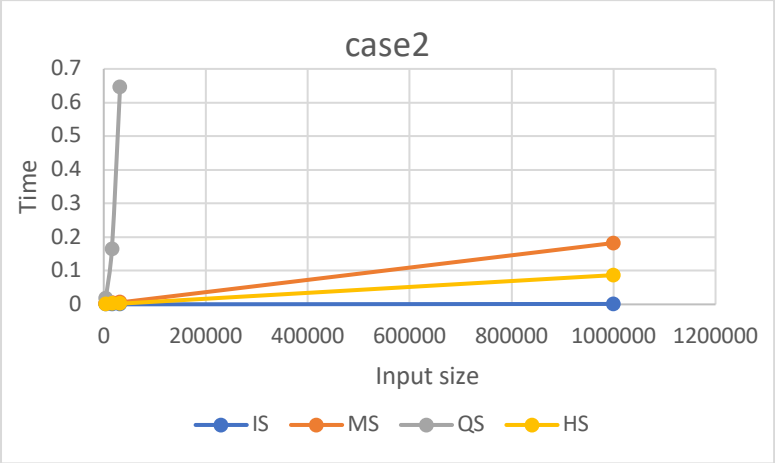
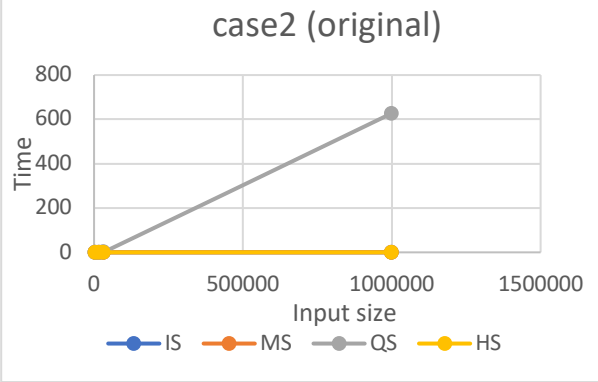
input size	IS		MS		QS		HS	
	CPU time(s)	Memory (KB)	CPU time(s)	Memory (KB)	CPU time(s)	Memory (KB)	CPU time(s)	Memory (KB)
4000.case2	0.000121	5892	0.002734	6028	0.018079	5960	0.000791	5892
4000.case3	0.008407	5892	0.002753	6028	0.014424	5892	0.000606	5892
4000.case1	0.005507	5892	0.003494	6028	0.000964	5892	0.000963	5892
16000.case2	0.00013	6044	0.004325	6044	0.165109	6672	0.002173	6044
16000.case3	0.064408	6044	0.004575	6044	0.123424	6288	0.001464	6044
16000.case1	0.036205	6044	0.007575	6044	0.00277	6044	0.002942	6044
32000.case2	0.000145	6176	0.005757	6304	0.646448	7488	0.002059	6176
32000.case3	0.244279	6176	0.006151	6304	0.477506	6728	0.002606	6176
32000.case1	0.123241	6176	0.00972	6304	0.003595	6176	0.00425	6176
1000000.case2	0.001279	12132	0.182448	16224	627.363	56836	0.087008	12132
1000000.case3	251.720	12132	0.187951	16224	330.857	27244	0.085963	12132
1000000.case1	123.897	12132	0.260448	16224	0.081425	12132	0.156897	12132

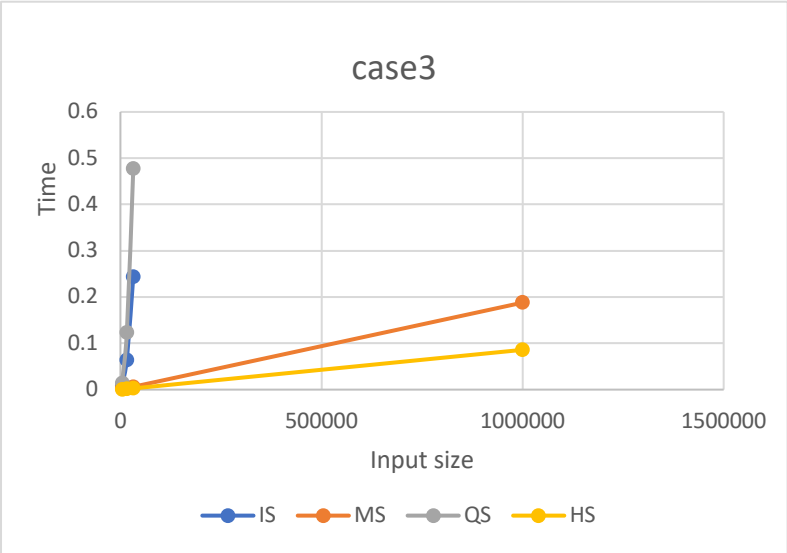
註：右圖為原始資料作圖，左圖為刪去時間過長的點後作圖

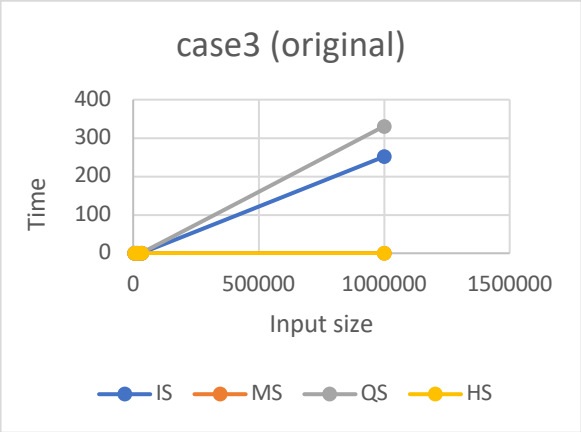












case1: random order

	Insertion sort	Merge sort	Quick sort	Heap sort
Time complexity	$O(n^2)$	O(nlgn)	O(nlgn)	O(nlgn)

case2: increasing order

	Insertion sort	Merge sort	Quick sort	Heap sort
Time complexity	O(n)	O(nlgn)	$O(n^2)$	O(nlgn)

case3: reverse order

	Insertion sort	Merge sort	Quick sort	Heap sort
Time complexity	$O(n^2)$	O(nlgn)	$O(n^2)$	O(nlgn)

1. Insertion sort:
   (1) 在 average case 時，所有排列方式機率相等，因此執行 insertion 時 iteration 約為 j/2 次，T(n) = $\sum_{j=2}^{n} \Theta(j/2)$ = $\Theta(n^2)$。
   (2) input 為 increasing order 時為 best case，即已完成排序，因此執行 insertion 時 iteration 只需 1 次，T(n) = $\sum_{j=2}^{n} \Theta(1)$ = $\Theta(n)$。
   (3) input 為 decreasing order 時為 worst case，即為倒序，因此執行 insertion 時 iteration 需完整走完 j 次，T(n) = $\sum_{j=2}^{n} \Theta(j)$ = $\Theta(n^2)$。
   (4) implementation: in-place
2. Merge sort:
   (1) 無論 input 如何排序，divide、conquer、combine 步驟都會全部走完，T(n) = 2T(n/2)+ Θ(1)+ Θ(n) = Θ(nlgn)。
   (2) implementation: not in-place，divide 後使用額外 vector 分別紀錄左右兩側的數值。
3. Quick sort
   (1) 以 average case 來說，partition 時 recursion tree 有部分會平衡分割，有部分不平衡，這些好或壞的分割隨機分佈，而上一次的 worst case 的缺點會被下一次的 best case 吸收，故最後 runtime 仍為 O(nlgn)。只要 partition 不是在最極端(sorted)時進行，T(n) = T(1-α)+T(α)+cn，runtime 仍可保持在 O(nlgn)。

(2) worst case 發生在 partition 時分成 n-1 : 0，也就是當 input 是 increasing order 或 reverse order 時，此時分割完都還是在同一側，partition 沒有太大作用，$T(n) = T(n-1) + \Theta(n) = \Theta(n^2)$。

(3) implementation: in-place

4. Heap sort

(1) 每一次 max heapify 需要 O(h)的時間(h: height of node)，而在高度 h 時最多有 $n/2^{h+1}$ 個 nodes，$T(n) = \sum_{h=0}^{lgn} O(h) = O(n)$，進行 extract max 時則是每次把 max(root)放到尾端，在進行 heapify 修正，T(n) = O(nlgn)。故完整的 heap sort runtime 為 O(nlgn)。

(2) implementation: in-place