FACULTY OF ENGINEERING
CHULALONGKORN UNIVERSITY
2110327 Algorithm Design

YEAR III, First Semester, Mid-term Examination, July 23, 2009, Time 8:30 – 11:30

ชื่อ-นามสกุล_____เลขประจำตัว | | | | | | | | | 2 | 1 | CR58_____

หมายเหตุ

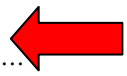
1. ข้อสอบมีทั้งหมด ๑ ข้อในกระดาษคำถามคำตอบจำนวน 2 แผ่น 3 หน้า คะแนนเต็ม 50 คะแนน
2. ไม่อนุญาตให้นำตำราและเครื่องคำนวณต่างๆ ใดๆ เข้าห้องสอบ
3. ควรเขียนตอบด้วยลายมือที่อ่านง่ายและชัดเจน
4. ห้ามการหยิบยืมสิ่งใดๆ ทั้งสิ้น จากผู้สอบอื่นๆ เว้นแต่ผู้คุมสอบจะหยิบยืมให้
5. ห้ามนำส่วนใดส่วนหนึ่งของข้อสอบออกจากห้องสอบ ข้อสอบเป็นทรัพย์สินของราชการซึ่งผู้ลักพาอาจมีโทษทางคดีอาญา
6. ผู้ที่ประสงค์จะออกจากห้องสอบก่อนหมดเวลาสอบ แต่ต้องไม่น้อยกว่า 45 นาที
7. เมื่อหมดเวลาสอบ ผู้เข้าสอบต้องหยุดการเขียนใดๆ ทั้งสิ้น
8. ผู้ที่ปฏิบัติเข้าข่ายทุจริตในการสอบ ตามประกาศคณะวิศวกรรมศาสตร์
   มีโทษ คือ ได้รับ สัญลักษณ์ F ในรายวิชาที่ทุจริต และพักการศึกษาอย่างน้อย 1 ภาคการศึกษา

รับทราบ

ลงชื่อนิสิต (……..……)

หมายเหตุ (เพิ่มเติม)

1. ข้อใดที่ให้ออกแบบอัลกอริทึมนั้น นิสิตสามารถตอบโดยเขียนบรรยายแนวคิดที่ implement ได้ในทางปฏิบัติ หรือจะเขียนเป็นรหัสเทียมประกอบแนวคิดที่นำเสนอด้วยก็ได้ และต้องวิเคราะห์ประสิทธิภาพเชิงเวลาของอัลกอริทึมที่นำเสนอด้วย
2. ต้องแสดงวิธีทำทุกข้อ การเขียนคำตอบเพียงอย่างเดียวจะไม่มีคะแนนให้ (ยกเว้นว่าจะเขียนในคำสั่ง)
3. ให้เขียนตอบในสมุดคำตอบ โดยเขียน**ตอบข้อที่ *k* ไว้ที่หน้าที่ *k* ในสมุดคำตอบ** (k = 1,2,…,9) เขียนไม่พอ ไปต่อที่หน้า 10, …



---

1. (5 คะแนน) กำหนดให้ปัญหา Maximum Sum of Subsequence (MSS) มีนิยามดังต่อไปนี้

- ให้ $A = (a_0,a_1,a_2,\ldots,a_{n-1})$ เป็น sequence ขนาด n ตัว, sum of subsequence ของ A จากตำแหน่ง p ถึง q มีค่าเท่ากับ $\sum_{i=p}^{q} a_i$ และค่า MSS ของ A คือ sum of subsequence ที่มีค่ามากที่สุดของ A

จงเขียนฟังก์ชัน FindMSS (ด้วยภาษา C) ที่ทำงานในเวลา $\Theta(n^2)$ เพื่อหาค่า MSS ของ sequence A ตามโครงฟังก์ชันต่อไปนี้ (ห้ามแก้ไขหัวฟังก์ชัน)

```
int FindMSS(int a[], int n) {    // a คืออาเรย์ขนาด n ช่องที่แทน sequence A
    // insert your code here     // ฟังก์ชันนี้คืนค่าของ MSS ของ sequence A
}
```

2. (7 คะแนน) จงหาคำตอบของ recurrences ข้างล่างนี้ (ในรูปของ Θ) ไม่ต้องแสดงวิธีทำ (ให้ทุกข้อมี $T(1) = O(1)$)

1. $T(n) = 2T(n/3) + 1$
2. $T(n) = 5T(n/4) + n$
3. $T(n) = 7T(n/7) + n$
4. $T(n) = 9T(n/3) + n^2$
5. $T(n) = T(n-1) + 2$
6. $T(n) = T(n-1) + n^{-1}$
7. $T(n) = 2T(n-1) + 3$

3. (3 คะแนน) จงหาคำตอบของ $T(n) = T(0.6n) + T(0.7n) + n^2$, $T(1) = O(1)$ (ตอบในรูปของ Θ) แสดงวิธีทำด้วย

4. (5 คะแนน) จงวิเคราะห์จำนวนการเปรียบเทียบข้อมูล ในบรรทัดที่ 11, 12, และ 16 (แบบ exact analysis ไม่ใช่ asymptotic analysis) เมื่อเรียกฟังก์ชัน maxmin3 ข้างล่างนี้

```
01: void maxmin3(float d[], int n, float *max, float *min) {
02:   int i;
03:   float minV, maxV;
04:   if (n % 2 == 0) {
05:     maxmin(d[0], d[1], max, min);  i = 2;
06:   } else {
07:     *max = *min = d[0];            i = 1;
08:   }
09:   for (; i<= n-2; i+=2) {
10:     maxmin(d[i], d[i+1], &maxV, &minV);
11:     if (maxV > *max) *max = maxV;
12:     if (minV < *min) *min = minV;
13:   }
14: }
15: void maxmin(float a, float b, float *max, float *min) {
16:   if (a > b) {
17:     *max = a; *min = b;
18:   } else {
19:     *max = b; *min = a;
20:   }
21: }
```

5. (7 คะแนน) กำหนดให้ปัญหาการทายกองเป็นดังนี้ ปัญหาทายกองเป็นเกมที่ประกอบด้วยผู้เล่นสองคนคือ ผู้เลือก และ ผู้ทาย เกมจะเริ่มต้นด้วยกองหิน *n* ก้อนที่มีหมายเลข *1, 2, 3, …, n* กำกับ ผู้เลือกจะเลือกหินหนึ่งก้อน และจำหมายเลขหินก้อนนั้นไว้ พร้อมทั้งคืนหินก้อนดังกล่าวลงไปในกอง ขั้นตอนนี้ผู้ทายจะไม่สามารถทราบได้ว่าผู้เลือกได้ทำการเลือกหินหมายเลขใดไว้ หลังจากนั้น ผู้ทาย จะต้องหาว่าหินก้อนใดที่ผู้เลือกได้เลือกไว้ สิ่งที่ผู้ทายสามารถทำได้ คือ ผู้ทายสามารถแบ่งก้อนหินออกมาเป็นกอง ๆ กองละกี่ก้อนก็ได้ แล้วถามผู้เลือกว่า "หินก้อนที่คุณเลือกอยู่ในกองนี้หรือไม่" *<u>โดยมีกติกาว่า ผู้เลือกสามารถโกหกได้เพียงแค่หนึ่งครั้งเท่านั้น</u>* ประสิทธิภาพของผู้ทายในเกมนี้ขึ้นอยู่กับจำนวนครั้งที่มีการถามคำถาม

นายหินเสนออัลกอริทึมแบบ divide and conquer ที่มีกลวิธีในการแบ่งและถามสามแบบข้างล่างนี้ <u>จงเขียน recurrence</u> ที่แทนจำนวนครั้งในการทาย ในแต่ละแบบที่นายหินนำเสนอ

ก) แบ่งหินออกเป็นสองกองเท่า ๆ กัน และทุกครั้งหลังการแบ่งจะถามว่า "มีก้อนหินที่เลือกอยู่ในกองแรกใช่หรือไม่" 3 ครั้ง แล้วตัดกองที่คิดว่าไม่ใช่ออก หลังจากนั้นจึงเริ่มเวียนซ้ำ

ข) แบ่งหินออกเป็นสี่กอง และถามสองครั้ง โดยถามครั้งแรกว่า "ก้อนหินที่เลือกอยู่ในกอง 1 หรือกอง 2 ใช่หรือไม่" ถ้าผู้เลือกตอบว่า "ใช่" ให้ทำเครื่องหมาย ✓ ไว้ที่กอง 1 และ กอง 2 และให้ทำเครื่องหมาย ✗ ไว้ที่กอง 3 และ กอง 4 แต่ถ้าผู้เลือกตอบว่า "ไม่" ให้ทำเครื่องหมายกลับกัน ส่วนคำถามครั้งที่สอง ให้ถามว่า "ก้อนหินที่เลือกอยู่ในกองหนึ่งหรือกองสามใช่หรือไม่" และทำเครื่องหมายเช่นเดียวกันกับคำถามแรก (ดูตัวอย่าง) แล้วให้ตัดกองที่ถูกทำเครื่องหมาย ✗ สองครั้งทิ้งไป หลังจากนั้นจึงเริ่มเวียนซ้ำ

กอง 1 ✓✓	กอง 2 ✓✗
กอง 2 ✓✗	กอง 4 ✗✗

ตัวอย่าง กรณี คำถามข้อแรกตอบว่า "ใช่" และข้อสองตอบว่า "ใช่" ให้ตัดกอง 4 ทิ้งไป

ค) แบ่งกองเป็นสี่กอง ถามคำถามเช่นเดียวกับ ข) ตัดกองที่มีเครื่องหมาย ✗✗ ออก แล้วเก็บกองที่มีเครื่องหมาย ✓✓ ไว้เรียกว่า กอง a โดยรวมกองที่มีเครื่องหมาย ✓✗ เข้าด้วยกันเรียกว่า b ทำการแบ่งครึ่งกอง a และ b อย่างละครึ่งจนได้สี่กอง แล้วจึงเริ่มเวียนซ้ำ

6. (8 คะแนน) เราสามารถมองการทำงานของ merge sort แบบ bottom-up ได้ ด้วยการแบ่งข้อมูลออกเป็น n ชุด ชุดละ 1 ตัว
   - เพื่อนำข้อมูลชุดละตัวนี้มา merge ทีละ 2 ชุด จะได้ข้อมูล $\lceil n/2 \rceil$ ชุด ชุดละไม่เกิน 2 ตัว
   - จากนั้น merge ข้อมูลทีละ 2 ชุด ได้ข้อมูล $\lceil n/4 \rceil$ ชุด ชุดละไม่เกิน 4 ตัว
   - แล้ว merge ต่อทีละ 2 ชุด ได้ $\lceil n/8 \rceil$ ชุด ชุดละไม่เกิน 8 ตัว, …
   - กระทำเช่นนี้ไปจน merge แล้วได้ข้อมูลเพียงชุดเดียว จะได้ข้อมูลในชุดนี้เรียงลำดับ

   ก) จงเขียนรหัสเทียมแสดงอัลกอริทึมของการทำ mergesort แบบ bottom-up

   ข) จงยกตัวอย่างอาเรย์ขนาด 8 ช่องของจำนวนที่มีค่าต่างกันหมด ที่ทำให้การใช้ mergesort แบบ bottom-up มีการเปรียบเทียบข้อมูลในอาเรย์เป็นจำนวนครั้งน้อยสุด

   ค) จงยกตัวอย่างอาเรย์ขนาด 8 ช่องของจำนวนที่มีค่าต่างกันหมด ที่ทำให้การใช้ mergesort แบบ bottom-up มีการเปรียบเทียบข้อมูลในอาเรย์เป็นจำนวนครั้งมากสุด

   ง) การแบ่งเป็นชุด ๆ ด้วยวิธีที่นำเสนอมาไม่ได้พิจารณาลักษณะของข้อมูลเลย หากคิดสักเล็กน้อยก่อนแบ่ง แบ่งให้ฉลาดหน่อย น่าจะทำให้เรียงลำดับได้ดีขึ้น เช่น กรณีที่ข้อมูลในอาเรย์มีลักษณะที่เรียงกันจากมากไปน้อย หรือน้อยไปมาก ต่อกันยาว ๆ เช่น

   [0 , 7 , 11 , 14 , 13 , 12 , 7 , 3 , 3 , 1]

   จงออกแบบวิธีการแบ่งข้อมูลแบบใหม่ให้กับอัลกอริทึม merge sort แบบ bottom-up เพื่อให้สามารถทำงานกับลักษณะของข้อมูลดังกล่าว ที่มีประสิทธิภาพสูงขึ้น และจงวิเคราะห์ประสิทธิภาพเชิงเวลาของอัลกอริทึมนี้ด้วย

7. (5 คะแนน) ให้ S เป็น set ของจำนวนเต็ม n ตัว $S = (a_1,a_2,\ldots,a_n)$ และ ให้ t เป็นจำนวนเต็มค่าหนึ่ง เราอยากทราบว่า มี subset ของ S ที่ผลรวมของสมาชิกทุกตัวใน subset นั้นมีค่าเท่ากับ t หรือไม่ (เช่น ……) จงออกแบบอัลกอริทึมที่ใช้เวลา $O(nt)$ ในการตอบคำถามดังกล่าว โดยให้ออกแบบเฉพาะ recurrence ของปัญหาดังกล่าว พร้อมคำอธิบายและนิยามของ recurrence นั้น โดยไม่ต้องเขียนอัลกอริทึมหรือรหัสเทียมใด ๆ

8. (5 คะแนน) กำหนดให้ฟังก์ชัน $f(x)$ มีนิยามดังนี้

$$f(x) = \begin{cases} x/2, & x \text{ is even} \\ 3x+1, & x \text{ is odd} \end{cases}$$

   และกำหนดให้ Hailstone sequence ของตัวเลข x คือ sequence ดังนี้ $(x, f(x), f(f(x)), f(f(f(x))), \ldots)$ โดยที่ sequence นี้จะหยุดที่ตัวเลข 1 ตัวอย่างเช่น Hailstone sequence ของ 5 คือ (5,16,8,4,2,1) กำหนดให้ความยาวของ Hailstone sequence คือจำนวนของสมาชิกใน sequence นั้น (ความยาวของ Hailstone sequence ของตัวเลข 5 คือ 6)

   จงออกแบบอัลกอริทึมโดยใช้เทคนิค Memoization ในการคำนวณหาความยาวของ Hailstone sequence ของจำนวนเต็มตั้งแต่ 1 ถึง 9,999

9. (5 คะแนน) กำหนดให้ `A[1..n]` เป็นอาเรย์ขนาด n ช่องของจำนวนเต็มที่มีค่าต่างกันหมด และ ข้อมูลในอาเรย์นั้นเรียงจากน้อยไปมากเรียบร้อยแล้ว จงออกแบบอัลกอริทึมประเภท divide-and-conquer ที่ใช้เวลา $O((\log n)^2)$ เพื่อตรวจสอบว่าในอาเรย์ดังกล่าว มีช่องที่ `A[i] = i` หรือไม่