Problem 1

(1)

假設 $n_0 = 1$

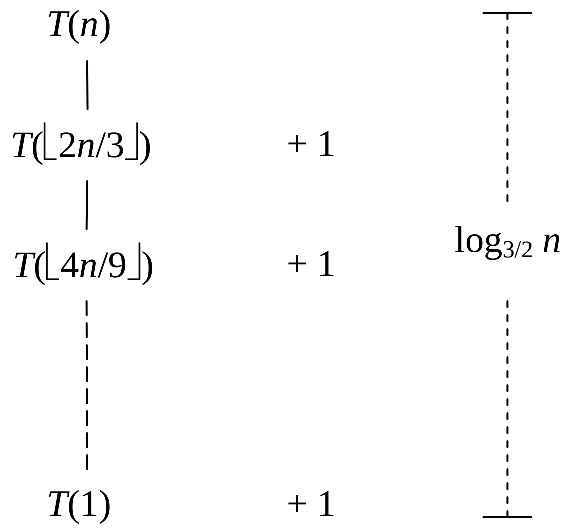
$0 < 9n^3 + 41n^2 \le cn^4$ for all $n \ge n_0 = 1$

$0 < 9/n + 41/n^2 \le c$

設 c = 50，因爲 $n \ge 1$ 所以 $0 < 9/n + 41/n^2 \le 50$ 一定成立

by the definition, $9n^3 + 41n^2 = O(n^4)$

(2)



由圖可知，$T(n) = \log_{3/2} n = O(\log n)$

Problem 2

(1)請參考期中考解答，不過因爲其範圍更改爲$[1, n^{\log\log n}]$，所以 radix sort 的部份需要多處理 loglog n 時間。

total time complexity is $O(n \log\log n)$.

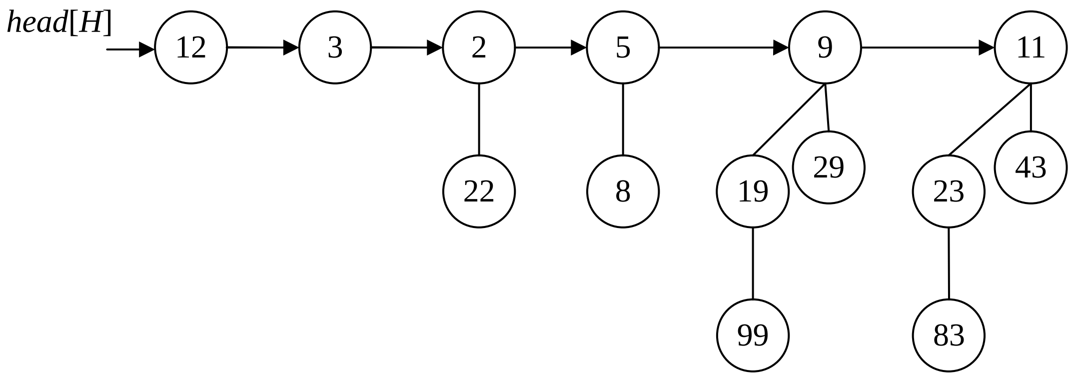
(2)請參考期中考解答，時間單位範圍更改爲$[1, n^3]$，所以 total time 變

大爲 n^4，整個時間分析變大爲 $O(n^5)$。

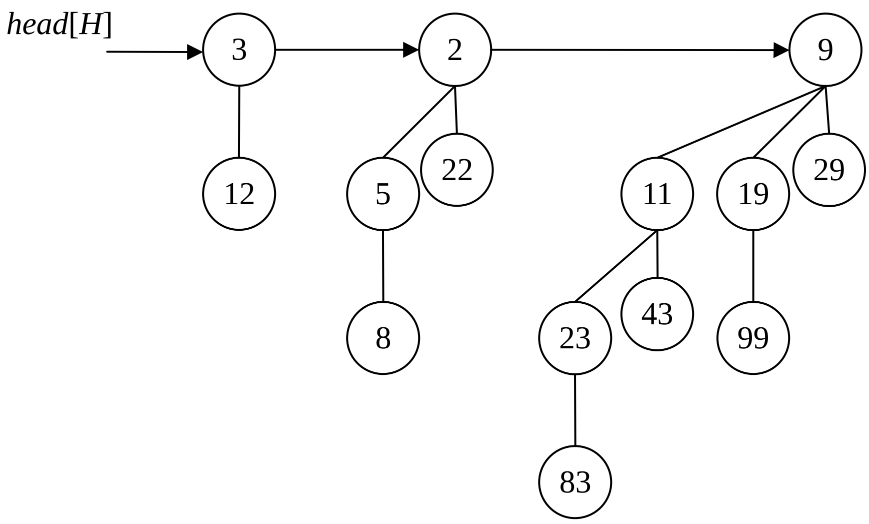
Problem 3

(1)

step 1: 先將兩個 binomial heap merge 成一個，依照 degree 大小 sort：



step 2: 然後一個一個 scan 過去，遇到兩個一樣大小的就依照大小關係合併起來。



最後會形成像上面圖形的 binomial heap。

(2)

假設兩個 heap 各是 n_1, n_2 個 node，則最多會有 $\log n_1$, $\log n_2$ 的 link

(let $n = n_1 + n_2$)

兩個步驟都花了：$O(\log n_1 + \log n_2) = O(\log n)$

Problem 4

(請參考課本 23.2)

(1)

Prim’s algorithm: vertices in A always form a single tree.

```
MST-PRIM(G, w, r)
 1  for each u ∈ V[G]
 2      do key[u] ← ∞
 3         π[u] ← NIL
 4  key[r] ← 0
 5  Q ← V[G]
 6  while Q ≠ Ø
 7      do u ← EXTRACT-MIN(Q)
 8         for each v ∈ Adj[u]
 9             do if v ∈ Q and w(u, v) < key[v]
10                   then π[v] ← u
11                        key[v] ← w(u, v)
```

(2) Implement priority queue Q as a binary heap

Steps 1~5:	$O(V)$	(Build Q)
Step 7:	$O(V\lg V)$	(V times *Extract-Min*)
Steps 8~11:	$O(E\lg V)$	(E times *Decrease-Key*)

$T(m, n)=O(E\lg V)$

Problem 5

(請參考課本 22.4)

(1)

TOPOLOGICAL-SORT(G)

call DFS(G) to compute finishing times $f[v]$ for each vertex v
as each vertex is finished, insert it onto the front of a linked list
return the linked list of vertices

(2)

Output vertices in order of decreasing $f[u]$.

$O(V+E)$

Problem 6.

方法

(1)找尋 P 中 y 座標最小的點，如果有兩點以上取 x 座標最小的。

(2)設那點爲 s_0，把已給定的 P 逆時針順序從 s_0 開始重新編號成 s_0, s_1, …, s_n。

(3)從 s_0 開始，push s_0, s_1 到 stack S，接下來從 s_2 開始和前一條檢查是否爲向左轉，若不是則 pop 到是左轉或是只剩下一點爲止再 push，一直做到 s_n。

(4)stack S 內的點即爲逆時針順序的 convex hull(bottom to top)。

時間

(1)$O(n)$
(2)$O(n)$

(3)$O(n)$ 一個點頂多被 push pop 各一次

(4)$O(n)$
Total = $O(n)$

正確性

上述方法是照原本的圖形周圍磨一圈，因爲原本的圖形已經包含了所有的點，照著周圍依序磨一圈，只會把內縮的拿掉，依然會包住所有的點，而因爲保證每一步都是左轉，且頂點都在原本的點上(被釘住無法縮更小)，故此爲 convex hull。

Problem 7.

方法

(1)原本 bipartite graph 左邊的點稱作 L_i，右邊的點稱作 R_i。

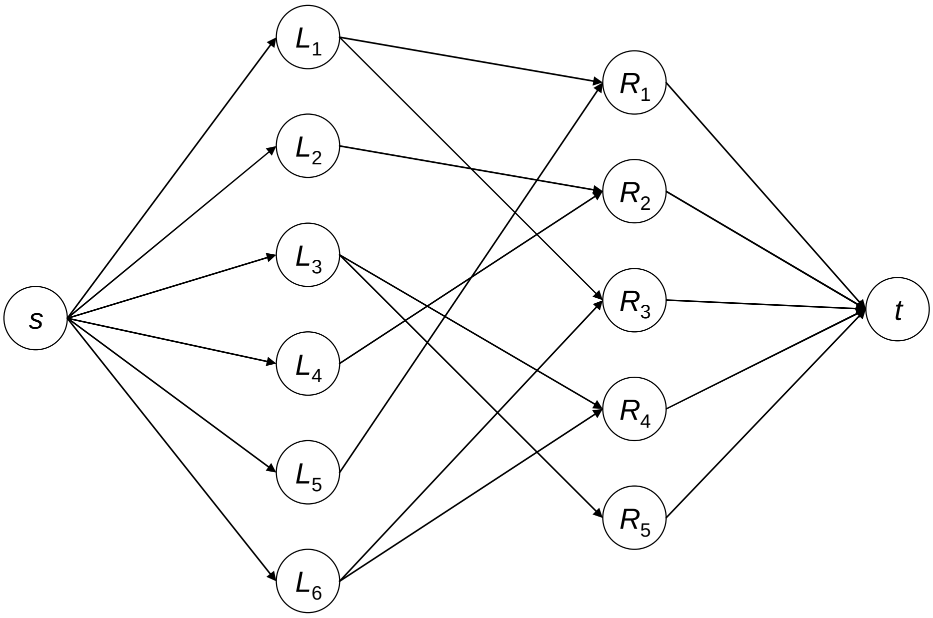
(2)L_i 和 R_i 相連的 edge 都改成 direct edge，方向爲 L_i 指向 R_i。

(3)新增一個點 s，對每個點 L_i 新增一條 direct edge 指向 L_i。

(4)新增一個點 t，每個點 R_i 新增一條 direct edge 指向 t。

(5)所有 edge 的流量設爲 1，以 s 爲起點 t 爲終點，跑 *max-flow* algorithm

(6)最後出來的流量 f 即爲最大 match 數，有用到的 edge 爲 match 的配對。



時間

(1)$O(V)$
(2)$O(E)$
(3)$O(V)$
(4)$O(V)$

(5)$O(Ef) = O(EV)$　　流量不可能比點多

(6)$O(E)$
Total = $O(EV)$

Problem 8.
(a)
Approx-Subset-Sum(S, t, ε)

$n \leftarrow |S|$
$L_0 \leftarrow \{0\}$
For $i \leftarrow 1$ to n
    $L_i \leftarrow$ *Merge-Lists*$(L_{i-1}, L_{i-1} + x_i)$
    $Li \leftarrow$ *Trim*$(L_i, \varepsilon/2n)$
    Remove from L_i every element that is greater than t
Let z^* be the largest value in L_n
return z^*

Let $L = (y_1, y_2, \ldots, y_m)$

Trim(L, δ)

$m \leftarrow |L|$
$L' \leftarrow \{y_1\}$
$last \leftarrow y_1$
For $i \leftarrow 2$ to m
    if $y_i > last * (1 + \delta)$
        append y_i onto the end of L'
            $last \leftarrow y_i$
return L'

(b)

Approx-Subset-Sum(S, t, ε)

$n \leftarrow |S|$
$L_0 \leftarrow \{\sum x_i\}$
For $i \leftarrow 1$ to n
    $L_i \leftarrow$ *Merge-Lists*$(L_{i-1}, L_{i-1} - x_i)$
    $Li \leftarrow$ *Trim*$(L_i, \varepsilon/n)$
    Remove from L_i every element that is smaller than t
Let z^* be the smallest value in L_n
return z^*

Let $L = (y_1, y_2, \ldots, y_m)$

Trim(L, δ)

$m \leftarrow |L|$
$L' \leftarrow \{y_m\}$
$last \leftarrow y_m$
For $i \leftarrow m - 1$ to 1
    if $y_i < last / (1 + \delta)$
        append y_i onto the head of L'
            $last \leftarrow y_i$
return L'

Bonus

(a) 請參考作業 12

(b) 請參考作業 13