

FACULTY OF ENGINEERING  
CHULALONGKORN UNIVERSITY  
2110327 ALGORITHM DESIGN

Year II, Second Semester, Midterm Examination, March 8, 2019 13:00-16:00

ชื่อ-นามสกุล.....เลขประจำตัว.....ตอนเรียนที่.....เลขที่นั่ง CR58.....

หมายเหตุ

1. ข้อสอบมีทั้งหมด 9 ข้อ ในกระดาษคำถาม 6 หน้า
2. ไม่อนุญาตให้นำตำราและเอกสารใดๆ เข้าในห้องสอบ
3. ไม่อนุญาตให้ใช้เครื่องคำนวณใดๆ
4. ห้ามการหยิบยื่นสิ่งใดๆ ทั้งสิ้น จากผู้สอบอื่นๆ เว้นแต่เจ้าหน้าที่ควบคุมการสอบจะหยิบยื่นให้
5. ห้ามนำส่วนใดส่วนหนึ่งของข้อสอบและสมุดคำตอบออกจากห้องสอบ
6. ผู้เข้าสอบสามารถออกจากห้องสอบได้ หลังจากผ่านการสอบไปแล้ว 45 นาที
7. เมื่อหมดเวลาสอบ ผู้เข้าสอบต้องหยุดการเขียนใดๆ ทั้งสิ้น
8. นิสิตกระทำผิดเกี่ยวกับการสอบ ตามข้อบังคับจุฬาลงกรณ์มหาวิทยาลัย มีโทษ คือ พ้นสภาพการเป็นนิสิต หรือ ได้รับ สัญลักษณ์ F ในรายวิชาที่กระทำผิด และอาจพิจารณาให้ถอนรายวิชาอื่นทั้งหมดที่ลงทะเบียนไว้ในภาคการศึกษานี้

ห้ามนิสิตพกโทรศัพท์และอุปกรณ์สื่อสารไว้กับตัวระหว่างสอบ หากตรวจพบจะถือว่า  
นิสิตกระทำผิดเกี่ยวกับการสอบ อาจต้องพ้นสภาพการเป็นนิสิต หรือ ให้ได้รับ F และ  
อาจพิจารณาให้ถอนรายวิชาอื่นทั้งหมดที่ลงทะเบียนไว้ในภาคการศึกษานี้

**\* ร่วมรณรงค์การไม่กระทำผิดและไม่ทุจริตการสอบที่คณะวิศวกรรมศาสตร์ \***

ข้าพเจ้ายอมรับในข้อกำหนดที่กล่าวมานี้ ข้าพเจ้าเป็นผู้ทำข้อสอบนี้ด้วยตนเองโดยไม่ได้รับการช่วยเหลือ  
หรือให้ความช่วยเหลือ ในการทำข้อสอบนี้

ลงชื่อนิสิต.....

วันที่.....

1. (10 คะแนน) จงตอบคำถามต่อไปนี้สั้น ๆ ไม่ต้องอธิบาย

- $\log 1 + \log 2 + \log 3 + \dots + \log (n-1) + \log n = \Theta(\frac{n!}{n})$
- $1 + 2 + 3 + \dots + n = \Theta(\frac{n^2}{2})$
- การเรียงลำดับข้อมูลที่เรียงลำดับอยู่แล้ว  $n$  ตัวด้วย selection sort ใช้เวลา  $= \Theta(\frac{n^2}{2})$
- การเรียงลำดับข้อมูลที่เรียงลำดับอยู่แล้ว  $n$  ตัวด้วย insertion sort ใช้เวลา  $= \Theta(\frac{n^2}{2})$   $T(n) = T(n-1) + n$
- การเรียงลำดับข้อมูลที่เรียงลำดับอยู่แล้ว  $n$  ตัวด้วย merge sort ใช้เวลา  $= \Theta(\frac{n \log n}{2})$   $T(n) = T(n/2) + n \cdot \log n$
- ให้  $T(n) = T(n-2) + \Theta(n)$ ,  $T(0) = \Theta(1)$  จะได้ว่า  $T(n) = \Theta(\frac{n^2}{2})$
- ให้  $T(n) = 3T(n-1) + \Theta(1)$ ,  $T(0) = \Theta(1)$  จะได้ว่า  $T(n) = \Theta(3^n)$
- ให้  $T(n) = 3T(n/3) + \Theta(1)$ ,  $T(0) = \Theta(1)$  จะได้ว่า  $T(n) = \Theta(n)$
- การหา median of medians of fives ของข้อมูล  $n$  ตัว ด้วยวิธีที่เรียนมา ใช้เวลา  $= \Theta(n)$
- การหา longest common subsequence ของสตริงที่ยาว  $n$  สองตัวในกรณีแย่งสุด ด้วยวิธี dynamic programming ที่เรียนมา ใช้เวลา  $= \Theta(n^2)$

2. (10 คะแนน) จงวิเคราะห์ว่าอัลกอริทึมข้างล่างนี้ใช้เวลาเป็น  $\Theta$  อะไรของตัวแปร  $n$  (ให้การดำเนินการพื้นฐานเช่น  $*$  /  $+$  - และอื่น ๆ ใช้เวลา  $\Theta(1)$  และการหาร / เป็นการหารแบบปัดเศษทิ้ง) แสดงวิธีทำด้วยในช่องทางขวา

<pre>EE(A[1..n][1..n], B[1..n][1..n]) {     C = new array[1..n][1..n]     for (i=0; i&lt;n; i++) {         for (j=0; j&lt;n; j++) {             C[i][j] = 0             for (k=0; k&lt;n; k++)                 C[i][j] += A[i][k]*B[k][j]         }     }     return C }</pre>	$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \Theta(1)$ $= \Theta(n^3)$
<pre>RR(n) {     if (n==2) return 2*n     for (k=0; k&lt;n; k++)         for (i=0; i&lt;k; i++)             s += 2*k - i*i + 4     for (k=0; k&lt;4; k++)         s += RR(n/2)     return s }</pre>	$T(n) = 4(\frac{n^2}{2}) + \Theta(1) + \Theta(n^2)$ $\log_2 n = 2$ $f(n) = n^2$ $\therefore T(n) = \Theta(n^2 \log n)$
<pre>MM(d[1..n], t[1..n]) {     i = 1; j = n; k = 1     m = n/2     while (i&lt;=m and j&lt;=n) {         if (d[i] &lt; d[j]) t[k++] = d[i++]         else t[k++] = d[j++]     }     while (i&lt;=m) t[k++] = d[i++]     while (j&lt;=n) t[k++] = d[j++]     for (k = 1; k &lt; n; k++) d[k] = t[k] }</pre>	<p>พิจารณา while <math>(i \leq m \text{ and } j \leq n)</math></p> <p>1. มี time complexity เป็น <math>O(n)</math> → นอกสุดสุดสุด (cut)</p> <p>2. while <math>(i \leq m)</math>, while <math>(j \leq n)</math></p> <p>3. time complexity เป็น <math>O(n)</math></p> <p>4. for loop สุดท้าย เป็น <math>O(n)</math></p> <p><math>\therefore \Theta(n) + O(n) + O(n) + O(n) = \Theta(n)</math></p>
<pre>SS(d[1..n]) { # ทุกช่องใน d เก็บจำนวนเต็มบวก     SS(d, n, new array_of_zeros[n]) } SS(d[1..n], m, A[1..n]) {     if (m &lt; 1) return 1     if (m &lt; 4) return d[m]     if (A[m] &gt; 0) return A[m]     x1 = d[m]*SS(d, m-1, A)     x2 = d[m-1]*SS(d, m-2, A)     x3 = d[m-2]*SS(d, m-3, A)     A[m] = x1 + x2 + x3     return A[m] }</pre>	 <p>จะเห็นว่ามีการเรียกซ้ำอยู่</p> <p>ถ้าให้</p> <p><math>\therefore \Theta(n^4)</math></p>
<pre>SO(d[1..n]) {     for (k = 0; k &lt; n; k++)         insertion_sort(d) }</pre>	<p>การหา <math>\Theta(n^2)</math></p> <p>รวมทั้งหมด <math>\sum_{i=1}^n \Theta(n^2) = \Theta(n^3)</math> <math>\therefore</math> มี <math>\Theta(n^3)</math> #</p>

3. (10 คะแนน) จงตอบคำถามต่อไปนี้ในช่องว่างที่กำหนดให้ ไม่ต้องแสดงวิธีทำ

- $7^{47} \bmod 11$  มีค่าเท่ากับ ..... 6
- median-of-medians of 5 ของ [1,3,29,4,5], [14,11,19,18,21], [32,12,25,7,24], [13,17,2,15,20], [21,29,26,8,9] คือ .....
- การ merge ข้อมูล [1, 3, 8, 9] กับ [2, 4, 6, 7] จะเกิดการเปรียบเทียบข้อมูลจาก 2 อาร์เรย์นี้ ..... 6 ..... ครั้ง
- ให้  $X = \text{"ABCABC"}$ , ค่าของ  $Y$  ที่ทำให้การหา longest common subsequence ของ  $X$  กับ  $Y$  แบบ top-down (ไม่มี memoization) ทำงานได้เร็วสุด ๆ คือ .....
- ให้  $v = [15, 20, 20, 24, 40]$  และ  $w = [5, 4, 2, 4, 8]$  แทนมูลค่าและน้ำหนักของของ 5 ชิ้น และให้มีถุงที่รับน้ำหนักไม่เกิน 10 ต้องเลือกชิ้นไหน เท่าไร แบบ fractional จึงได้มูลค่ารวมสูงสุด ..... 2, 3, 4 ..... (เลือกชิ้น 3, 4, 5 ชิ้น 1 และ 5 ชิ้น 1)
- ให้  $v = [15, 20, 20, 24, 40]$  และ  $w = [5, 4, 2, 4, 8]$  แทนมูลค่าและน้ำหนักของของ 5 ชิ้น และให้มีถุงที่รับน้ำหนักไม่เกิน 10 ต้องเลือกชิ้นไหน แบบ 0/1 จึงได้มูลค่ารวมสูงสุด ..... 3, 3, 1, 4 .....
- ให้  $A = [(0,3), (5,8), (2,4), (4,5), (5,7), (7,9), (3,6)]$  แทนรายการของเวลาเริ่มและเวลาสิ้นสุดของกิจกรรมต่าง ๆ เราต้องเลือกกิจกรรมใดบ้าง (ที่ไม่ซ้อนเหลื่อมกัน) จึงได้จำนวนกิจกรรมมากที่สุด ..... 1, 3, 5, 6 .....
- ให้  $F = [('a',1), ('b',2), ('c',4), ('d',8), ('e',16), ('f',32), ('g',64)]$  แทนรายการของตัวอักษรและความถี่ของข้อมูลชุดหนึ่ง ถ้าเราเข้ารหัสตัวอักษรชุดนี้ด้วยรหัสแบบ Huffman จะใช้ข้อมูลจำนวน ..... 246 ..... บิตในการแทนข้อมูลชุดนี้ทั้งหมด
- มีข้อมูลอยู่ 1,000 ตัวที่เรียงลำดับจากมากมาน้อยแล้ว (ซ้ายไปขวา) ในอาร์เรย์หนึ่ง การค้นข้อมูลในอาร์เรย์นี้ด้วย binary search จะพิจารณาข้อมูลในอาร์เรย์ไม่เกิน ..... 9 ..... ตัว
- ให้  $A_1, A_2, A_3$  และ  $A_4$  คือเมทริกซ์ขนาด  $5 \times 100, 100 \times 2, 2 \times 100$  และ  $100 \times 5$  ตามลำดับ ถ้าต้องการหาผลคูณของ  $A_1 \times A_2 \times A_3 \times A_4$  ควรจัดลำดับการคูณอย่างไรจึงจะหาผลคูณได้เร็วสุด .....  $((A_1 A_2) (A_3 A_4))$  ..... (ตอบโดยใส่วงเล็บเพื่อกำหนดลำดับการคูณ)

4. (6 คะแนน) จากตารางค่าของคำตอบของปัญหาที่ใช้กำหนดการพลวัตต่อไปนี้ จงสร้างตารางของการตัดสินใจที่ถูกต้อง

• Longest Common Subsequence

(เติม ↑ หรือ ← หรือ ↖)

	A	A	B	C	A	C	A
A	0	0	0	0	0	0	0
B	0	1	1	1	1	1	1
C	0	1	1	2	2	2	2
A	0	1	1	2	3	3	3
C	0	1	2	2	3	4	4
C	0	1	2	2	3	4	5
A	0	1	2	2	3	4	5
A	0	1	2	2	3	4	5
A	0	1	2	2	3	4	5
A	0	1	2	2	3	4	5

	A	A	B	C	A	C	A
A							
B							
C							
A							
C							
C							
C							
A							
A							
A							

• 0/1 Knapsack

(เติม ✓ หรือ ✗ แทนการเลือกไม่เลือก)

	capacity								
empty	0	1	2	3	4	5	6	7	8
value=12 weight=4	0	0	0	0	12	12	12	12	12
value=10 weight=6	0	0	0	0	12	12	12	12	12
value=8 weight=5	0	0	0	0	12	12	12	12	12
value=14 weight=3	0	0	0	14	14	14	14	26	26
value=7 weight=1	0	7	7	14	21	21	21	26	33
value=9 weight=6	0	7	7	14	21	21	21	26	33
value=9 weight=2	0	7	9	16	21	23	30	30	33

	capacity								
empty	0	1	2	3	4	5	6	7	8
value=12 weight=4		X	X	X	✓	X	X	X	✓
value=10 weight=6		X	X	X	X	X	X	X	X
value=8 weight=5		X	X	X	X	X	X	X	X
value=14 weight=3		X	X	X	✓	✓	✓	✓	✓
value=7 weight=1		✓	X	✓	✓	X	✓	✓	✓
value=9 weight=6		X	X	X	X	X	X	X	X
value=9 weight=2		X	✓	✓	X	✓	✓	✓	X

$7^{47} \bmod 11$  มีค่าเท่ากับ

$28 \bmod 11 \approx 6$

$$(7 \bmod 11) ((7 \bmod 11)^2 \bmod 11) \bmod 11$$

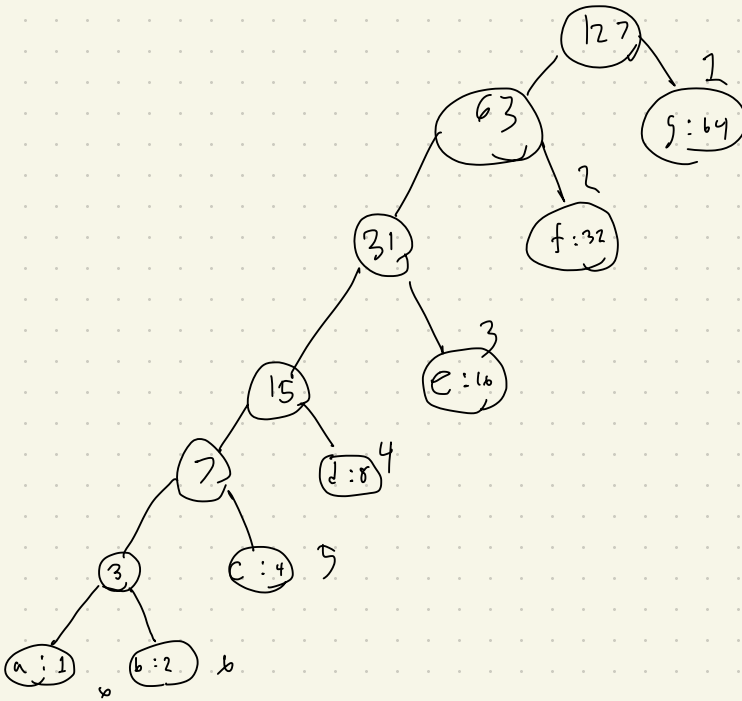
$$(7 \bmod 11) ((7^2 \bmod 11)^2 \bmod 11) \bmod 11 \quad 2$$

$$(7 \bmod 11) ((7^4 \bmod 11) \bmod 11) \bmod 11 \approx 7$$

$$(7 \bmod 11) ((7^8 \bmod 11)^2 \bmod 11) \bmod 11$$

$$7^4 \bmod 11 \approx 5 \quad 28 \bmod 11 \approx 6$$

- ให้  $F = \{(a,1), (b,2), (c,4), (d,8), (e,16), (f,32), (g,64)\}$  แผนรายการของตัวอักษรและความถี่ของข้อมูลชุดหนึ่ง ถ้าเราเข้ารหัสตัวอักษรชุดนี้ด้วยวิธีแบบ Huffman จะใช้ข้อมูลจำนวน ..... บิตในการแทนข้อมูลชุดนี้ทั้งหมด

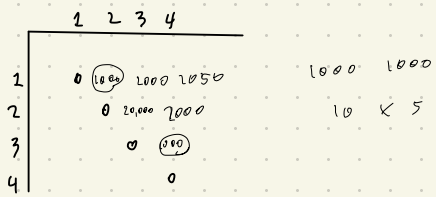


$$16 + 20 + 32 + 48 + 64 + 64$$

$$\begin{array}{r} 80 \\ 128 \\ 204 \\ 38 \\ \hline 248 \end{array}$$

1	1000	500
1	500	250
1	250	125
1	125	63
1	63	32
1	32	16
1	16	8
1	8	4
1	4	2
1	2	1

• ให้  $A_1, A_2, A_3$  และ  $A_4$  คือเมทริกซ์ขนาด  $5 \times 100, 100 \times 2, 2 \times 100$  และ  $100 \times 5$  ตามลำดับ ถ้าต้องการหาผลคูณของ  $A_1 \times A_2 \times A_3 \times A_4$   
 ควรจัดลำดับการคูณอย่างไรจึงจะหาผลคูณได้เร็วสุด  $(A_1 A_2)(A_3 A_4)$  ..... (ตอบโดยใส่ส่วนเรียงที่ก่อนและลำดับการคูณ)



5. (10 คะแนน) ให้  $D$  เป็นอาร์เรย์ขนาด  $n$  ช่อง ภายในเก็บจำนวนเต็ม 0 ถึง  $n$  แต่มีค่าหนึ่งหายไป (ที่ไม่ใช่  $n$ ) ข้อมูลใน  $D$  เรียงจากน้อยไปมากแล้ว เช่น  $D = [0, 1, 2, 4, 5, 6, 7]$  มี 3 หายไป,  $D = [1, 2, 3, 4, 5, 6]$  มี 0 หายไป โดยไม่มีกรณีไม่มีตัวหาย เช่น  $D = [0, 1, 2, 3, 4, 5]$  จงเขียนรหัสเทียมของอัลกอริทึมที่ใช้เวลา  $O(\sqrt{n})$  เพื่อหาข้อมูลที่หายไปในอาร์เรย์  $D$

```
missing( D[0..n-1] ) {
```

```
    int L = 0 ;
    int R = n-1 ;
    while ( L < R )
    {
        int mid = (L+R)/2 ;
        if ( D[mid] == mid )
        {
            L = mid+1 ;
        }
        else
        {
            R = mid ;
        }
    }
    return R ;
}
```

ยกตัวอย่างประกอบ

```

1 2 3
0 1 2

0 1 2 3 5 6 7
0 1 2 3 4 5 6

```

6. (10 คะแนน) ให้  $D$  คืออาร์เรย์ขนาด  $n$  ช่องที่เก็บจำนวนเต็ม จงเขียนรหัสเทียมของอัลกอริทึมที่ใช้เวลา  $O(\log n)$  เพื่อหาค่า "peak" (ขอค่า peak สักหนึ่งค่า) ใน  $D$  โดย peak คือค่าในอาร์เรย์ที่มีค่ามากกว่าหรือเท่ากับ ค่าของตัวก่อนหน้าทางซ้ายหนึ่งตัวและตัวถัดไปทางขวาหนึ่งตัว (ถ้าไม่มีตัวก่อนหน้าหรือตัวถัดไป ก็พิจารณาอีกข้างหนึ่งที่มีก็พอ) เช่น  $D = [9, 7, 7, 99, 4, 5, 6, 6, 5, 8]$  มี 9, 99, 6 และ 8 เป็น peak

```
peak( D[0..n-1] ) {
```

```
    peak( D, 0, n-1 )
```

```
}
```

```
peak( D, start, stop )
```

```
{ if ( n == 1 ) return D[n-1]
```

```
else
```

```
{ int mid = (start + stop) / 2 ;
```

```
int comp1 = max( 0, mid-1 ) ;
```

```
int comp2 = min( n-1, mid+1 ) ;
```

```
if ( D[mid] >= (comp1 && D[mid]) > D[comp2] )
```

```
return D[mid]
```

```
else if ( D[mid] < (comp1)
```

```
return peak( D, start, mid )
```

```
else if ( D[mid] < (comp2)
```

```
return peak( D, comp2, stop )
```

```
}
```

ยกตัวอย่างประกอบ

7. (10 คะแนน) จากความสัมพันธ์เวียนบังเกิดข้างล่างนี้ จงเขียนรหัสเทียมเพื่อแก้ปัญหาหานี้ด้วย bottom up dynamic programming

$$F(i, j) = \begin{cases} 0 & \text{if } i == 0 \text{ or } j == n \\ \max(F(i-1, j), F(i, j+p[i]) + q[i]) & \text{if } j+p[i] \leq n \\ F(i-1, j) & \text{otherwise} \end{cases}$$

รับประกันว่า  $p[i] > 0$  และ  $0 \leq i, j \leq n$

F( p[0..n], q[0..n] ) {

}

8. (10 คะแนน) นัทที่ต้องการขับรถจากกิโลเมตรที่ 1 ไปกิโลเมตรที่  $n$  ( $n \geq 2$ ) รถยนต์คันนี้มีถังน้ำมันซึ่งจุได้  $v$  ลิตร ( $v \geq 1$ ) แต่กินน้ำมันมาก คือ ต้องใช้น้ำมัน 1 ลิตรในการเดินทาง 1 กิโลเมตร ตอนเริ่มเดินทางไม่มีน้ำมันเลย โชคดีที่มีปั๊มน้ำมันทุก ๆ หลักกิโลเมตร โดยปั๊มน้ำมันที่หลักกิโลเมตรที่  $k$  ขายน้ำมันในราคา  $P_k$  บาทต่อลิตร คำถามคือ นัทที่ต้องจ่ายค่าน้ำมันน้อยสุดกี่บาท เพื่อที่จะเดินทางถึงจุดหมายที่กิโลเมตรที่  $n$  ได้ และน้ำมันหมดถังพอดี โดยนัทที่สามารถเติมน้ำมันที่หลักกิโลเมตรใดก็ได้

เช่น ถ้า  $n = 4$ ,  $v = 2$  และ  $P = [1, 2, 3, 4]$  คำตอบคือ 4 เพราะ ค่าน้ำมันน้อยสุดคือ เติมน้ำมัน 2 ลิตรที่ กม. 1 (2 ลิตร  $\times$  1 บาท/ลิตร = 2 บาท) ขับถึง กม. 2 เหลือ 1 ลิตร เติมน้ำมันอีก 1 ลิตร (1 ลิตร  $\times$  2 บาท/ลิตร = 2 บาท) ขับถึง กม. 4 ถึงจุดหมายและน้ำมันหมดพอดี

จงเขียนรหัสเทียมของอัลกอริทึมสำหรับแก้ปัญหาข้างบนนี้ด้วยกำหนดการพลวัต (dynamic programming) โดยต้องระบุความสัมพันธ์เวียนบังเกิด (recurrence), ขนาดของตารางที่ต้องใช้ และคำอธิบายย่อ ๆ

min\_cost( P[1..n], n, v ) { # P[k] contains oil price at k-th kilometer.

}

ความสัมพันธ์เวียนบังเกิด:

ขนาดของตาราง:

คำอธิบายย่อ ๆ (ยกตัวอย่างประกอบ):

8. (10 คะแนน) นัทที่ต้องการขับรถจากกิโลเมตรที่ 1 ไปกิโลเมตรที่  $n$  ( $n \geq 2$ ) รถยนต์คันนี้มีถังน้ำมันซึ่งจุได้  $v$  ลิตร ( $v \geq 1$ ) แต่กินน้ำมันมาก คือต้องใช้น้ำมัน 1 ลิตรในการเดินทาง 1 กิโลเมตร ตอนเริ่มเดินทางไม่มีน้ำมันเลย โชคดีที่มีปั๊มน้ำมันทุก ๆ หลักกิโลเมตร โดยปั๊มน้ำมันที่หลักกิโลเมตรที่  $k$  ขายน้ำมันในราคา  $P_k$  บาทต่อลิตร ค่าถามคือ นัทที่ต้องจ่ายค่าน้ำมันน้อยสุดกี่บาท เพื่อที่จะเดินทางถึงจุดหมายที่กิโลเมตรที่  $n$  ได้ และน้ำมันหมดถังพอดี โดยนัทที่สามารถเติมน้ำมันที่หลักกิโลเมตรใดก็ได้
- เช่น ถ้า  $n = 4$ ,  $v = 2$  และ  $P = [1, 2, 3, 4]$  คำตอบคือ 4 เพราะ ค่าน้ำมันน้อยสุดคือ เติมน้ำมัน 2 ลิตรที่ กม. 1 (2 ลิตร  $\times$  1 บาท/ลิตร = 2 บาท) ขับถึง กม. 2 เหลือ 1 ลิตร เติมน้ำมัน 1 ลิตร (1 ลิตร  $\times$  2 บาท/ลิตร = 2 บาท) ขับถึง กม. 4 ถึงจุดหมายและน้ำมันหมดพอดี
- จงเขียนรหัสเทียมของอัลกอริทึมสำหรับแก้ปัญหาข้างบนนี้ด้วยการกำหนดการพลวัต (dynamic programming) โดยต้องระบุความสัมพันธ์เวียนบังเกิด (recurrence), ขนาดของตารางที่ต้องใช้ และคำอธิบายย่อ ๆ

# save [1...n] [1...v] # with every element is -1

min\_cost( P[1..n], n, v )

n with 0

{ min\_cost ( P, 0, n, v )

n-1 with 1 to n-1 with 0  
n-1 with 1

{ min\_cost ( P, n, n, v )

{ if c save t n [n] == -1 )

{ int min = 10000000 ;

for (int i = 1 to v)

{ int tmp = mincost ( P, i,

}



9. ให้  $D = [d_1, d_2, d_3, \dots, d_n]$  เป็นรายการที่แต่ละช่องเก็บเลขโดด 0 ถึง 9 เช่น  $[5, 1, 8, 7, 1, 1]$  นิยามให้  $D_{i,j}$  คือจำนวนที่ได้จากการนำเลขโดดใน  $D$  มาต่อกันตั้งแต่ตัวที่  $i$  ถึง  $j$  เช่น  $D_{2,2} = 1$ ,  $D_{2,3} = 18$ ,  $D_{2,4} = 187$  เป็นต้น

- (5 คะแนน) จงเขียนรหัสเทียมของอัลกอริทึม  $\text{max\_2}(D)$  ที่ใช้เวลา  $O(n)$  เพื่อหา  $D_{i,j}$  ที่มีจำนวนหลักมากที่สุดที่มีค่าที่หารด้วย 2 ลงตัว ถ้าไม่มี ให้คืน 0 (วิเคราะห์ประสิทธิภาพการทำงานเชิงเวลาด้วย) เช่น  $\text{max\_2}([5, 1, 8, 7, 1, 1])$  ได้ผลคือ 518

```
max_2( D[1..n] ) {
```

```
}
```

วิเคราะห์เวลาการทำงาน:

คำอธิบายหลักการทำงาน (ยกตัวอย่างประกอบ):

- (10 คะแนน) จงเขียนรหัสเทียมของอัลกอริทึม  $\text{max\_3}(D)$  ที่ใช้เวลา  $O(n)$  เพื่อหา  $D_{i,j}$  ที่มีจำนวนหลักมากที่สุดที่มีค่าที่หารด้วย 3 ลงตัว ถ้าไม่มี ให้คืน 0 (วิเคราะห์ประสิทธิภาพการทำงานเชิงเวลาด้วย) เช่น  $\text{max\_3}([5, 1, 8, 7, 1, 1])$  ได้ผลคือ 18711 (ข้อเสนอแนะ: จำนวนเต็ม  $N$  หารด้วย 3 ลงตัว เมื่อผลบวกของเลขโดดทุกตัวใน  $N$  หารด้วย 3 ลงตัว เช่น  $1+8+7+1+1 = 18$  หารด้วย 3 ลงตัว ดังนั้น 18711 หารด้วย 3 ลงตัว)

```
max_3( D[1..n] ) {
```

```
}
```

วิเคราะห์เวลาการทำงาน:

คำอธิบายหลักการทำงาน (ยกตัวอย่างประกอบ):