1. (12 คะแนน) จาก master method ที่กำหนดให้ จงหาความซับซ้อนเชิงเวลาจาก Recurrent Relation T(N) ในแต่ละข้อ โดยให้เติมคำในช่องว่าง สำหรับทุก ๆ ข้อให้ถือว่า $T(n) = \Theta(1)$

$$T(n) = aT(\frac{n}{b}) + cn^{k},$$

with a, b, c, k are positive constants, and $a \le 1$ and $b \le 2$,

$$T(n) = \begin{cases} O(n^{\log_b a}), & \text{if } a > b^k \\ O(n^k \log_{\mathbb{N}} n), & \text{if } a = b^k \\ O(n^k), & \text{if } a < b^k \end{cases}.$$

_		
ข้อ	Recurrent	ความซับซ้อนเชิงเวลา
(1)	$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n)$	
(2)	T(n) = 9T(n/3) + n	
(3)	T(n) = T(2n/3) + 1	
(4)	$T(n) = 3T(n/4) + n \lg n$	
(5)	$T(n) = 3 T(n/2) + n^{3/2}$	
(6)	To multiply two <i>n</i> -by- <i>n</i> matrices <i>A</i> and <i>B</i> : • Divide: partition <i>A</i> and <i>B</i> into $\frac{1}{2}n$ -by- $\frac{1}{2}n$ blocks. • Conquer: multiply 8 pairs of $\frac{1}{2}n$ -by- $\frac{1}{2}n$ matrices, recursively. • Combine: add appropriate products using 4 matrix additions. $\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \qquad C_{12} = (A_{11} \times B_{11}) + (A_{12} \times B_{22}) \\ C_{21} = (A_{21} \times B_{11}) + (A_{22} \times B_{21}) \\ C_{22} = (A_{21} \times B_{12}) + (A_{22} \times B_{22}) \end{bmatrix}$	
	$T(n) = 8T(n/2) + n^2$	

2. จงเติมความซับซ้อนเชิงเวลาของอัลกอริทึมต่อไปนี้เมื่อใช้เรียงข้อมูลในอาเรย์ (ข้อละ 1 คะแนน)

อัลกอริทึม	best	average	worst
Quicksort			
Mergesort			
Heapsort			

ข้อสอบย่อยวิชา 2110327 การออกแบบอัลกอริทึม คะแนนเต็ม 10 คะแนน				
เลขประจำตัวนิสิตชื่อ-สกุล				
Bubble Sort				
Insertion Sort				
Selection Sort				

3. (5 คะแนน) ในการเลือกกิจกรรมให้จัดในห้องประชุมแห่งหนึ่ง สมมติให้มีกิจกรรม n กิจกรรม มี หมายเลขกำกับตั้งแต่ 1 ถึง n กิจกรรมแต่ละกิจกรรมจะมีเวลาเริ่มต้นและเวลาสิ้นสุดกำกับไว้ และมี จำนวนคนที่จะเข้าร่วมแต่ละกิจกรรมกำกับอยู่ เราจะต้องเลือกว่ากิจกรรมใดบ้างที่จะได้ใช้ห้องประชุม โดยที่มีเป้าหมายคือเพื่อให้ได้ผลรวมของจำนวนคนที่มาเข้าร่วม รวมทุกกิจกรรมที่เลือกไว้นั้นมากที่สุด อย่างไรก็ตามกิจกรรมที่เลือกนั้นต้องไม่ใช้เวลาซ้อนทับกัน (ถ้าเวลาสิ้นสุดของกิจกรรมหนึ่งเท่ากับเวลาเริ่ม ของอีกกิจกรรมหนึ่งพอดีจะถือว่าไม่ซ้อนทับกัน)

กิจกรรมดังกล่าวนั้นเรียงตามลำดับที่ของเวลาเลิกใช้กิจกรรม กล่าวคือ กิจกรรมที่ i จะมีเวลาสิ้นสุดการขอ ใช้สถานที่ก่อนกิจกรรมที่ j เมื่อ i < j

กำหนดให้ P(j) เป็นหมายเลขกิจกรรมที่มากที่สุดที่ขอใช้สถานที่มาก่อนกิจกรรม j โดยเวลาไม่ทับซ้อนกัน กำหนดให้ OPT(n) คือ ฟังก์ชันที่คำนวณผลรวมจำนวนคนเข้ากิจกรรมที่เลือกทั้งหมดที่มากที่สุดเท่าที่ เป็นไปได้ ทั้งหมด เมื่อพิจารณาถึงกิจกรรมที่ n

โจทย์ข้อนี้ต้องการให้นิสิตคำนวณค่า OPT(i) โดยไม่จำเป็นต้องเขียน recurrence แต่ให้เขียนค่าลงไปใน ตารางด้านล่างนี้ ซึ่งได้กำหนดเวลาเริ่มต้นและเวลาสิ้นสุดของแต่ละกิจกรรมมาแล้ว พร้อมด้วยจำนวนคน ที่จะเข้าร่วมกิจกรรมต่าง ๆ

กิจกรรม	1	2	3	4	5	6	7	8	9
เริ่ม	04.00	02.00	06.00	08.00	11.00	15.00	11.00	12.00	18.00
เลิก	07.00	09.00	10.00	13.00	14.00	17.00	18.00	19.00	20.00
จน.คน	30	70	40	50	30	20	70	70	20
P(i)									
Opt(i)									

ข้อสอบย่อยวิชา 211032	?7 การออกแบบอัลกอริทึม	คะแนนเต็ม 10	คะแนน
เลขประจำตัวนิสิต	ชื่อ-สกุล		

3. จงเขียนเติมส่วนของโค้ดเมื่อต้องการเขียน recurrences ต่อไปนี้ แบบ bottom-up 3.1 (8 คะแนน) กำหนดให้มี string \times โดยที่ $x_{i...j}$ คือ substring ตั้งแต่ตัวที่ i ถึง j และให้ $D_{i,j}$ เป็น จำนวนตัวอักษรที่น้อยที่สุดที่ต้องแทรกเพิ่มเข้าไปใน คำที่ประกอบด้วยอักษร $x_{i...j}$ เพื่อให้คำนั้นเป็น palindrome (คือคำอ่านจากหน้าหรือหลังได้คำเดียวกัน) คำที่มีอักษรตัวเดียวหรือสองตัวก็ถือว่าเป็น palindrome

$$D_{ij} = \begin{cases} 1 + \min\{D_{i+1,j}, D_{i,j-1}\} & x_i \neq x_j \\ D_{i+1,j-1} & x_i = x_j \end{cases}$$

3.2 (8 คะแนน) กำหนดตาราเมทริกซ์ $cost\Pi\Pi$ ซึ่งระบต้นทนในการเดินผ่านช่องแต่ละช่อง (i.i) เมื่อ

3.2 (8 คะแนน) กำหนดตาราเมทริกซ์ cost[][] ซึ่งระบุต้นทุนในการเดินผ่านช่องแต่ละช่อง (i,j) เมื่อ เราต้องการหาวิถีที่มีต้นทุกต่ำสุดจากช่อง (0,0) ไปถึงช่อง (i,j) จะได้ ฟังก์ชัน MinCost() ดังนี้

MinCost(i,j) = min(MinCost(i-1,j),MinCost(i,j-1)) + Cost[i][j]

4. จงพิจารณาปัญหาต่อไปนี้ แล้วระบุ recurrences และ ผลเฉลยที่ base cases ของปัญหา
4.1 (8 คะแนน) กำหนดตัวเลขจำนวนเต็ม **n** ตัวจงหา D(n) เมื่อ D(n) คือจำนวนของลำดับที่แตกต่างกัน ที่ผลรวมของสมาชิกในลำดับนั้นเป็น n และสมาชิกในลำดับมีค่าได้เพียงแค่ 1, 3 และ 4 (ตัวอย่างเช่น D(4) = 4 เพราะว่ามีลำดับที่แตกต่างกัน 4 ลำดับต่อไปนี้ ซึ่งผลรวมของสมาชิกเป็น 4 {1,1,1,1}, {1,3}, {3,1} และ {4}

ขอสอบย่อยวิชา 2	110327 การออกแบบอัลกอริทึม คะแนนเต็ม 10 คะแนน
เลขประจำตัวนิสิต	ชื่อ-สกุล
Recurrences	
Base cases	
เครื่องหมาย and, ตัวอย่างเช่นเมื่อ s & T จากนิพจน์ที่ใน	ะแนน) กำหนด Boolean expression ให้ มีค่าความจริง n ตัว เชื่อมกันด้วย or และ xor โดยที่ s[i] คือค่าความจริงแต่ละตัว และ o[i] คือเครื่องหมายแต่ละตัว = { T, F, T} และ o = { ^, & } หมายความว่า boolean expression ที่ให้คือ T ^ F ห้นี้ เราสามารถใส่วงเล็บได้ 2 แบบ คือ "((T ^ F) & T)" and "(T ^ (F & T))" โดยทีทั้ง นผลได้เป็น true ทั้งคู่
	T(i,j) แทนจำนวนวิธีในการใส่วงเล็บตั้งแต่ค่าความจริง i ถึง j เพื่อให้การคำนวณผลมี rue และ F(i,j) แทนจำนวนวิธีในการใส่วงเล็บตั้งแต่ค่าความจริง i ถึง j เพื่อให้การ มจริงเป็น True
Recurrences	
Base cases	