ชื่อ _____ รหัส _____ section ___

สอบปลายภาค: ภาคการศึกษาที่ 2 **ปีการศึกษา**: 2556

วันสอบ: 7 มีนาคม 2557 **เวลาสอบ**: 13.30 – 16.30 น.

ห้องสอบ: A400, A401

ผู้สอน: อ.เสกสรรค์ สุวรรณมณี อ.อัมรินทร์ ดีมะการ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

รหัสและชื่อวิชา: 242-310 Introduction to Algorithm and Complexity

แนะนำขั้นตอนวิธีและความซับซ้อน

ทุจริตในการสอบมีโทษขั้นต่ำคือ ปรับตกในรายวิชาที่ทุจริตและพักการเรียน 1 ภาคการศึกษา

คำสั่ง: อ่านรายละเอียดของข้อสอบ และคำแนะนำให้เข้าใจก่อนเริ่มทำข้อสอบ

อนุญาต: เครื่องเขียนต่างๆ เช่น ปากกา หรือดินสอ เข้าห้องสอบ และ กระดาษขนาด A4 หนึ่งแผ่น จดบันทึกด้วยลายมือเท่านั้น (ห้าม print หรือ ถ่ายเอกสาร)

ไม่อนุญาต: หนังสือ หรือเครื่องคิดเลขเข้าห้องสอบ และ**เอกสารใดๆ เข้าและออกห้องสอบ**

เวลา: 3 ชั่วโมง (180 นาที)

คำแนะนำ

- ☐ ข้อสอบมี 13 **หน้า** (*รวมหน้าปก*) แบ่งออกเป็น 5 **ตอน** คะแนนรวม 105 คะแนน (คิดเป็นคะแนน 35%)
- ☐ **<u>เขียนคำตอบในข้อสอบ คำตอบส่วนใดอ่านไม่ออก จะถือว่าคำตอบนั้นผิด</u>**
- ☐ อ่านคำสั่งในแต่ละข้อให้เข้าใจก่อนลงมือทำ
- ☐ เวลาที่ใช้เวลาทำตอนให้เหมาะสม ตามคำแนะนำ
- ☐ หากข้อใดเขียนคำตอบไม่พอ ให้เขียนเพิ่มที่ด้านหลังของหน้านั้นเท่านั้น
- ☐ *ให้ส่งกระดาษจดโน้ตพร้อมข้อสอบ* เขียนชื่อ-รหัสนักศึกษา ให้ชัดเจน

ตอน	**1** (17)	**2** (23)	**3** (20)	**4** (30)	**5** (15)	**รวม** (105) 35%
คะแนน						

นักศึกษารับทราบ ลงชื่อ .....

ตอนที่ 1 (17 คะแนน, 35 นาที)

การแบ่งแยกและเอาชนะ (Divide and Conquer)

1. จงอธิบายหลักการของอัลกอริทึมแบบ Divide and conquer และอธิบายสามขั้นตอนหลักมีอะไรบ้าง และให้ยกตัวอย่างอัลกอริทึมแบบ divide and conquer ที่ใช้ในการแก้ปัญหาได้อย่างมีประสิทธิภาพ (5 คะแนน)

2. อัลกอริทึม A และ B ใช้หลักการ Divide and conquer ในการแก้ปัญหาอย่างหนึ่ง มีลักษณะของการทำงานของอัลกอริทึมดังนี้

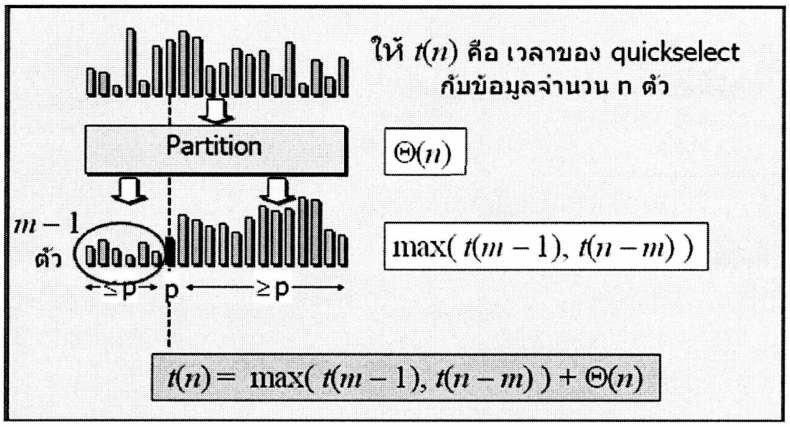
Algorithm A. แก้ปัญหาขนาด *n* โดยการแบ่งเป็น 9 ปัญหาย่อยขนาด *n/3* และแก้ปัญหาด้วยการเรียกซ้ำ (recursion) และการรวมคำตอบของปัญหาย่อยใช้เวลา $O(n^2)$

Algorithm B. แก้ปัญหาขนาด *n* โดยการแบ่งเป็น 2 ปัญหาย่อยขนาด *n-1* และแก้ปัญหาด้วยการเรียกซ้ำ (recursion) และการรวมคำตอบของปัญหาย่อยใช้เวลาเป็นค่าคงที่ O(1)

จงเขียนสมการการเวียนเกิด (Recurrence relation) ของเวลาของอัลกอริทึมทั้งสอง (***t(n)***) เพื่อหาเวลาของอัลกอริทึมในรูป Big-O แล้วเปรียบเทียบว่าอัลกอริทึมใดมีประสิทธิภาพเชิงเวลาดีกว่ากัน (6 คะแนน)

คำตอบข้อ 2 (ต่อ)

3. จากปัญหาการเลือก (Selection Problem: select the k-th min) มีการใช้อัลกอริทึม QuickSelect ซึ่งใช้หลักการ Partition แบ่งข้อมูลโดยใช้เวลาเป็น $\Theta(n)$ และมีสมการเวลา $t(n)$ ของอัลกอริทึมนี้ในลักษณะความสัมพันธ์เวียนเกิด (recurrence relation) ดังรูป จากสมการดังกล่าวจงวิเคราะห์เวลาการทำงานในรูป Big-O ในกรณีที่ดีที่สุด(best-case analysis) และ กรณีที่แย่ที่สุด (worst-case analysis) ของอัลกอริทึมนี้ (6 คะแนน)



t(n) = max(t(m-1), t(n-m)) + Θ(n)

/********** **จบตอนที่ 1** **********/

ตอนที่ 2 (23 คะแนน, 50 นาที)

Dynamic Programming Algorithm

1. หลักการของ Dynamic Programming มีการจัดการแก้ปัญหาในลักษณะ Top-down หรือ Bottom-up จงอธิบายพอสังเขป (5 คะแนน)

2. สัมประสิทธิ์ทวินามหรือ Binomial Coefficient มีนิยามดังนี้

$$C(n,k) = \begin{cases} 1, & k = 0 \; or \; k = n \\ C(n-1,k-1) + C(n-1,k), & k < n \end{cases}$$

$$for \; 0 \leq k \leq n \; and \; n \geq 0$$

จงเขียนอัลกอริทึมสำหรับการคำนวณ *C(n, k)* โดยใช้หลักการ Dynamic Programming (6 คะแนน)

3. Container Truck Problem (12 คะแนน)

มีรถบรรทุกคันหนึ่ง บรรจุสินค้าได้ปริมาตรสูงสุด *Vm* (max volume of container) มีสินค้าที่ต้องการบรรจุในรถบรรทุกจำนวน *n* ชิ้น แต่ละชิ้นมีปริมาตร *v1,v2,...,vn* และ น้ำหนัก *w1,w2,...,wn* ตามลำดับ ในการขนส่งสินค้าของรถบรรทุก มีต้นทุนการขนส่งแปรผันตรงกับน้ำหนักของสินค้าที่บรรทุก(น้ำหนักมากจะใช้ต้นทุนขนส่งมาก) จึงต้องหาวิธีการบรรจุสินค้าให้ได้มากที่สุดตามปริมาตรของรถบรรทุก แต่ให้ได้น้ำหนักของสินค้าที่เบาที่สุดเพื่อให้ประหยัดต้นทุน (สมมุติว่าถ้าสินค้ามีปริมาตรรวมไม่เกินความจุ สามารถหาวิธีบรรจุสินค้าในรถได้เสมอ)

จงใช้ Dynamic Programming สำหรับแก้ปัญหานี้ (***Hint:*** 0/1 Knapsack problem)

3.1) กำหนดให้รูปแบบของผลเฉลยคืออาร์เรย์ *x[1...n]* โดย x_i มีค่าเป็น 1 หรือ 0 (หมายถึงเลือกหรือไม่เลือกสินค้าใส่รถบรรทุก) จงเขียนฟังก์ชันจุดประสงค์ (Objective function) และเงื่อนไข(constraint) ของปัญหานี้ (3 คะแนน)

3.2) จงเขียนความสัมพันธ์เวียนเกิดของ Optimal substructure ของปัญหานี้ (3 คะแนน)

(***Hint:*** M(i ,j) = *?*)

3.3) จงเขียนอัลกอริทึมเพื่อหาว่ารถบรรทุกจะบรรจุน้ำหนักที่เบาที่สุดได้เท่าใด (Optimal solution) (6 คะแนน)
(Bonus: หากสามารถทราบได้ด้วยว่าต้องบรรจุของชิ้นใดบ้าง จะได้คะแนนเพิ่ม)
(***Hint:*** ContainerDP(v[1..n], w[1..n], Vm))

/********** **จบตอนที่** 2 **********/

ตอนที่ 3 (20 คะแนน, 35 นาที)

Greedy Algorithm

1. จงระบุว่าถูกหรือผิด (√ หรือ X) ในแต่ละข้อต่อไปนี้ สำหรับอัลกอริทึมเชิงละโมบ(Greedy algorithm) และ Dynamic Programming (6 คะแนน)

	Greedy	Dynamic Programming
1. ใช้แก้ปัญหาที่เป็น Optimization problem		
2. ปัญหามี Optimal substructure		
3. ได้คำตอบเป็น globally optimal solution เสมอ		
4. แก้ปัญหา 0/1 knapsack problem ได้		
5. แก้ปัญหา fractional knapsack problem		
6. แก้ปัญหาการทอนเหรียญได้ดี (รวดเร็ว)		

2. ปัญหาการเลือกกิจกรรม Activity Selection (14 คะแนน)

มีกิจกรรม n อย่าง(a1,a2,..,an) ที่ต้องการจัดให้ห้องๆหนึ่ง มีการจองเวลาใช้งานเริ่มต้นและสิ้นสุดแต่ละกิจกรรมเป็น *s1,s2,...,sn* และ *f1,f2,...fn* จะเลือกกิจกรรมอย่างไรให้ได้จำนวนกิจกรรมมากที่สุด โดยที่ใช้ห้องไม่เหลื่อมล้ำกัน (เช่น ถ้า $f_i \leq s_j$ สามารถจัดกิจกรรม i และ j ได้)

2.1) จงอธิบายหลักการแก้ปัญหา Activity selection ด้วย Greedy algorithm (4 คะแนน)

2.2) จากข้อมูลตัวอย่างปัญหาการเลือกกิจกรรม ดังข้างล่างนี้ (ข้อมูลเวลาเริ่มต้นและสิ้นสุดของ 10 กิจกรรม $a_1, a_2,\ldots,a_{10}$) จงหาว่าจำนวนกิจกรรมสูงสุดที่เลือกได้เป็นเท่าไร และต้องเลือกกิจกรรมใดบ้าง (10 คะแนน)

Activity Start time *s[1..10]* = { 1, 4, 5, 8, 6, 9, 7, 4, 5, 2 }

Activity Finish time *f[1..10]* = { 6, 5, 8, 11, 7, 12, 8, 7, 6, 4 }

/********** **จบตอนที่** 3 **********/

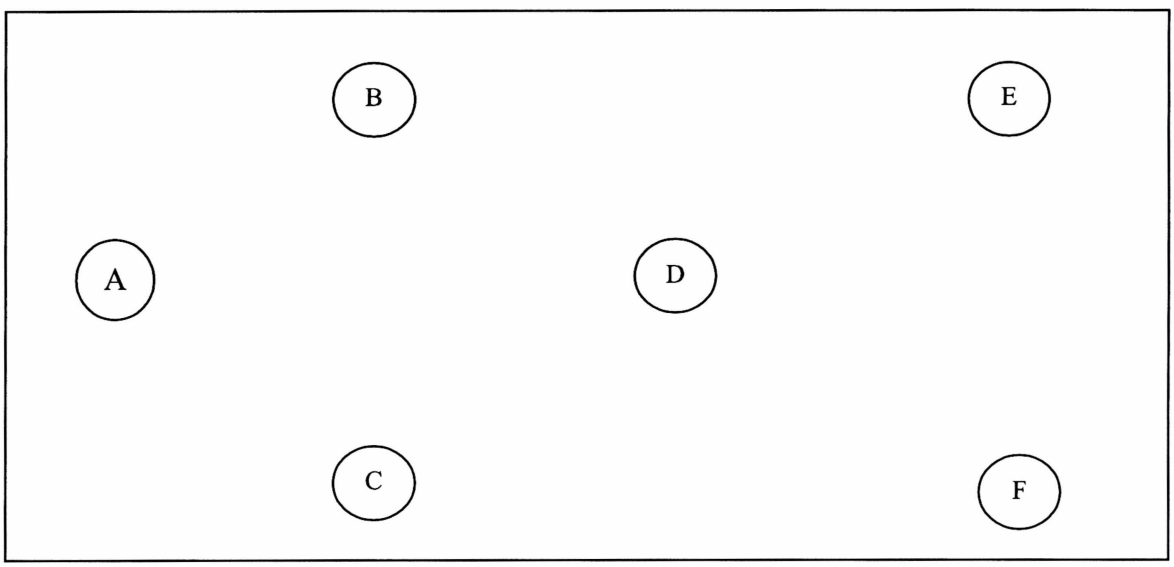
ตอนที่ 4 (30 คะแนน, 40 นาที)

Graph Algorithms

1. บริษัท Graph Corporation มีสาขาที่ทำการกระจายอยู่ใน 6 เมือง (สมมติชื่อเมือง A,B,C,D,E,F) มีระยะทางระหว่างเมืองต่างๆดังนี้ (x หมายถึงไม่มีถนนเชื่อมต่อระหว่างเมืองหรือเป็นเมืองเดียวกัน)

City\City	A	B	C	D	E	F
A	x	10	20	2	x	3
B	10	x	x	10	4	x
C	20	x	x	5	x	x
D	2	10	5	x	2	8
E	x	4	x	2	x	5
F	3	x	x	8	5	x

จงเขียนรูป Undirected-Weighted Graph แทนข้อมูลในตารางข้างต้น (5 คะแนน)



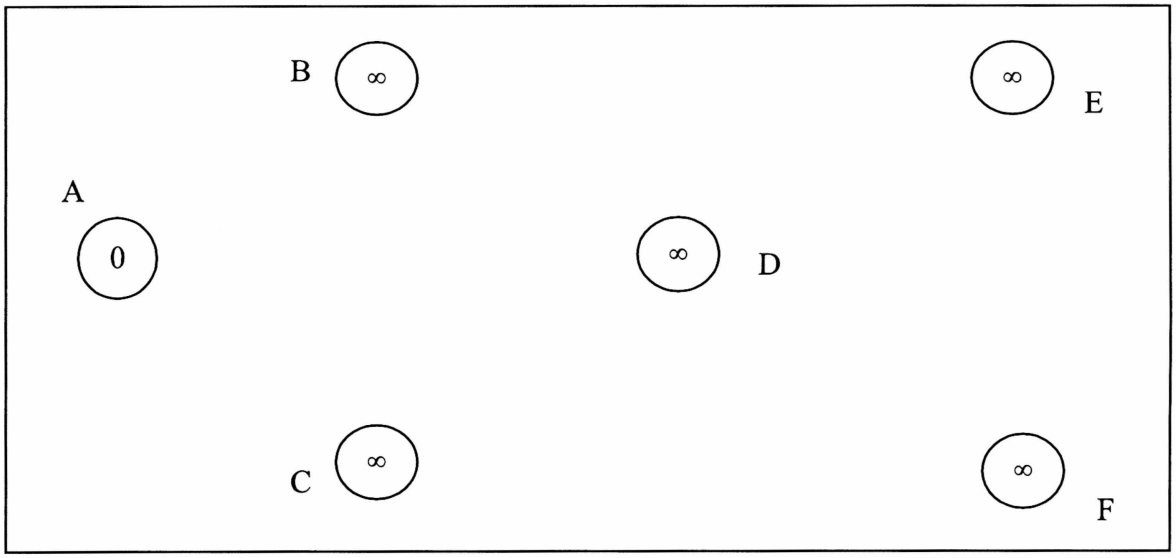
2. บริษัทดังกล่าว มีสำนักงานใหญ่ (Head Quarter) ที่เมือง A ให้หาระยะทางที่สั้นที่สุดในการเดินทางจาก A ไปยังเมืองต่างๆ ทั้งหมด ปัญหานี้คือ Single-Source Shortest Path ที่สามารถแก้ได้ด้วยอัลกอริทึมของดิจค์สตรา (Dijkstra's Algorithm)

2.1) Dijkstra's algorithm จัดเป็นอัลกอริทึมแบบใด (วงกลมข้อที่ถูก) (2 คะแนน)

A. Dynamic Programming B. Greedy C. Divide and Conquer D. Sorting

2.2) จงบอกข้อจำกัดของอัลกอริทึมนี้ (อัลกอริทึมนี้ใช้ไม่ได้กับกราฟแบบใด) (3 คะแนน)

2.3) จงแสดงวิธีคิดโดยใช้ Dijkstra's algorithm หาระยะทางที่สั้นที่สุดจาก A ไปยังปมต่างๆในกราฟ ให้เขียนระยะทางที่คิดได้ไว้ในปม(vertex) โดยสถานะเริ่มต้น ให้ค่าทุกปมเป็น infinity ยกเว้นปม A (จุดเริ่มต้น)เป็นศูนย์ วาดรูปกราฟแสดงการเปลี่ยนแปลงค่าข้อมูลของแต่ละปม และสรุปสุดท้ายได้ระยะที่สั้นที่สุดเท่าไรในแต่ละปมและใช้เส้นใด (20 คะแนน)



ตอนที่ 5 (15 คะแนน, 20 นาที)

State Space Search and Complexity Class

1. ในปริภูมิสถานะ (state space) ที่เป็น binary tree จากวิธีการค้นปริภูมิสถานะ(state space search) ต่อไปนี้ วิธีการใดมีประสิทธิภาพต่ำที่สุดเมื่อคำตอบของปัญหาอยู่ที่โหนดใบลึกสุด เพราะเหตุใด (3 คะแนน)

A. Depth-First Search B. Breadth-First Search

C. Backtracking D. Least-cost Search

2. การค้นหาด้วย Depth-First Search กับ Backtracking เหมือนหรือต่างกันอย่างไร แบบใดมีประสิทธิภาพมากกว่า (5 คะแนน)

3. จากทฤษฎีการจัดกลุ่มความซับซ้อนของปัญหาด้านการคำนวณ จงเขียนชื่อเต็มและอธิบายความหมายโดยสังเขปของกลุ่มปัญหา (problem classes) ต่อไปนี้ *P, NP, NPC, NP-Hard* และวาดแผนภาพเซ็ตแสดงกลุ่มปัญหาดังกล่าว บนสมมติฐานที่ว่า $P \neq NP$ (7 คะแนน)