資工三 110590029 陳思群

1.

(a) Each sublist with length k takes $\theta(k^2)$ worst-case time using Insertion Sort. To sort n/k such sublists, it take $n/k \times \theta(k^2) = \theta(nk)$ worst-case time

(b) Merging 2 sublists into one list take $\theta(n)$ worst-case time, we have $lg(n/k)$ such merges, so merging n/k sublists into one list takes $\theta(n\lg(n/k))$ worst-case time.

(c) In order for $\theta(nk + n\lg(n/k)) = \theta(n\lg n)$, either $nk = \theta(n\lg n)$ or $n\lg(n/k) = \theta(n\lg n)$. From the above two possibilities, we know the largest asymptotic value for k is $\theta(\lg n)$.

(d) In practice, k should be the largest list length on which insertion sort is faster than merge sort.

2.

(a)

```
BinarySearch(arr, target):
    left = 0
    right = len(arr) - 1
    while left <= right:
        mid = (left + right) / 2
        if arr[mid] == target:
            return mid
        else if arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1 // if target is not found
```

(b) $T(n) = T(\frac{n}{2}) + O(1)$

∵ 每次遞迴會將 arr/2

∴ Binary Search 時間複雜度為 $O(\lg n)$

3. $T(n) \le c_1(n-1)^2 + c_2 n$

$= c_1(n^2-2n+1)+c_2 n$

$= c_1 n^2 - 2c_1 n + c_1 + c_2 n$

In order to let $T(n) \le c_1 n^2$, we would set $-2c_1 n + c_1 + c_2 n \le 0$

$-2c_1 n + c_1 + c_2 n \le 0$

$c_1(1-2n) \le -c_2 n$

$c_1(2n-1) \ge c_2 n$

$c_1 \ge \frac{c_2 n}{2n-1}$

Because the largest value of $\frac{n}{2n-1}$ for $n \ge 1$ is 1, we can choose $c_1 = c_2$

$T(n) \le c_1 n^2 - 2c_1 n + c_1 + c_1 n$

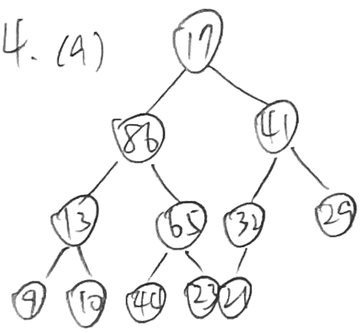
$= c_1 n^2 - c_1 n + c_1$

$\le c_1 n^2 = O(n^2)$

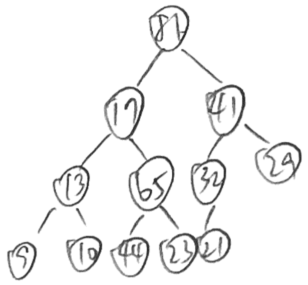
The procedure to prove lower bound is similar with upper bound.

Therefore, $T(n) = \theta(n^2)$

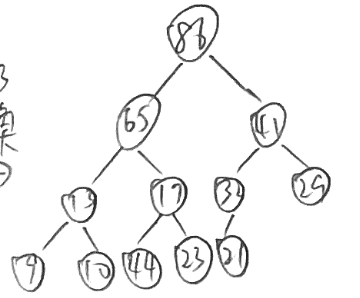
4. (a)



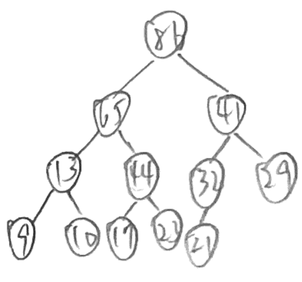
∵第1,2,3層已是MAX HEAP
∴從第0層開始
86>17>41,
所以86,17交換



∵65>17>13
∴65,17交換

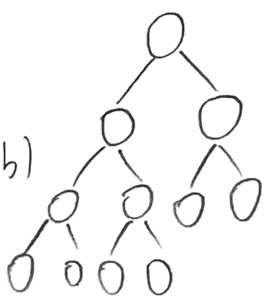


∵44>23>17
∴44,17交換



#

(b)



左圖為worst-case，樹的最後一層為半滿狀態。

右節點數為3 設為k，左節點數為$7 = 2k+1$

總節點數為$k+2k+1+1 = 3k+2$

左節點數佔總節點數比率為 $\frac{2k+1}{3k+2}$

$\lim_{k\to\infty} \frac{2k+1}{3k+2} = \frac{2}{3} \Rightarrow$ if total number of nodes is n,

worst-case running time is $\frac{2n}{3}$ #