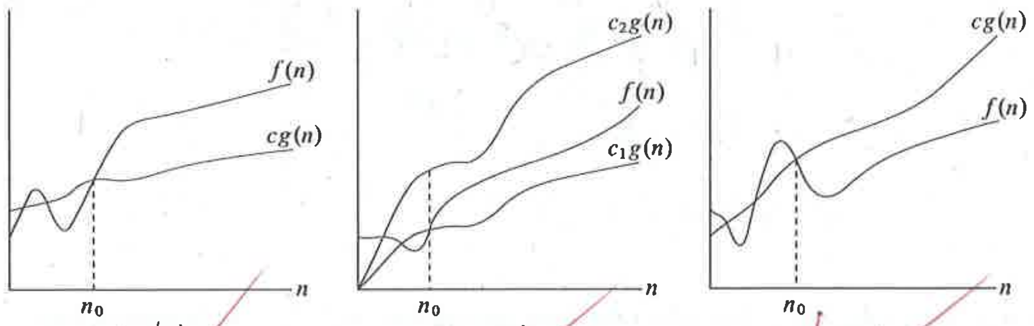
90

Algorithm Midterm (2016 Spring)

ID: B10315013 Name: 周紀愷

1. Please select the appropriate notation for the following figures: (6%)

(1) $f(n) = \Theta(g(n))$ (2) $g(n) = \Theta(f(n))$ (3) $f(n) = O(g(n))$ (4) $g(n) = O(f(n))$
(5) $f(n) = \Omega(g(n))$ (6) $g(n) = \Omega(f(n))$ (7) $f(n) = o(g(n))$ (9) $g(n) = o(f(n))$
(9) $f(n) = \omega(g(n))$ (10) $g(n) = \omega(f(n))$



(a) (5) (b) (1) (c) (3)

2. (d) For the following statements, which one is incorrect? (2%)

(a) $2n = O(n^2)$ (b) $2n^2 = O(n^2)$ (c) $2n = o(n^2)$ (d) $2n^2 = o(n^2)$

3. (a) What does "programming" mean in "dynamic programming"? (2%)

建表的方式(tabular method)

(b) What is the major difference between "divide-and-conquer" and "dynamic programming"? (3%)

divide-and-conquer 用切的方式(主要是二分法)找出分別的最佳解再組成最佳解

dynamic programming 是從最小單位或是最前頭建表，大單位或是後頭利用前面的小單位找出最佳解

差別：前者 一個是一邊切一邊找，一個是先找好再最後直接取用

-2

(c) What is the major difference between "dynamic programming" and "greedy algorithm"? (3%)

greedy 是找出目前狀態下最好的方式

差別：dynamic 會根據其它資訊組出表

greedy 只有判別當前狀況

↳因此有些情況需要跟別的資訊分析，greedy 就無法解出

4. Recall in Chapter 4, we talk about the problem of buying one unit of stock one time and then selling it at a later date to maximize your profit. Please explain the basic idea of how to using divide-and-conquer to solve the problem. (6%)

先把股票走勢轉成單時增加量的表

利用 divide-and-conquer 找出最大區間

找最大區間：① 把目前的範圍切一半

② 找左邊最大區間、右邊最大區間跟跨中間最大区間

③ 挑最大的回傳

利用找左右最大区間來 recursive 達到目的

5. Given a chain $\langle A_1, A_2, \ldots, A_n \rangle$ of n matrices, where for $i = 1, 2, \ldots, n$, matrix A_i has dimension $p_{i-1} \times p_i$, fully parenthesize the product $A_1 A_2 \ldots A_n$ in a way that minimizes the number of scalar multiplications.

(a) What does "fully parenthesize" mean? (3%)

-1

將 $A_1 \sim A_n$ 用括號分開，~~分~~到完全是兩兩相乘的狀態

(b) Suppose that to optimally parenthesize $A_i A_{i+1} \ldots A_j$, we split the product between A_k and A_{k+1}. Let $m[i, j]$ be the minimum number of scalar multiplications needed to compute the matrix $A_{i..j}$; for the full problem, the lowest-cost way to compute $A_{1..n}$ would thus be $m[1, n]$. Please define $m[i, j]$. (3%)

$$m[i,j] = \begin{cases} 0, & \text{if } i = j \\ \min\limits_{k=i \text{ to } j-1} \left(m[i,k] + m[k+1,j] + P_{i-1}P_kP_j \right), & \text{if } i < j \end{cases}$$

(c) Let $s[i, j]$ be a value of k at which we split the product $A_i A_{i+1} \ldots A_j$ in an optimal parenthesization. Please define $s[i, j]$. (3%)

$$s[i,j] = \{ k \mid k \in \min\limits_{k=i \text{ to } j-1} (m[i,k] + m[k+1,j] + P_{i-1}P_kP_j),$$

$$\text{if } i < j$$

(d) Suppose the sequence of dimensions $p = \langle p_0, p_1, \ldots, p_n \rangle$ is $\langle 5, 10, 3, 12, 5, 50, 6 \rangle$. Please complete the following tables. (12%)

A_1	A_2	A_3	A_4	A_5	A_6
5×10	10×3	3×12	12×5	5×50	50×6

(i, 1~3) (2~4) 1,4

× 1950 + 5×10×6

✓ 1920 + 5×3×6 = 2010

1905 + 5×5×6 = 2055

1655 + 5×50×6 = 3155

1500

m

j \ i	1	2	3	4	5	6
1	0					
2	150	0				
3	330	360	0			
4	405	330	180	0		
5	1655	2430	930	3000	0	
6	2010	1950	1770	5100	1500	0

s

j \ i	1	2	3	4	5
2	1				
3	2	1			
4	2	2	3		
5	4	2	4	4	
6	2	2	4	4	5

−1

(e) The initial call `PRINT-OPTIMAL-PARENS(s,1,n)` prints an optimal parenthesization of $\langle A_1, A_2, \ldots, A_n \rangle$. Please complete the following function. (4%)

PRINT-OPTIMAL-PARENS(s, i, j)

```
1  if i == j
2      print "A"_i
3  else print "("
4      PRINT-OPTIMAL-PARENS(s, i, s[i,j])
5      PRINT-OPTIMAL-PARENS(s, s[i,j]+1, j)
6      print ")"
```

(f) Following (d) and (e), what would be printed when we call `PRINT-OPTIMAL-PARENS(s,1,6)`? (3%)

−1

((A1A2)((A3A4)(A5A6)))

6. (a) Please define "prefix codes". (3%)

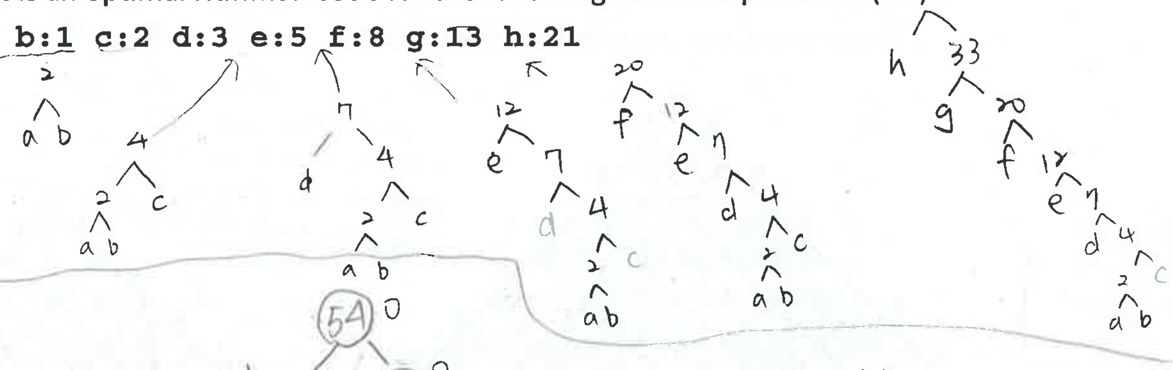
code的前面不管幾碼都無法找到有对應的值，一定要有完整的code → 可以免除誤判

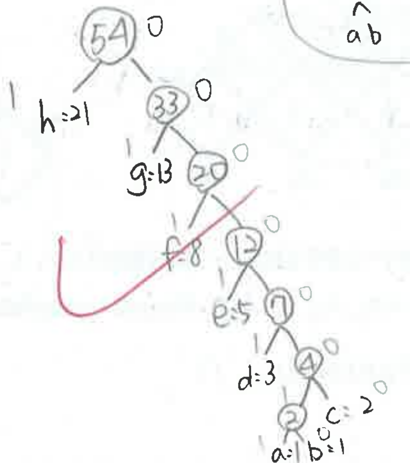
(b) Please define "full binary tree". (3%)

每個節点(node)都有2個子樹或是沒有子樹

(c) What is an optimal Huffman code for the following set of frequencies? (6%) 54

`a:1 b:1 c:2 d:3 e:5 f:8 g:13 h:21`





a: 00000011
b: 00000010
c: 0000000
d: 000001
e: 00001
f: 0001
g: 001
h: 1

7. Determine the cost and structure of an optimal binary search tree for a set of $n = 7$ keys with the following probabilities:

i	0	1	2	3	4	5	6	7
p_i		0.04	0.06	0.08	0.02	0.10	0.12	0.14
q_i	0.06	0.06	0.06	0.06	0.05	0.05	0.05	0.05

(a) The following pseudocode has some bugs. Please find them out. (8%)

```
OPTIMAL-BST(p, q, n)
let e[1..n+1, 0..n], w[1..n+1, 0..n],
        and root[1..n, 1..n] be new tables
for i = 1 to n + 1
    e[i, i-1] = q_{i-1}
    w[i, i-1] = q_{i-1}
for l = 1 to n
    for i = 1 to n - l        n-l+1
        j = i + l             i+l-1
        e[i, j] = ∞
        w[i, j] = w[i, j-1] + p_i + q_j        p_j
        for r = i to j
            t = e[i, r-1] + e[r-1, j] + w[i, j]        r+1
            if t < e[i, j]
                e[i, j] = t
                root[i, j] = r
return e and root
```