FACULTY OF ENGINEERING
CHULALONGKORN UNIVERSITY
2110327 ALGORITHM DESIGN
Year II, Second Semester, Midterm Examination, March 10, 2023 13:00-16:00

ชื่อ-นามสกุล.....Jomnoi2.....เลขประจำตัว..... เลขที่ใน CR58.....

หมายเหตุ

1. ข้อสอบมีทั้งหมด 11 ข้อ ในกระดาษคำถามคำตอบ 7 หน้า
2. **ไม่อนุญาตให้นำตำราและเอกสารใดๆ เข้าในห้องสอบ**
3. **ไม่อนุญาตให้ใช้เครื่องคำนวณใดๆ**
4. ห้ามการหยิบยืมสิ่งใดๆ ทั้งสิ้น จากผู้สอบอื่นๆ เว้นแต่เจ้าหน้าที่ควบคุมการสอบจะหยิบยืมให้
5. ห้ามนำส่วนใดส่วนหนึ่งของข้อสอบและสมุดคำตอบออกจากห้องสอบ
6. ผู้เข้าสอบสามารถออกจากห้องสอบได้ หลังจากผ่านการสอบไปแล้ว 45 นาที
7. เมื่อหมดเวลาสอบ ผู้เข้าสอบต้องหยุดการเขียนใดๆ ทั้งสิ้น
8. **นิสิตกระทำผิดเกี่ยวกับการสอบ ตามข้อบังคับจุฬาลงกรณ์มหาวิทยาลัย มีโทษ คือ พ้นสภาพการเป็นนิสิต หรือ ได้รับสัญลักษณ์ F ในรายวิชาที่กระทำผิด และอาจพิจารณาให้ถอนรายวิชาอื่นทั้งหมดที่ลงทะเบียนไว้ในภาคการศึกษานี้**

*** ร่วมรณรงค์การไม่กระทำผิดและไม่ทุจริตการสอบที่คณะวิศวกรรมศาสตร์ ***

ข้าพเจ้ายอมรับในข้อกำหนดที่กล่าวมานี้ ข้าพเจ้าเป็นผู้ทำข้อสอบนี้ด้วยตนเองโดยมิได้รับการช่วยเหลือ หรือให้ความช่วยเหลือ ในการทำข้อสอบนี้

ลงชื่อนิสิต.....

วันที่.....

- ใช้ดินสอทำข้อสอบได้
- ให้เขียนเลขที่ในเซ็นชื่อเข้าสอบ และ รหัสนิสิต ในทุกหน้าของข้อสอบ **<u>หน้าใดไม่ได้เขียนจะไม่ได้รับการตรวจ</u>**
- หากพื้นที่สำหรับเขียนคำตอบไม่เพียงพอ ให้เขียนไว้ด้านหลังของหน้านั้น **<u>ห้ามเขียนข้ามไปหน้าอื่น</u>** และให้ระบุไว้ในพื้นที่สำหรับเขียนคำตอบว่า “มีต่อด้านหลัง”
- นิสิตสามารถใช้ข้อมูลสำหรับ Master Method ในรูปข้างล่างนี้ช่วยในการตอบคำตอบได้

$$t(n) = at(n/b) + \Theta(n^d) \qquad a \geq 1, b > 1$$

$$t(n) = \begin{cases} \Theta(n^c) & \text{if } n^d < n^c \\ \Theta(n^c \log n) & \text{if } n^d = n^c \\ \Theta(n^d) & \text{if } n^d > n^c \end{cases} \qquad c = \log_b a$$

$$t(n) = at(n/b) + f(n)$$

$$t(n) = \begin{cases} \Theta(n^c) & \text{if } f(n) = O(n^{c-\varepsilon}) \ \textbf{1} \quad \varepsilon > 0 \\ \Theta(n^c \log n) & \text{if } f(n) = \Theta(n^c) \ \textbf{2} \\ \Theta(f(n)) & \text{if } f(n) = \Omega(n^{c+\varepsilon}) \ \textbf{3} \\ & af(n/b) \leq kf(n), k < 1, n \geq n_0 \end{cases}$$

1. (4 คะแนน) จงระบุ Time Complexity ของ Recurrence Relation ต่อไปนี้โดยให้ตอบเป็น Asymptotic Notation และให้ถือว่า T(1) = 1 ในทุกข้อ
   1.1. $T(n) = 4T(n/3) + O(4n^2)$ เวลาการทำงานคือ $\Theta(4n^2)$
   1.2. $T(n) = 2T(n/2) + O(2n)$ เวลาการทำงานคือ $\Theta(n \log n)$
   1.3. $T(n) = T(n/2) + O(3^n)$ เวลาการทำงานคือ $\Theta(3^n)$
   1.4. $T(n) = T(n/4) + O(3)$ เวลาการทำงานคือ $\Theta(\log n)$
2. (3 คะแนน) จงเรียงลำดับของ Time Complexity ในการทำงานใน Worst Case ของอัลกอริทึมดังต่อไปนี้จากเร็วไปช้า
   A) Selection Sort ของรายการของตัวเลข n ตัว
   B) หาตัวหารร่วมมากของตัวเลขจำนวนเต็มสองตัวที่มีค่าไม่เกิน n
   C) พิมพ์ Permutation ทั้งหมดของตัวเลข n ตัว
   D) Linear Search ของรายการของตัวเลข n ตัว
   E) Merge Sort ของรายการของตัวเลข n ตัว
   F) พิมพ์เลขฐานสองทั้งหมดของ n บิท

   B) D) E) A) F) C) (ให้ตอบโดยเขียนตัวอักษร 6 ตัวตามลำดับ เช่น A B C D E F)
3. (10 คะแนน) ในข้อย่อยต่อไปนี้ให้ตอบโดยเลือกคำตอบที่ถูกต้องที่สุด แต่ละข้อย่อยมีคะแนน 1 คะแนน หากไม่ตอบในข้อย่อยใด จะได้คะแนน 0 แต่ถ้าหากตอบผิดในข้อย่อยใด จะได้คะแนน -0.5 ต่อข้อ อย่างไรก็ตาม คะแนนที่น้อยที่สุดที่เป็นไปได้ของข้อนี้คือ 0 (กล่าวคือ ถึงแม้จะตอบผิดจนได้คะแนนรวมติดลบ ก็จะถือว่าได้คะแนนเป็น 0) ให้ตอบโดยการเขียนตัวเลือกที่ต้องการลงในตารางข้างล่างนี้เท่านั้น

ข้อ 3.1	ข้อ 3.2	ข้อ 3.3	ข้อ 3.4	ข้อ 3.5	ข้อ 3.6	ข้อ 3.7	ข้อ 3.8	ข้อ 3.9	ข้อ 3.10
ง.	ค.	ข.	ง.	ก.	ง.	ก.	ง.	ค.	ข.

   3.1. อัลกอริทึมในข้อใดโดยทั่วไปแล้วไม่จัดเป็นประเภท Divide and Conquer
      ก. Merge Sort
      ข. Quick Sort
      ค. Binary Search
      ง. Linear Search
   3.2. ข้อความในข้อใดถูกต้องที่สุด
      ก. Dynamic Programming ลดเวลาในการคำนวณโดยการประหยัดหน่วยความจำ
      ข. Divide and Conquer ลดเวลาการคำนวณโดยการจำคำตอบของปัญหา
      ค. Dynamic Programming ลดเวลาในการคำนวณได้โดยการไม่แก้ปัญหาย่อยซ้ำ
      ง. Divide and Conquer ลดเวลาในการคำนวณโดยแก้ปัญหาย่อยให้เร็ว
   3.3. ข้อใดถูกต้องที่สุดเกี่ยวกับ Brute Force
      ก. เป็นการใช้คอมพิวเตอร์ให้เต็มแรง
      ข. เป็นการลองพิจารณาคำตอบที่เป็นไปได้ทุกวิธี
      ค. เป็นการแก้ปัญหาโดยการแบ่งปัญหาเป็นปัญหาย่อยๆ
      ง. เป็นการลดเวลาในการคำนวณด้วยการใช้แรงอย่างมาก
   3.4. Keyword ในข้อใดเกี่ยวข้องกับ Master Method น้อยที่สุด
      ก. Divide and Conquer
      ข. Time Complexity
      ค. Recurrence Relation
      ง. Dynamic Programming
   3.5. Keyword ในข้อใดเกี่ยวข้องกับ Dynamic Programming น้อยที่สุด
      ก. Memorization
      ข. Memoization
      ค. Top Down
      ง. Bottom Up

3.6. Recurrence Relation f(n) = 3f(n/4)+5 แบ่งปัญหาเป็นกี่ปัญหาย่อยและแต่ละส่วนขนาดเท่าไหร่
- ก. 4 ปัญหาย่อย แต่ละส่วนขนาด 3
- ข. 4 ปัญหาย่อย แต่ละส่วนขนาด n/3
- ค. 3 ปัญหาย่อย แต่ละส่วนขนาด 4
- ง. 3 ปัญหาย่อย แต่ละส่วนขนาด n/4

3.7. ข้อใดเป็นสิ่งที่ Master Method ประมาณการ
- ก. เวลาที่ใช้ในการแก้ปัญหา
- ข. จำนวนคำสั่งที่ใช้ในการแก้ปัญหา
- ค. ปริมาณหน่วยความจำที่ใช้ในการแก้ปัญหา
- ง. จำนวณบรรทัดของอัลกอริทึมที่ใช้ในการแก้ปัญหา

3.8. ข้อใดถูกต้องเกี่ยวกับอัลกอริทึม Quick Select เมื่อใช้กับข้อมูล n ตัว
- ก. สามารถใช้คำนวณหาค่าที่น้อยที่สุดลำดับที่ k ได้ด้วย
- ข. มักมีการเรียก Recursive Call ในการแก้ปัญหา
- ค. มีวิธีที่สามารถรับประกันว่าเวลาเป็น O(n) ได้
- ง. ถูกทุกข้อ

3.9. หากใช้ Huffman Coding ในการจัดเก็บรายการอักขระ a,b,c,d,e ที่มีความถี่ในการปรากฏ 100,500,200,250,200 ครั้งตามลำดับจะต้องใช้กี่บิท
- ก. 1750
- ข. 1950
- ค. 2750
- ง. 2950

3.10. การคูณเมทริกซ์ขนาด 4x3, 3x5, 5x1, 1x2 ด้วยวิธี Matrix Chain Multiplication ที่เรียนในวิชานี้ใช้การคูณทั้งหมดกี่ครั้ง
- ก. 69
- ข. 35
- ค. 88
- ง. 37

4. (5 คะแนน) จากโจทย์ข้อ Pressure Station ใน grader (หากจำไม่ได้ คำอธิบายโจทย์จะอยู่ในย่อหน้าสุดท้าย) สมชายได้ออกแบบ Recurrence Relation สำหรับแก้ปัญหาดังกล่าวเป็นดังนี้ ให้ B(x) คือคำตอบของปัญหา Pressure Station เมื่อ input ของปัญหาคือ p[1..x] สมชายได้ออกแบบให้ B มีค่าเป็นดังนี้

$$B(x) = \begin{cases} \min(p[1..x]) & ; \quad x \le k+1 \\ \min\limits_{1 \le i \le k} (p[x-i] + B(x-i-1)) & ; \quad x > k+1 \end{cases}$$

จงตอบคำถามต่อไปนี้

4.1. Recurrence Relation ข้างบนนี้ผิดอยู่ จงยกตัวอย่างค่าของ p ที่ทำให้ recurrence นี้ผิด โดยกำหนดให้ k = 2 และ n = 5 และ 1 <= p[i] <= 5 โดยให้ระบุ p และค่าของ B(5) และค่าที่เป็นคำตอบที่ควรเป็น
- p = [100 100 1 100 100]
- B(5) = 101
- คำตอบที่ควรเป็นคือ 1

4.2. จงอธิบายโดยสังเขปว่า Recurrence Relation ข้างบนนี้ผิดอย่างไร

- B(x) ไม่ได้พิจารณากรณีที่วางป้อมที่ตำแหน่ง x
- ถ้าวางป้อมที่ตำแหน่ง x-i แล้ว คำตอบที่ดีที่สุดอาจจะไม่ได้เกิดจากปัญหาย่อย B[x-i-1] เสมอไป

(คำอธิบายโจทย์ข้อ Pressure Station) กำหนดให้ p[1..n] เป็นอาเรย์ขนาด n ช่อง และมีค่า k <= n อยู่ เราต้องเลือกช่องมาบางช่องจาก n ช่องนี้ โดยเราจะมี “ค่าใช้จ่ายรวม” เป็นค่า p[i] ของช่องที่เลือกทุกช่องรวมกัน (เช่น เลือกช่อง 1,2,5 จะมีค่าใช้จ่ายเป็น p[1] + p[2] + p[5]) และเมื่อเราเลือกช่อง i ใด ๆ แล้วจะถือว่าเรา “ครอบคลุม” ช่องทุกช่องในช่วง i-k ถึง i+k เราต้องการเลือกช่องให้ได้ค่าใช้จ่ายรวมน้อยที่สุดโดยทำให้ช่องทุกช่องตั้งแต่ช่อง 1 ถึง n ถูกครอบคลุมทั้งหมด

5. (6 คะแนน) จาก Recurrence Relation ที่ให้ต่อไปนี้ จงเขียนส่วนของโปรแกรมแบบ Dynamic Programming โดยใช้ตัวแปรตามที่กำหนดให้ในแต่ละข้อ เพื่อทำการเติมค่าในตาราง โดยใช้เทคนิค Dynamic Programming แบบ Bottom-Up

5.1. กำหนดให้ n p q และ K เป็นจำนวนเต็มบวก โดยที่ p < q < K และ Recurrence Relation ของเราคือ

$$B(n) = \begin{cases} 0 & ; \quad 0 < n \leq K \\ B(n-p) + B(n-q) + 1 & ; \quad n > K \end{cases}$$

กำหนดให้ n p q และ K คือข้อมูลนำเข้า และให้ B[1..n] คือตารางสำหรับเก็บข้อมูล เราต้องการคำนวณ B[n]

```
for (int i=1; i<=k; i++) B[i] = 0;
for (int i = K+1 <= i <= n; i++) B[i] = B[i-p]+ B[i-q] + 1;
```

5.2. กำหนดให้ n เป็นจำนวนเต็มบวก E เป็น set ของคู่อันดับ (a,b) โดยที่ 1 <= a,b <= k และ A[1..k] เป็นอาเรย์ขนาด k ช่องที่เก็บจำนวนเต็ม Recurrence Relation ของเราคือ

$$B(p,q) = \begin{cases} A[p] & ; \quad q = 0 \\ \min\limits_{(s,p)\in E} (2 * B(p, \lfloor q/2 \rfloor) + A[s]) & ; \quad q > 0 \end{cases}$$

กำหนดให้ n k และ E คือข้อมูลนำเข้า และให้ B[1..k][0..n] คือตารางสำหรับเก็บข้อมูล เราต้องการคำนวณ B[1..k][n]

```
for (int p = 1; p <= k; p++) {
    B[p][0] = A[p];
    for (int q=1; q <= n; q++) {
        B[p][q] = 1e9 + 7;
        for (auto &s : E[p]) {  // สมมติว่า E เป็น adjacency list :)
            B[p][q] = min(B[p][q], 2 * B[p][q/2] + A[s]);
        }
    }
}
```

6. (7 คะแนน) จาก Recurrence Relation และค่าของฟังก์ชันที่ต้องการทราบที่กำหนดให้ต่อไปนี้ ให้ระบุว่ามีปัญหาย่อยซ้ำกันหรือไม่ และถ้ามี จงยกตัวอย่างอย่างน้อย 1 คู่ปัญหาย่อยที่ซ้ำกัน

6.1. $$DP(a,b) = \begin{cases} a+b & ; \quad a = b \\ \min\limits_{a \leq i < b} DP(a,i) + DP(i+1,b) & ; \quad a < b \end{cases}$$

- ต้องการทราบ DP(1,4)
- มีปัญหาย่อยซ้ำกันหรือไม่ ☑ มี ☐ ไม่มี
- ถ้ามี จงระบุปัญหาย่อยที่เกิดขึ้นมากกว่า 1 ครั้งเป็นจำนวน 1 ปัญหาย่อย

DP(1, 2)

6.2. กำหนดให้ $m = \lfloor (r-l)/4 \rfloor$ และให้ถือว่า X เป็นอาเรย์ที่มีการกำหนดค่ามาแล้ว

$$DP(l,r) = \begin{cases} \Sigma_{i=l}^{r}(X[i]) & ; \quad r-l < 4 \\ DP(l, l+2m) + DP(l+m, l+3m) + DP(l+2m, R) & ; \quad r-l \geq 4 \end{cases}$$

- ต้องการทราบ DP(0,4)
- มีปัญหาย่อยซ้ำกันหรือไม่ ☐ มี ☑ ไม่มี
- ถ้ามี จงระบุปัญหาย่อยที่เกิดขึ้นมากกว่า 1 ครั้งเป็นจำนวน 1 ปัญหาย่อย

6.3. (ข้อนี้ recurrence relation เหมือนข้อที่แล้ว แต่ค่าที่ต้องการทราบแตกต่างกัน)

- ต้องการทราบ DP(0,8)
- มีปัญหาย่อยซ้ำกันหรือไม่ ☑ มี ☐ ไม่มี
- ถ้ามี จงระบุปัญหาย่อยที่เกิดขึ้นมากกว่า 1 ครั้งเป็นจำนวน 1 ปัญหาย่อย

DP(2,4)

6.4. กำหนดให้ n เป็นจำนวนเต็ม และ X เป็นอาเรย์ โดยทั้ง X และ n มีการกำหนดค่ามาแล้ว และให้ L = i*2+1 และ R คือ L+1

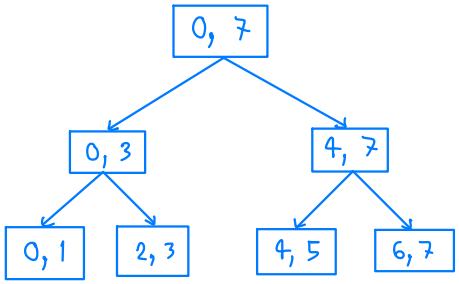
$$DP(i,b) = \begin{cases} 0 & ; \quad i > n \text{ และ } b = 0 \\ X[i] & ; \quad i > n \text{ และ } b = 1 \\ \min\big(DP(L,0),DP(L,1)\big) + \min(DP(R,0),DP(R,1)) & ; \quad i \le n \text{ และ } b = 0 \\ DP(L,0) + DP(R,0) & ; \quad i \le n \text{ และ } b = 1 \end{cases}$$

- ต้องการทราบ DP(0,0) และให้ n = 14
- มีปัญหาย่อยซ้ำกันหรือไม่ ☑ มี ☐ ไม่มี
- ถ้ามี จงระบุปัญหาย่อยที่เกิดขึ้นมากกว่า 1 ครั้งเป็นจำนวน 1 ปัญหาย่อย

DP(3,0)

7. (3 คะแนน) จากส่วนของโปรแกรมด้านขวานี้ จงวาด Recursion Tree เมื่อเราเรียกฟังก์ชัน recur({1,2,3,4,10,11,12,13},0,7) โดยในแต่ละปมให้ระบุเฉพาะค่า l, r ก็พอ

```
int recur(vector<int> &a,int l,int r) {
  if (l + 1 == r) return max(a[l],a[r])
  if (l == r) return a[l]
  int m = (l + r) / 2
  if (m is even) {
    return max(recur(a,l,l+1),recur(a,l+2,r))
  } else {
    return max(recur(a,l,m),recur(a,m+1,r))
  }
}
```



8. (6 คะแนน) ในแต่ละข้อต่อไปนี้ ให้สมมติว่าเรากำลังพยายามแก้ปัญหาด้วยวิธีการ Brute Force จงระบุ เซ็ตของ candidate solution ที่เหมาะสมในการแก้ปัญหา พร้อมด้วยขนาดของ set ดังกล่าว

8.1. (ตัวอย่าง ปัญหา MCS) มีอาเรย์ A[1..n] เราต้องการหาช่วงติดกันที่ผลรวมของสมาชิกในช่วงมีค่ามากที่สุด

- เซ็ตของ candidate solution: { (i,j) | 1 <= i <= j <= n }
- ขนาดของเซ็ตดังกล่าว: n(n+1)/2

8.2. มีตารางขนาดกว้าง w ช่องและสูง h ช่อง โดยที่แต่ละช่องในตารางเป็นสีขาวหรือสีดำเท่านั้น เราต้องการตัดตารางนี้ (โดยตัดตามรอยต่อของช่องต่าง ๆ เท่านั้น ห้ามตัดผ่าช่อง และเมื่อตัดแล้วต้องทิ้ง) ให้ตารางมีพื้นที่ใหญ่ที่สุดที่ทุกช่องที่เหลืออยู่มีสีดำเท่านั้น

- เซ็ตของ candidate solution: How to cut the grid?!
- ขนาดของเซ็ตดังกล่าว:

8.3. มีตารางขนาด n X n ช่อง มีหุ่นยนต์ตัวหนึ่งอยู่ ณ ช่องบนสุดซ้ายสุด (พิกัด (1,1)) หุ่นยนต์ตัวนี้สามารถเดินไปในทิศ “ไปช่องล่างที่ติดกับช่องปัจจุบัน” หรือ “ไปช่องขวาที่ติดกับช่องปัจจุบัน” เท่านั้น หุ่นยนต์ต้องการเดินทางไปยังช่องล่างสุดขวาสุด (พิกัด (n,n)) โดยแต่ละช่องมีตัวเลขกำกับอยู่ เมื่อหุ่นยนต์ผ่านช่องไหนก็จะได้คะแนนเท่ากับค่าของช่องนั้น อยากทราบวิธีการเดินที่ทำให้หุ่นยนต์ได้คะแนนรวมมากที่สุด

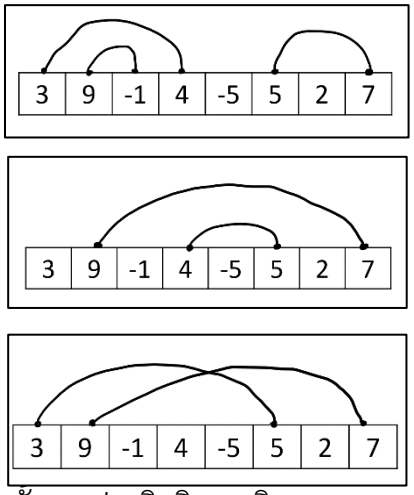
- เซ็ตของ candidate solution: { ลำดับการเดิน โดยที่ต้อง "ลง" n-1 ครั้ง และ "ขวา" n-1 ครั้ง }
- ขนาดของเซ็ตดังกล่าว: $\binom{2n-2}{n-1}$

8.4. มีตารางหมากรุกขนาด n x n ช่องอยู่ ม้าหมากรุกเป็นตัวหมากรุกแบบหนึ่งซึ่งมีกฎการเดิน คือ หากม้าตัวนี้อยู่ ณ ช่อง (x, y) ใด ๆ การเดิน 1 ครั้งของ จะสามารถเดินจากช่องดังกล่าวไปยังช่อง $(x + a, y + b)$ ได้เท่านั้น โดยที่ a และ b ต้องมีคุณสมบัติคือ $|a| + |b| = 3$ และ $|a| \in \{1,2\}$ และ $|b| \in \{1,2\}$ เราอยากทราบว่ามีรูปแบบการเดินกี่รูปแบบที่ทำให้ม้าสามารถเดินต่อกันไปจากช่องหนึ่งในตารางหมากรุกไปยังแต่ละช่องในตารางหมากรุกครบทุกช่องโดยไม่ซ้ำกันเลย

- เซ็ตของ candidate solution: { พิกัดที่ม้าเลือกเดิน ความยาว n^2 ตำแหน่ง แต่ละตำแหน่งเลือกได้อย่างมาก 8 ทิศทาง }
- ขนาดของเซ็ตดังกล่าว: $\leq 8^{(n^2-1)}$?!

- **สำหรับข้อที่ 9 เป็นต้นไป เป็นการออกแบบอัลกอริทึม ในแต่ละข้อสามารถตอบโดยการอธิบายอัลกอริทึม โดยใช้รหัสเทียม (Pseudocode) หรือ programming language ภาษาใดที่เคยเรียนมาก็ได้ และต้องวิเคราะห์ประสิทธิภาพในการทำงานของอัลกอริทึมด้วย**
- **คะแนนที่ได้จะแปรตามประสิทธิภาพในการทำงาน**
- **ในทุกข้อให้วิเคราะห์ประสิทธิภาพในการทำงาน โดยตอบเป็นสัญกรเชิงเส้นกำกับด้วย**

9. (10 คะแนน) มีกระดาษอยู่หนึ่งใบ ด้านล่างสุดของกระดาษใบนี้มีตารางขนาด 1 แถว n ช่องอยู่ โดยที่แต่ช่องมีตัวเลขกำกับอยู่ ให้ A[i] คือตัวเลขที่กำกับอยู่บนช่องลำดับที่ i เราจะต้องลากเส้นในกระดาษใบนี้เพื่อจับคู่ช่องต่าง ๆ โดยมีกฎคือ 1) เส้นเชื่อมต้องเริ่มและจบจากตรงกลางของขอบด้านบนของช่องเท่านั้น 2) ช่องแต่ละช่องสามารถจับคู่กับช่องอื่นได้มากสุด 1 ช่องเท่านั้น 3) เส้นที่ลากเชื่อมกล่องนั้นจะต้องไม่ตัดกันเลย



รูปด้านขวามือนี้เป็นตัวอย่างการลากเส้น โดยสองรูปบนเป็นการลากเส้นเชื่อมที่ถูกต้องตามกฎ แต่รูปล่างสุดไม่ถูกกฎเนื่องจากมีเส้นเชื่อมที่ตัดกัน

โดยเมื่อเราลากเส้นเชื่อมระหว่างช่องสองช่องใด ๆ เราจะได้คะแนนเท่ากับค่าในช่องทั้งสองช่องนั้นคูณกัน เราต้องการลากเส้นเชื่อมให้ได้คะแนนรวมจากการลากเส้นทั้งหมดให้มากที่สุด จงออกแบบอัลกอริทึมสำหรับหาว่าคะแนนรวมมากที่สุดที่เราสามารถทำได้คือเท่าไร โดยกำหนดให้ข้อมูลนำเข้าคือ A[1..n] และ n พร้อมทั้งระบุประสิทธิภาพเชิงเวลา

// Already done in grader :) // a65_q2b_arch_match #536844

// Time complexity : $O(n^3)$

10. (10 คะแนน) ปัญหาข้อนี้เราจะพิจารณาถึงสายอักขระ (string) และคุณสมบัติบางอย่างของสายอักขระ ขอนิยามให้ สายอักขระ S ความยาว n คืออาเรย์ของตัวอักษร n ตัวเขียนแทนได้ด้วย “s[0]s[1]s[2]...s[n-1]” เราจะเรียกสายอักขระ S ว่ามีคุณสมบัติเป็น Palindrome ก็ต่อเมื่อสายอักขระนั้นมีค่าเท่าเดิมเมื่อกลับหลังไปหน้า กล่าวคือ “s[0]s[1]s[2]...s[n-1]” จะมีค่าเท่ากับ “s[n-1]s[n-2]...s[1]s[0]” ตัวอย่างเช่น “a” และ “aba” และ “caac” และ “bhahb” ต่างก็เป็น Palindrome ในขณะที่ “ba” และ “abc” และ “bba” และ “ccdc” ไม่เป็น Palindrome โดยเราจะถือว่าสายอักขระว่าง (“”) นั้นนับเป็น Palindrome ด้วยเช่นกัน

ในข้อนี้ กำหนดให้มีสายอักขระ S อยู่ เราต้องการแทรกอักขระน้อยที่สุดเท่าที่จำเป็นเข้าไปใน S ในตำแหน่ง ต่าง ๆ เพื่อทำให้ S เป็น Palindrome ตัวอย่างเช่น

หาก S มีค่าเป็น “abcb” เราสามารถแทรก “a” ไปในตำแหน่งท้ายสุดเพื่อให้ S เป็น “abcba”
หาก S มีค่าเป็น “acbb” เราสามารถแทรก “ca” ไปในตำแหน่งท้ายสุดเพื่อให้ S เป็น “acbbca”
หาก S มีค่าเป็น “baefb” เราสามารถแทรก “f” และ “a” ในตำแหน่งต่าง ๆ S เป็น “bfaeafb”
ให้สังเกตว่าอักขระที่แทรกจะทำการขีดเส้นใต้ไว้ และหลังจากการแทรกนั้น S จะเป็น Palindrome
จงตอบคำถามต่อไปนี้

10.1. จาก S ที่กำหนดให้ต่อไปนี้ จงระบุว่าเราต้องแทรกอักขระเป็นจำนวนน้อยที่สุดกี่ตัวเพื่อให้ S กลายเป็น Palindrome

- S = “adba” ต้องแทรกอักขระ 1 ตัว
- S = “acc” ต้องแทรกอักขระ 1 ตัว
- S = “abcdba” ต้องแทรกอักขระ 1 ตัว
- S = “abdegjkgdba” ต้องแทรกอักขระ 2 ตัว
- S = “fabbfbbak” ต้องแทรกอักขระ 2 ตัว

10.2. กำหนดให้ C(L,R) โดยที่ 0 <= L <= R < n เป็น Recurrence Relation ที่คืนค่าจำนวนอักขระน้อยสุดที่ต้องแทรกเข้าไปเพื่อให้สายอักขระ “S[L] S[L+1] ... S[R-1] S[R]” เป็น palindrome คำตอบของปัญหานี้คือ C(0,n-1) จงออกแบบ Recurrence Relation นี้

$$C(L,R) = \begin{cases} 0 & ; \ L \geq R \\ \min(C(L+1,R)+1,\ C(L,R-1)+1) & \\ C(L+1,R-1) & ; \ S[L] == S[R] \end{cases}$$

10.3. จงเขียนโปรแกรมแบบ Bottom Up เพื่อคำนวณค่า C(0,n-1) เมื่อกำหนดให้ S และ n คือข้อมูลนำเข้า (ข้อนี้จะได้คะแนนก็ต่อเมื่อข้อก่อนหน้าถูกต้องเท่านั้น) พร้อมทั้งระบุประสิทธิภาพเชิงเวลา

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    cin.tie(nullptr)->sync_with_stdio(false);
    int n;
    string S;
    cin >> n >> S;
    vector<vector<int>> C(n, vector<int>(n));
    for(int s = 2; s <= n; s++) {
        for(int L=1, R=s; R<=n; L++,R++) {
            if(S[L] == S[R]) C[L][R] = C[L+1][R-1];
            else C[L][R] = min(C[L+1][R], C[L][R-1]) + 1;
        }
    }
    cout << C(0, n-1);
    return 0;
}
// Time complexity : Θ(n²)
```

11. (10 คะแนน) ขอให้พิจารณาปัญหาการเรียงข้อมูลดังต่อไปนี้ กำหนดให้มีข้อมูลคือ A[1..n] ซึ่งเราต้องการเรียงข้อมูลเหล่านี้จากน้อยไปหามาก โดยเราต้องการทราบ “ตำแหน่งของข้อมูลในลำดับต่าง ๆ” กล่าวคือ เราต้องการทราบ o[1..n] ที่ทำให้ A[o[1]] <= A[o[2]] <= ... <= A[o[n]] โดยที่ o[1..n] นั้นต้องเป็นการเรียงสับเปลี่ยนของตัวเลข 1 ถึง n (หมายความว่าเงื่อนไขสองข้อนี้ต้องเป็นจริง 1 <= o[i] <= n และ o[i] != o[j] ก็ต่อเมื่อ i != j) ตัวอย่างเช่น หากกำหนดให้ A = [20,30,10] แล้ว o จะต้องมีค่าเป็น o = [3,1,2] เนื่องจาก A[o[1]] <= A[o[2]] <= A[o[3]] และ o เป็นการเรียงสับเปลี่ยนของ 1 ถึง n)

ให้สังเกตว่ามันเป็นไปได้ที่จะมี o ที่แตกต่างกันหลายรูปแบบที่ตรงตามเงื่อนไขที่เราต้องการ จงตอบคำถามต่อไปนี้

11.1. กำหนดให้ n มีค่าเป็น 4 จงยกตัวอย่าง A ที่ทำให้ o ที่ตรงตามข้อกำหนดมีเพียงรูปแบบเดียว และ o = [4,1,3,2]

[2, 4, 3, 1]

11.2. กำหนดให้ n มีค่าเป็น 4 และ A[i] มีค่าเป็น 0 หรือ 1 เท่านั้น จงหา A ที่ทำให้มี o ที่ตรงตามข้อกำหนดอยู่ 6 แบบ

[0, 1, 1, 1]

11.3. กำหนดให้ข้อมูลนำเข้าคือ A และ n จงออกแบบอัลกอริทึมที่คำนวณ “จำนวนรูปแบบของ o ที่เป็นไปได้ทั้งหมด” ที่ตรงตามเงื่อนไขข้างต้น โดยอัลกอริทึมนี้จะต้องคืนค่าเป็นตัวเลขจำนวนเต็ม

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    cin.tie(nullptr)->sync_with_stdio(false);

    int n;
    cin >> n;
    unordered_map<int, int> freq;  // Assume we can access insert/access in O(1) :)
    for (int i=1; i<=n; i++) {
        int A;
        cin >> A;
        freq[A]++;
    }
    vector<int> fac(n+1);
    fac[0] = 1;
    for (int i=1; i<=n; i++) fac[i] = fac[i-1] * i;  // Do we need mod 10^9+7 :(
    int ans = 1;
    for (auto [k,v] : freq) ans *= fac[v];
    cout << ans;
    return 0;
}
```

ประสิทธิภาพเชิงเวลาคือ $\Theta(n)$ // using hash table :)