บทที่ 2 แถวลำดับ (Arrays)

2.1 แถวลำดับหนึ่งมิติ
2.2 แถวลำดับหลายมิติ
2.3 แถวลำดับใน COBOL และ Pascal
2.4 การแปลงส่งไปยังหน่วยเก็บ : แถวลำดับหนึ่งมิติ
2.5 การแปลงส่งไปยังหน่วยเก็บ : แถวลำดับหลายมิติ
2.6 แถวลำดับแบบสามเหลี่ยม
2.7 แถวลำดับสพาซ
แบบฝึกหัด

IT 204 33

บทที่ 2
แถวลำดับ

2.1 แถวลำดับหนึ่งมิติ (One – dimensional arrays)

แถวลำดับ หมายถึงเซตจำกัดแบบอันดับของสมาชิกที่มีแบบชนิดข้อมูลเหมือนกัน
(An array is a finite, ordered set of homogeneous elements)

ตัวอย่าง

- แถวลำดับชุดหนึ่งประกอบด้วยสมาชิกที่เป็น strings ทั้งหมด
- แถวลำดับอีกหนึ่งชุดประกอบด้วยสมาชิกที่เป็น integers ทั้งหมด

สมาชิกของแถวลำดับแต่ละตัวอาจเป็นแถวลำดับได้แถวลำดับ อาจเรียกว่า ตาราง (tables) รูปแบบที่ง่ายที่สุดของแถวลำดับคือ แถวลำดับหนึ่งมิติ หรือ เวกเตอร์
(The simplest form of array is one – dimensional array, or vector.)

ตัวอย่าง แถวลำดับหนึ่งมิติชื่อ VICTOR มีสมาชิก N ตัว

VICTOR(1)	VICTOR(2)	. . . . .	VICTOR(I)	. . . . .	VICTOR(N)

ดรรชนีล่าง (Subscripts)

ดรรชนีล่างของสมาชิกคือ สิ่งที่บอกตำแหน่งของสมาชิกในการเรียงอันดับของแถวลำดับ
(The subscript, or index, of an element designates its position in the array's ordering).

ดรรชนีล่างของสมาชิกของแถวลำดับจะอยู่ภายในเครื่องหมายวงเล็บ
โปรดสังเกตว่า แถวลำดับหนึ่งชุดต้องกำหนดชื่อหนึ่งชื่อและสมาชิกของแถวลำดับอ้างโดยดรรชนีล่างของมัน

บทนิยาม (definition)

The one–dimensional array A with elements of data structure type T having subscripts extending from L through U is

A (L : U) = { A (I)}

for I = L, L+1, . . ., U+1, U

where each element A(I) is a data structure of type T.

สัญกรณ์ A(L : U) แสดงว่าค่าดรรชนีล่างของ A มีพิสัยจาก L ถึง U
จำนวนสมาชิกของแถวลำดับเรียกว่า พิสัย (range) ของมันดังนั้นพิสัยของแถว
ลำดับ A(L : U) คือ U-L+1
พิสัยของแถวลำดับ (B1: N) คือ N
ขอบเขตล่าง(Lower bound) หมายถึง ค่าต่ำสุดของดรรชนีล่างของแถวลำดับ
ขอบเขตบน (Upper bound) หมายถึง ค่าสูงสุดของดรรชนีล่างของแถวลำดับ
ขอบเขตล่างไม่จำเป็นต้องเป็น 1 เสมอไป อาจมีค่าเป็น 0 หรืออาจมีค่าเป็นลบได้

ตัวอย่าง แถวลำดับหนึ่งมิติชื่อ TEMP ซึ่งบันทึกอุณหภูมิทุกชั่วโมงในช่วง 24 ชั่วโมง

TEMP(I) , สำหรับ $1 \leq I \leq 24$

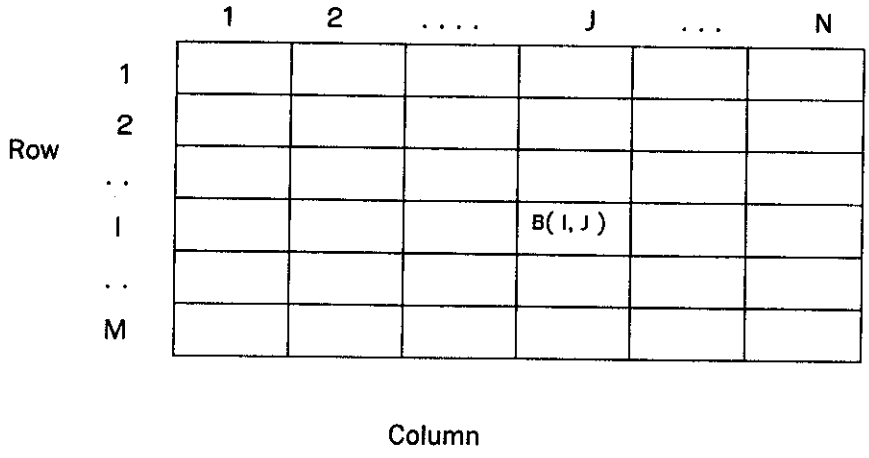
ตัวอย่าง
แถวลำดับหนึ่งมิติซึ่งเป็นตารางรายได้เฉลี่ยของประชากรในแต่ละจังหวัดของประเทศไทย

AVG-INCOME (I) , สำหรับ $1 \leq I \leq 76$

2.2 **แถวลำดับหลายมิติ (Multi – dimensional arrays)**
แถวลำดับสองมิติ หมายถึง แถวลำดับซึ่งสมาชิกแต่ละตัวนั้นตัวมันเองเป็นแถวลำดับ
(A two – dimensional array is an array in which each element is itself an array).

ตัวอย่าง แถวลำดับชื่อ B มีสมาชิก M ตัว , สมาชิกแต่ละตัวเป็นแถวลำดับของสมาชิก

N ตัว ทั้งหมดนี้คือตารางขนาด M × N



ดรรชนีล่าง (Subscripts)

แถวลำดับสองมิติมีค่าดรรชนีล่างสองตัวเพื่อแสดงถึงสมาชิกแต่ละตัว ดรรชนีล่างตัวแรก หมายถึง แถว (row) ของแถวลำดับส่วนดรรชนีล่างตัวที่สอง หมายถึง สดมภ์ (column) ของแถวลำดับและ B (I,J) คือสมาชิกของแถวลำดับสองมิติชื่อ B อยู่ในตำแหน่งแถวที่ I และสดมภ์ที่ J ของแถวลำดับ B

บทนิยาม (Definitions)

The two-dimensional array B with individual elements of type T hawing row subscripts extending from 1 to M and column subscripts form 1 to N is

B (1 : M, 1 : N) = { B(I , J) }

for I = 1, . . . , M and J = 1, . . . , N

where each B (I,J) is a data structure of type T.

มิติของแถวลำดับ B คือ M × N ในแต่ละแถวมีสมาชิก N ตัวในแต่ละสดมภ์มีสมาชิก

M ตัวเพราะฉะนั้นจำนวนสมาชิกทั้งหมดของแถวลำดับ B คือ

$$M \times N \quad \text{ตัว}$$

โดยทั่วไป แถวลำดับสองมิติ B ซึ่งดรรชนีล่างตัวแรกมี Lower bound L_1 และ upper bound U_1 และดรรชนีล่างตัวที่สอง lower bound L_2 และ upper bound U_2 นิยามดังนี้

$$B(L_1:U_1, L_2:U_2) = \{(B(I,J))\}$$

for $L_1 \leq I \leq U_1$ and $L_2 \leq J \leq U_2$

where each B(I,J) is a data structure of type T.

จำนวนสมาชิกในหนึ่งแถวของ B เท่ากับ $U_2 - L_2 + 1$

จำนวนสมาชิกในหนึ่งสดมภ์ของ B เท่ากับ $U_1 - L_1 + 1$

ดังนั้น จำนวนสมาชิกทั้งหมดของแถวลำดับ B คือ

$$(U_2 - L_2 + 1) * (U_1 - L_1 + 1)$$

ตัวอย่าง แถวลำดับสองมิติซึ่งข้อมูลคือเกรดของนักศึกษาในชั้นเรียนหนึ่งสำหรับการสอบสี่ภาคการศึกษา

GRADE (I, J) คือเกรดของนักศึกษาคนที่ I
สำหรับการสอบครั้งที่ J
$1 \leq I \leq M$ และ $1 \leq J \leq 4$

สมาชิกแต่ละตัวของแถวลำดับคือ grade ซึ่งอาจจะมีค่าเป็น integer หรือ เลข fixed – point มีทศนิยมสองตำแหน่ง หรืออาจจะเป็น character ทั้งหมดนี้ขึ้นอยู่กับข้อตกลงที่นำมาใช้

ภาคตัดขวาง (Cross–section)

cross–section ของแถวลำดับสองมิติได้มาโดยการให้ดรรชนีล่างหนึ่งตัวคงที่ ขณะที่มีการผันแปรดรรชนีล่างอีกตัวหนึ่ง ผ่านพิสัยทั้งหมดของค่าต่าง ๆ

สัญกรณ์ที่ใช้แทนภาคตัดขวางคือ * (asterisk) สำหรับค่าดรรชนีล่างซึ่งเป็นพิสัยทั้งหมดของค่าต่าง ๆ

ตัวอย่าง B (*, 4)หมายถึง สดมภ์ที่สี่ของแถวลำดับ B นั่นคือ

B (*,4) = { B (1,4), B(2,4), B(3,4), . . . , B(M,4)}

ในทำนองเดียวกัน

B (I,*) คือแถวที่ I ของแถวลำดับ B

GRAPE (*, 3) คือสดมภ์ของค่า GRADE สำหรับการบันทึก grade ผลสอบสำหรับนักศึกษาทั้งหมดในการสอบครั้งที่สาม

ตัวอย่าง ภาษา PL/1ซึ่งเขียนดังนี้

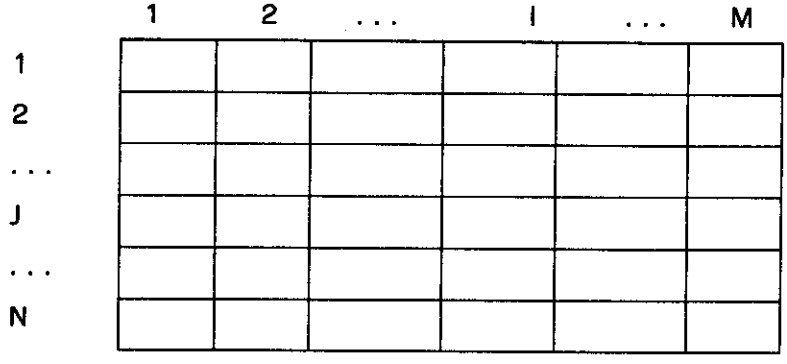
A (*,*)หมายถึง สมาชิกทุกตัวของแถวลำดับสองมิติที่ A

การสลับเปลี่ยน (Transpose)

การสลับเปลี่ยนของแถวลำดับสองมิติได้มาโดยการย้อนลำดับตำแหน่งดรรชนีล่าง (The transpose of a two-dimensional array is obtained by reversing the subscript position.)

การสลับเปลี่ยนของแถวลำดับขนาด M ×N คือแถวลำดับขนาด N× M

การสลับเปลี่ยนของแถวลำดับ B ใช้สัญลักษณ์ B^T



โดยบทนิยาม

$$B(I,J) = B^T(I,J)$$

ตัวอย่าง

$$B(3,8) = B^T(8,3)$$

หมายถึงสมาชิกในแถวที่ 3 สดมภ์ที่ 8 ของแถวลำดับ B เท่ากับสมาชิกในแถวที่ 8 สดมภ์ที่ 3 ของการสลับเปลี่ยนของแถวลำดับ B

ส่วนขยายให้กับมิติที่เพิ่มขึ้น (extensions to more dimensions)

แถวลำดับสามารถให้นิยามเป็นสามมิติ , สี่มิติ, N มิติแนวคิดของ subscript range และจำนวนของสมาชิกสามารถขยายได้โดยตรงจากแถวลำดับหนึ่งมิติและสองมิติไปเป็นแถวลำดับที่มีลำดับสูงขึ้น

โดยทั่วไป แถวลำดับ N – มิติ ต้องมีดรรชนีล่าง N ตัวกำหนดอันดับเพื่อระบุสมาชิกแต่ละตัวของแถวลำดับแถวลำดับ N – มิติ กำหนดดังนี้

$$A(L_1 : U_1, L_2 : U_2, \ldots, L_N : U_N)$$

สมาชิกแต่ละตัวของแถวลำดับ A กำหนดโดย

$$A(I_1, I_2, \ldots, I_N)$$

เมื่อดรรชนีล่าง I_k แต่ละตัวอยู่ภายในขอบเขตที่ถูกต้อง

$$I_k \leq I_k \leq U_k \text{ for eack } k = 1,2, \ldots, N$$

จำนวนสมาชิกทั้งหมดของแถวลำดับ A คือ

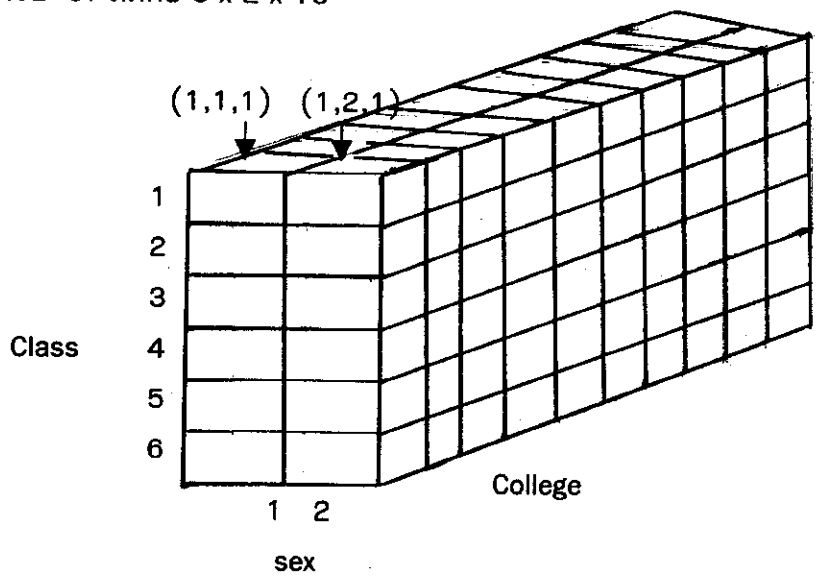
$$\prod_{K=1}^{N} (U_k - L_k + 1)$$

ซึ่งอีกทางเลือกหนึ่งเขียนดังนี้

$$(U_1 - L_1 + 1) * (U_2 - L_2 + 1) * .... * (U_N - L_N + 1)$$

ตัวอย่าง แถวลำดับสามมิติ ประกอบด้วยข้อมูลซึ่งเป็น จำนวนนักศึกษาในแต่ละชั้นปีนักศึกษาของมหาวิทยาลัย (ชั้นปีหนึ่ง, ปีที่สอง, ปีที่สาม, ปีที่สี่, นักศึกษาปริญญาโทและกลุ่มแยกชั้นปีไม่ได้)แบ่งตามเพศ สำหรับวิทยาลัย 10 แห่ง ของมหาวิทยาลัย แถวลำดับชุดนี้ตั้งชื่อว่า

COLLECE-CT มิติคือ 6 x 2 x 10



เพราะฉะนั้นค่าของสมาชิก COLLEGE-CT (I,J,K) คือ integer แทนจำนวนนักศึกษาในชั้นเรียนที่ I เพศ J วิทยาลัย K เพื่อให้ถูกต้อง I ต้องเป็น 1,2,3,4,5 หรือ 6 ส่วนค่าของ J ต้องเป็น 1 หรือ 2 สำหรับ K ต้องมีค่าระหว่าง 1 และ 10

ภาคตัดขวางของแถวลำดับหลายมิติกระทำโดยการให้ค่าดรรชนีล่างหนึ่งตัว หรือมากกว่าหนึ่งตัวเป็นตัวคงที่ (constant) และแต่ละตัวของดรรชนีล่างตัวอื่นผันแปรผ่านค่าของพิสัยทั้งหมด

ตัวอย่าง COLLEGE-CT (5,*,3)

หมายถึง แถวของ COLLEGE-CT ประกอบด้วยจำนวนนักศึกษาในชั้นเรียนที่ห้า (ปริญญาโท) ของวิทยาลัยที่ 3 ทั้งสองเพศ

ตัวอย่าง COLLEGE-CT(*,*,3)

หมายถึง ระนาบของ COLLEGE-CT ประกอบด้วยจำนวนนักศึกษาในวิทยาลัยที่3 สำหรับทุกชั้นเรียนและทั้งสองเพศ

("plane " of COLLEGE – CT containing the number of students in college 3, for each class and both sexes.)

2.3 แถวลำดับใน COBOL และ Pascal (Arrays in COBOL and Pascal)

แถวลำดับหนึ่งมิติ

ภาษาโปรแกรมร่วมส่วนใหญ่ใช้การเข้าถึงเหมือนกันทั้งหมดกับการประกาศตัวแปรซึ่งเป็นแถวลำดับ

ในการประกาศแถวลำดับต้องกำหนดสามสิ่งดังนี้

1) ชื่อตัวแปร
2) พิสัยของดรรชนีล่าง
3) ชนิดของโครงสร้างข้อมูลสำหรับสมาชิกของแถวลำดับ

ตัวอย่าง จงพิจารณาการประกาศครั้งแรกของแถวลำดับหนึ่งมิติชื่อ TEMP โดยมีพิสัยของดรรชนีล่างจาก 1 ถึง 24 สมมติว่าสมาชิกแต่ละตัวของแถวลำดับคือ integer มีพิสัยของค่า 0 ถึง 99

ใน COBOL , ส่วนที่เป็น DATA DIVISION เขียนดังนี้

```
01 TEMP – TABLE.
        02 TEMP OCCURS 24 TIMES PICTURE 99.
```

ภาษา Pascal แถวลำดับถูกประกาศโดยใช้ข้อความสั่ง var

```
var     temp : array [1. . 24] of Integer;
```

lower bound ของดรรชนีล่างในภาษา Pascal อาจจะไม่ใช่ 1

ตัวอย่าง

```
var graphpts : array [-100 . . . 100] of Integer;
```

แถวลำดับใน COBOL, ดรรชนีล่างต้องเริ่มต้นจาก 1 ส่วนที่เป็น OCCURS CLAUSE หมายถึง upper bound ของดรรชนีล่าง

แถวลำดับสองมิติ (Two – dimension arrays)

ตัวอย่าง การประกาศของแถวลำดับประกอบด้วยคะแนนของนักศึกษา 40 คน สอบสี่ครั้ง

ใน COBOL เขียนดังนี้

```
01 GRADE-TABLE.
    02 STUDENT OCCURS 40 TIMES.
        03 GRADE OCCURS 4 TIMES  PICTURE 99V9.
```

ใน Pascal เขียนดังนี้

```
var grade : array [1. . 40, 1. .4] of real;
```

ใน COBOL เกรดของนักศึกษาคนที่ I สอบครั้งที่ J คือ

```
GRADE (I,J)
```

ใน Pascal เขียน grade [i,j]

ค่าดรรชนีต้องอยู่ภายในขอบเขตถูกต้อง

มิติที่มากขึ้น (More dimensions)

จำนวนมิติสูงสุดซึ่งสามารถประกาศสำหรับแถวลำดับ COBOL คือ สามมิติ

ตัวอย่าง

```
01 COLLEGE-CT-TABLE .
    02 CLASS OCCURS 6 TIMES.
        03 SEX OCCURS 2 TIMES.
            04 COLLEGE-CT OCCURS 10 TIMES
                PICTURE 9(5).
```

ใน Pascal, แถวลำดับสามมิตินิยามดังนี้

```
var collegect : array [1..6, 1..2, 1..10] of integer;
```

ในบางภาษา เช่น FORTRAN และ COBOL เนื้อที่หน่วยเก็บถูกจัดสรรให้กับแถวลำดับ ณ. เวลาคอมไพล์ (compile time) เพราะฉะนั้น ขอบเขต (bounds) ต้องถูกกำหนดชัดแจ้งเมื่อให้นิยามแถวลำดับ

ในภาษาอื่น ๆ เช่น PL/1 เนื้อที่ถูกจัดสรรอย่างพลวัต ณ.เวลากระทำการ (execution time)

การดำเนินการบนสมาชิกของแถวลำดับ (Operations on Array Elements)

ตัวอย่าง ใน COBOL

```
COMPUTE  TOTAL-PAY (I) = HRLY-RATE(I) * HAS-WORKED (I).
```

การดำเนินการบนแถวลำดับ

บางภาษามีการปฏิบัติการบนแถวลำดับ ตัวอย่างเช่นถ้า A ถูกประกาศให้เป็นแถวลำดับใน PL/1 เพราะฉะนั้น A = A+ 2

หมายถึง ให้บวก 2 กับสมาชิกทุกตัวของ A

ถ้า Aและ B ถูกประกาศเป็นแถวลำดับที่มีมิติเหมือนกันดังนั้นใน PL/1

```
A = A * B
```

หมายถึง ให้คูณสมาชิกแต่ละตัวของ A ด้วยสมาชิก B ที่สมนัยกัน(นั่นคือ สมาชิกที่มีดรรชนีล่างเหมือนกัน) ผลลัพธ์เก็บใน A การคูณแถวลำดับเช่นนี้ ไม่ใช่การคูณเมทริกซ์ (not matrix multiplication)

ใน PL/1 การดำเนินการสามารถกระทำบนภาคตัดขวางของแถวลำดับ

ตัวอย่าง

```
GRADE (20 , *) = 0 ;
```

หมายถึง กำหนดให้สมาชิก GRADE แต่ละตัว ในแถวที่ 20 มีค่าเท่ากับ 0

ตัวอย่าง

VECTOR (*) = ARRAY1 (I, *) * ARRAY2 (*,I); หมายถึง ให้คูณสมาชิกซึ่งอยู่แถวที่ I ของ ARRAY 1 กับสมาชิกของสดมภ์ที่ I ของ ARRAY2

แถวลำดับต้องประกาศค่าพิสัยดรรชนีล่างที่เข้ากันได้ ตัวอย่างเช่น ถ้าดรรชนีล่างตัวที่สองของ ARRAY 1 พิสัยจากค่า 0 ถึง 25 เพราะฉะนั้นดรรชนีล่างตัวแรกของ ARRAY ต้องมีพิสัยจาก 0 ถึง 25 ด้วยเช่นกันและเป็นดรรชนีล่างของลำดับ VECTORเพราะฉะนั้น การคูณข้างต้นว่ามีผลเช่นเดียวกับลูบข้างล่างนี้

```
DO K = 0 TO 25
        VECTOR(K) = ARRAY1(I,K) * ARRAY2(K,I);
END;
```

ขอบเขตล่าง : หนึ่ง

(LOWER BOUND : One)

เริ่มต้นจงพิจารณาการแปลงส่งหน่วยเก็บสำหรับแถวลำดับหนึ่งมิติ ชื่อ EMP-NO เมื่อ lower bound ของดรรชนีล่างคือ 1 และ upper bound ของดรรชีล่างคือ N วิธีหนึ่งที่ใช้เก็บแถวลำดับนี้ คือ เพื่อให้การเรียงอันดับเชิงกายภาพของสมาชิก เหมือนกับการเรียงอันดับเชิงตรรกะของสมาชิก

หน่วยเก็บสำหรับสมาชิก EMP-NO (I-1) จะประชิดกับหน่วยเก็บสำหรับสมาชิก EMP-NO (I) สำหรับ I = 1, 2 . . . , N-1

ในการคำนวณหาเลขที่อยู่เริ่มต้น (starting address) นั่นคือ ตำแหน่ง (location) ของสมาชิก EMP-NO (I) จำเป็นต้องทราบ

1. เลขที่อยู่เริ่มต้นของเนื้อที่หน่วยเก็บที่จัดสรรให้เก็บแถวลำดับ
2. ขนาดของสมาชิกแต่ละตัวในแถวลำดับ (The size of each element in the array)

ให้ B เป็นเลขที่อยู่เริ่มต้นของแถวลำดับ เราเรียกว่า Base location (ตำแหน่งฐาน)

สมมติว่าสมาชิกแต่ละตัวของแถวลำดับใช้เนื้อที่ S ไบต์

2.4 การแปลงส่งไปยังหน่วยเก็บ : แถวลำดับหนึ่งมิติ

(Mapping to storage : one – dimension arrays)

เช่นเดียวกับโครงสร้างข้อมูลอื่น ๆ มีหลายวิธีที่ใช้แทนที่แถวลำดับในหน่วยความจำ โครงร่างการแทนที่เหล่านี้ถูกประเมินผลบนหลักการของคุณสมบัติสี่ข้อดังนี้

(1) ความง่ายของการเข้าถึงสมาชิก (the simplicity of element access)

(2) ง่ายต่อการแวะผ่านทางเดินต่าง ๆ (ease of traversal along various paths)

(3) เก็บอย่างมีประสิทธิภาพ (storage efficiency)

(4) ง่ายต่อการเพิ่มจำนวน (ease of growth)

เป็นไปไม่ได้ที่จะจัดให้เหมาะสม (optimize) ทั้งสี่ปัจจัยให้เกิดพร้อมกัน

ดังนั้น ตำแหน่งของสมาชิกตัวที่ I ของแถวลำดับคือ

$$B + (I - 1) * S \qquad (1)$$

เพราะว่ามีสมาชิก (I- 1) ตัว , แต่ละตัวขนาด S ไบต์ อยู่ก่อนหน้าเชิงกายภาพสมาชิกตัวที่ I คอมไพเลอร์ (compiler) จำเป็นต้องทราบตำแหน่งของสมาชิก ซึ่งอาจจะถูกโปรแกรมเมอร์นำไปใช้โดยการให้ memory dump เพื่อแก้ไขที่ผิด

Generalizing the lower Bound

ขณะนี้เราจะขยายสมการ (1) เพื่อหาตำแหน่งของสมาชิกตัวที่ I ของแถวลำดับซึ่ง lower bound ของดรรชนีล่างไม่เท่ากับหนึ่ง

ตัวอย่างเช่น จงพิจารณาแถวลำดับซึ่งประกาศดังนี้

Z1(4 : 10) เป็นกรณีเฉพาะ, เลขที่อยู่เริ่มต้นของ Z1(6) คือ

B + (6-4) * 3

เพราะว่า (6 – 4) เท่ากับ 2 คือมีสมาชิกอยู่ 2 ตัวอยู่ข้างหน้า Z1(6)

ตัวอย่าง

สำหรับแถวลำดับประกาศดังนี้ Z2(-2 : 2) ตำแหน่งของ Z2(1) คือ

B + (1 – (-2)) * 3

เพราะว่า 1 – (-2) หรือ 3 มีสมาชิกสามตัวคือ Z2(-2), Z2(-1) , และ Z2(0) นำหน้า Z2(1)

กรณีทั่วไป สมาชิก ARRAY(I) ของแถวลำดับนิยามดังนี้ ARRAY(L : U) อยู่ที่ตำแหน่ง

B + (I – L) * S

สูตรนี้ถูกต้องไม่ว่า lower bound L จะมีค่าเป็นบวก เป็นลบหรือไม่ใช่ค่า เป็นลบก็ตาม

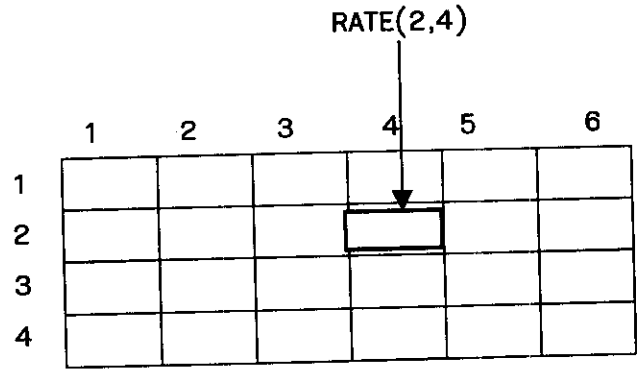
2.5 การแปลงส่งไปหน่วยเก็บ : แถวลำดับหลายมิติ

(Mapping to storage : Multi – dimensional arrays)

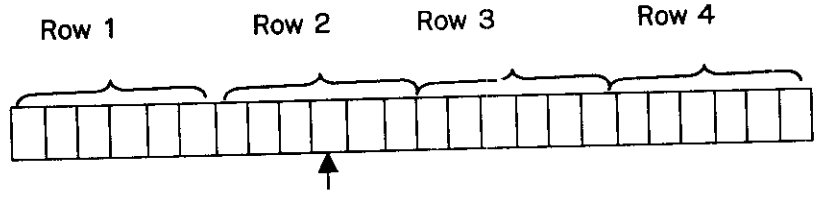
เรียงอันดับทีละแถว (Row – major order)

เนื่องจากหน่วยความจำคอมพิวเตอร์เป็นเชิงเส้นเพราะฉะนั้นแถวลำดับหลายมิติต้องเป็นแบบเชิงเส้นขณะที่แปลงส่งไปยังหน่วยเก็บ ทางเลือกหนึ่งสำหรับการทำให้เป็นเชิงเส้นคือขั้นแรก เก็บแถวที่หนึ่งของ array จากนั้นเก็บแถวที่สองของ array จากนั้นเก็บแถวที่สาม ทำเช่นนี้เรื่อยไป

ตัวอย่าง แถวลำดับนิยามโดย RATE (1 : 4, 1 : 6) ซึ่งปรากฏเชิงตรรกะในรูป 2-6 และปรากฏเชิงกายภาพในลักษณะ row – major order ในรูป 2-7



รูป 2-6 ตัวอย่างแถวลำดับสองมิติ



รูป 2-7 แถวลำดับรูป 2-6 ทำให้เป็นเชิงเส้น แบบ row – major order

นี่คือโครงร่างการจัดสรรหน่วยเก็บใช้สำหรับแถวลำดับซึ่งประกาศในการทำให้เกิดผลในทางปฏิบัติส่วนใหญ่ของ COBOL, Pascal, C , และ PL/1

สมมติว่า B คือเลขที่อยู่ฐานและสมาชิกแต่ละตัวของแถวลำดับใช้เนื้อที่ S ไบต์ เลขที่อยู่เริ่มต้นของ RATE (I, J) คือ

$$B + (I - 1) * 6 * S + (J - 1) * S$$

เพราะว่ามี I – 1 แถว แต่ละแถวความยาว 6 * S ซึ่งอยู่หน้าแถวซึ่ง RATE (I,J) อยู่ และมีสมาชิก J – 1 ตัว ซึ่งแต่ละตัวความยาว S ที่อยู่หน้า RATE (I , J) ในแถวที่ I จากตัวอย่าง RATE (2,4) จึงอยู่ที่ตำแหน่ง B + 9 * S กรณีทั่วไป สมาชิก ARRAY(I ,J) ของแถวลำดับนิยามดังนี้

ARRAY($L_1 : U_1 , L_2 : U_2$)
อยู่ที่ตำแหน่ง

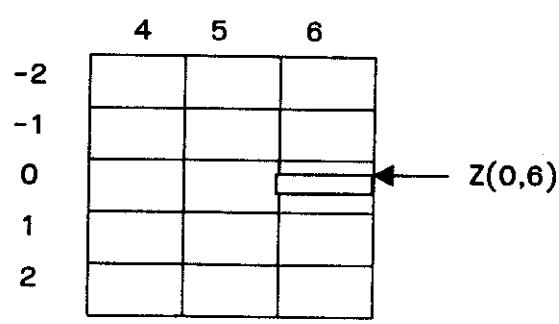
$$B + (I - L_1) * (U_2 - L_2 + 1) * S + (J - L_2) * S$$

เพราะว่ามี I – L แถวซึ่งแต่ละแถวความยาว

($U_2 - L_2 + 1$) * S ซึ่งอยู่หน้าแถวที่ ARRAY(I,J) อยู่

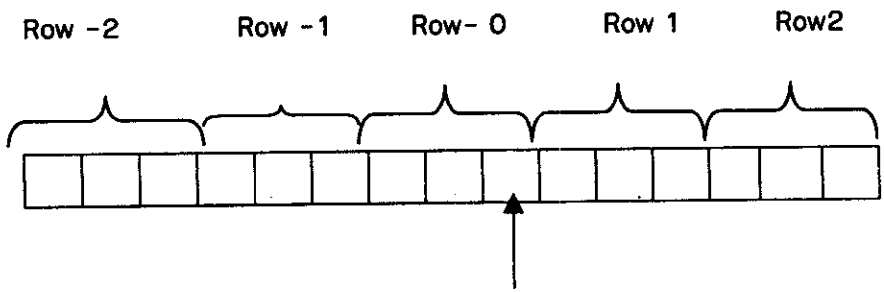
และมีสมาชิก J – L_2 ตัว แต่ละตัวความยาว S ซึ่งอยู่หน้า ARRAY(I , J) ในแถวที่ I

ตัวอย่าง แถวลำดับ Z(-2 : 2 , 4 : 6) อธิบายเชิงตรรกะในรูป 2-8



รูป 2-8 ตัวอย่างแถวลำดับสองมิติ

แถวลำดับข้างต้นนี้เชิงกายภาพเรียงลำดับทีละแถวแสดงในรูป 2-9



รูป 2-9 แถวลำดับรูป 2-8 ทำให้เป็นเชิงเส้นในการเรียงลำดับทีละแถว

มีสองแถว (แถว -2 และแถว-1) ซึ่งแต่ละแถวความยาวเท่ากับ 3 * S อยู่หน้าแถว 0
ในแถว 0 มีสมาชิกสองตัว (6 - 4 = 2) แต่ละตัวความยาว S อยู่หน้า Z (0,6)
เพราะฉะนั้นตำแหน่งเริ่มต้อนของ Z (0,6) คือ

B + (0 - (- 2)) * (6 - 4 + 1) * S + (6 - 4) * S เท่ากับ B + 8 * S
กรณีแถวลำดับ N มิติเรียงลำดับทีละแถวการผันแปรดรรชนีล่างจะเป็นจากขวาไปซ้าย
(right-to-left order)

ตัวอย่าง row-major order แถวลำดับ A ($L_1 : U_1 , L_2 : U_2 , \ldots , L_N : U_N$) สมาชิกเก็บ
ในอันดับข้างล่างนี้

$A (L_1 , L_2 , \ldots , L_N)$
$A (L_1 , L_2 , \ldots , L_N+1)$
. . .
$A (L_1 , L_2 , \ldots , U_N)$
$A (L_1 , L_2 , \ldots , L_{N-1}+1, L_N)$
. . .
$A (L_1 , L_2 , \ldots , L_{N-1} +1, L_N)$
. . .
$A (L_1 , L_2 , \ldots , U_{N-1} , U_N)$
. . .
$A (U_1 , L_2 , \ldots , L_{N-1} , L_N)$

$A(U_1, L_2, \ldots, L_{N-1}, L_N+1)$

...

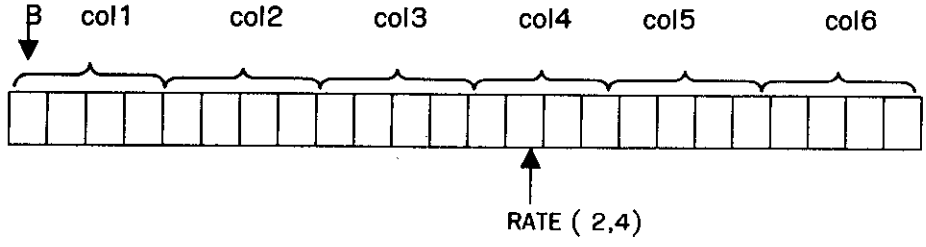
$A(U_1, L_2, \ldots, U_{N-1}, U_N)$

...

$A(U_1, U_2, \ldots, U_N)$

เรียงทีละสดมภ์ (Column – major order)

อีกทางเลือกหนึ่งสำหรับการทำให้เป็นเชิงเส้นของแถวลำดับสองมิติคือเก็บสมาชิกทีละสดมภ์ นั่นคือขั้นแรกเก็บสดมภ์ที่หนึ่ง จากนั้นเก็บสดมภ์ที่สอง จากนั้นสดมภ์ที่สามทำเช่นนี้เรื่อยไป



รูป 2-10 แถวลำดับ รูป 2-6 ทำให้เป็นเชิงเส้นเรียงทีละสดมภ์

โครงร่างการจัดสรรหน่วยเก็บวิธีนี้ใช้สำหรับแถวลำดับในทางปฏิบัติของ FORTRAN สมมติให้ B เป็นเลขที่อยู่ฐานของแถวลำดับ , สมาชิกแต่ละตัวของแถวลำดับใช้เนื้อที่ S ไบต์ เพราะฉะนั้นเลขที่อยู่ของ RATE (I , J) คือ

$$B + (J - 1) * 4 * S + (I - 1) * S$$

เพราะว่ามี J – 1 สดมภ์ แต่ละสดมภ์ความยาว 4 * S อยู่หน้าสดมภ์ซึ่งมีสมาชิก RATE (I , J) และมีสมาชิก I – 1 ตัวแต่ละตัวความยาว S อยู่หน้า RATE (I , J) ในสดมภ์ที่ J เพราะฉะนั้น RATE (2 , 4) อยู่ตำแหน่ง

$$B + (4 - 1) * 4 * S + (2 - 1) * S = B + 13 * S$$

กรณีทั่วไป ARRAY (I, J) ของแถวลำดับนิยามโดย

ARRAY ($L_1 : U_1 , L_2 : U_2$)

เก็บแบบทีละสดมภ์ สมาชิกตัวนี้จะอยู่ที่ตำแหน่ง

$$B + (J - L_2) * (U_1 - L_1 + 1) * S + (I - L_1) * S$$

แถวลำดับ N มิติ เก็บทีละสดมภ์ ตรรชนีล่างจะผันแปรจากซ้ายไปขวา (with an N-dimensional array , column-major order varies the subscripts in left-to-right order)

ตัวอย่าง แถวลำดับ A ($L_1 : U_1 , L_2 : U_2 , \ldots , L_N : U_N$) สมาชิกเก็บในหน่วยความจำตามลำดับดังนี้

A ($L_1 , L_2 , \ldots , L_N$)
A (L_1+1 , $L_2 , \ldots , L_N$)
. . .
A ($U_1 , L_2 , \ldots , L_N$)
A (L_1 , L_2+1 , ... , L_N)
A (L_1+1 , L_2+1 , ... , L_N)
. . .
A (U_1 , L_2+1 , ... , L_N)

. . .
A ($L_1 , L_2 , \ldots , L_N$+1)
A (L_1+1 , $L_2 , \ldots , L_N$+1)
. . .
A ($U_1 , L_2 , \ldots , L_N$ +1)
. . .
A ($L_1 , L_2 , \ldots , U_N$)
. . .
A ($L_1 , U_2 , \ldots , U_N$)
. . .
A ($U_1 , U_2 , \ldots , U_N$)

การเลือกเทคนิคการทำให้เป็นเชิงเส้น (Selecting a Linearization Technique)
เพื่อการตัดสินใจว่าการเก็บแถวลำดับทีละสดมภ์มีข้อดีมากกว่าการเก็บแถวลำดับทีละแถวอย่างไรนั้นจำเป็นต้องทราบว่าสมาชิกของแถวลำดับต้องเรียงอันดับอย่างไร ภาษาโปรแกรมที่มีให้ใช้ บางภาษาไม่ให้โปรแกรมเมอร์เลือกเทคนิคหน่วยเก็บแถวลำดับ

ตัวอย่าง จงพิจารณาโปรซีเดอร์คำนวณค่าเฉลี่ยของสมาชิกในแถวลำดับ A ขนาด 50X 225 เก็บทีละสดมภ์ (with column – major storage) ใน COBOL เขียนดังนี้

```
        COMPUTE TOTAL = 0.
        PERFORM SUM-UP VARYING J FROM 1 BY 1
                UNTIL J > 225 AFTER I FROM 1 BY 1
                UNTIL I > 50.
        COMPUTE AVERAGE = TOTAL / 11250.
        . . .
SUM-UP.
        COMPUTE TOTAL = TOTAL + A ( I, J).
```

อัลกอริทึมนี้บวกสมาชิกทุกตัวในหนึ่งสดมภ์เสร็จก่อนแล้วจึงประมวลผลสดมภ์ถัดไป
ภาษา Pascal เขียนดังนี้

```
        total := 0;
        for j := 1 to 225 do
        for I := 1 to 50 do
          total := total + a[i,j];
        average := total / 11250;
```

ในทางตรงกันข้าม หน่วยเก็บทีละแถว (with row – major storage) ซึ่งเป็นปกติสำหรับภาษา COBOL และ Pascal รูปแบบการเขียนอัลกอริทึมบวกสมาชิกทุกตัวในหนึ่งแถวก่อนแล้วจึงประมวลผลแถวถัดไปเขียนดังนี้

ใน COBOL

```
COMPUTE TOTAL = 0.
PERFORM SUM-UP VARYING I FROM 1 BY 1
        UNTIL I > 50 AFTER J FROM 1 BY 1
        UNTIL j > 225.
        . . .
SUM-UP.
        COMPUTE TOTAL = TOTAL + A( I,J).
```

ใน Pascal

```
total := 0;
for i := 1 to 50 do
for j := 1 to 225 do
  total := total + a[i,j];
```

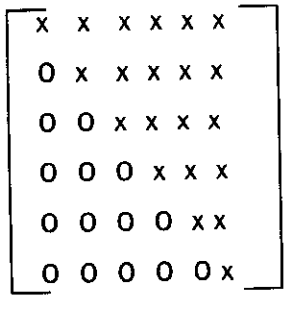
2.6 แถวลำดับแบบสามเหลี่ยม (Triangular Arrays)

ในหัวข้อที่แล้วได้อภิปรายการทำให้เป็นเชิงเส้นของแถวลำดับหลายมิติ ในหัวข้อนี้จะพิจารณาการทำให้เป็นเชิงเส้นของแถวลำดับชนิดพิเศษ : แถวลำดับสามเหลี่ยม

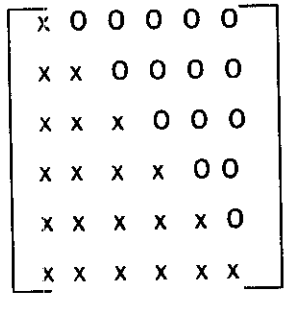
บทนิยาม

แถวลำดับแบบสามเหลี่ยมอาจเป็นสามเหลี่ยมบน (uper-triangular) ใน รูป 2-11 หรือสามเหลี่ยมล่าง (lower-triangular) รูป 2-12 เมื่อสมาชิกทั้งหมดตอนล่าง(หรือตอนบน) ของเส้นทแยงมุมหลักหลักเป็นศูนย์

แถวลำดับจะเรียกว่า strictly upper-(or lower-) triangular ถ้าสมาชิกทุกตัวบนเส้นทแยงมุมหลักเป็นศูนย์ด้วย



รูป 2-11 Upper-triangular array



รูป 2-12 Lower-triangular array

N × N

แถวลำดับแบบสามเหลี่ยมล่าง (lower-triangular array) ที่มี N แถว จำนวนสูงสุดของสมาชิกที่ไม่ใช่ศูนย์ (nonzero elements) ในแถวที่ I มี I ตัว
เพราะฉะนั้นจำนวนสมาชิกทั้งหมดที่ไม่ใช่ศูนย์ของแถวลำดับนี้จะไม่มากกว่า

$$\sum_{I=1}^{N} I = \frac{N(N+1)}{2}$$

นิพจน์นี้เป็นจริงเช่นกันสำหรับแถวลำดับแบบสามเหลี่ยมบน (upper-triangular) ที่มี N แถวสำหรับ N ที่มีขนาดใหญ่ จึงควรพิจารณาว่าไม่จำเป็นต้องเก็บสมาชิกทั้งหมดที่มีค่าเป็นศูนย์ (zero – valued elements)

การทำให้เป็นเชิงเส้น (Linearization)

วิธีหนึ่งของการศึกษาปัญหานี้คือการทำให้เป็นเชิงเส้นของแถวลำดับและเก็บเฉพาะสมาชิกของส่วนสามเหลี่ยมที่ไม่ใช่ศูนย์

ต้องการเก็บแถวลำดับแบบสามเหลี่ยมบนชื่อ T ทีละแถวในแถวลำดับหนึ่งมิติชื่อ S ที่มีดรรชนีล่างจาก 1 ถึง N (N+ 1) /2

สมาชิก T (1,1) เก็บเป็นสมาชิก S(1)
สมาชิก T (1,2) เก็บเป็นสมาชิก S(2)
. . .
สมาชิก T (1,N) เก็บเป็นสมาชิก S(N)
หลังจากนั้น
สมาชิก T (2,2) เก็บเป็นสมาชิก S(N+1) เพราะว่า T (2,1) เป็นศูนย์
สมาชิก T (N,N) เก็บเป็นสมาชิก S(N(N+1)/2
กรณีทั่วไป , T(I,J) จะเก็บใน S ตัวใด โปรดสังเกตว่า $I \leq j$ สำหรับสมาชิกทั้งหมดในส่วนไม่ใช่ O ของ T

(In general , where in S is T(I,J) stored ? Note that $I \leq J$ for all elements in the nonzero part of T.)

การใช้เนื้อที่ร่วมกัน (Sharing space)

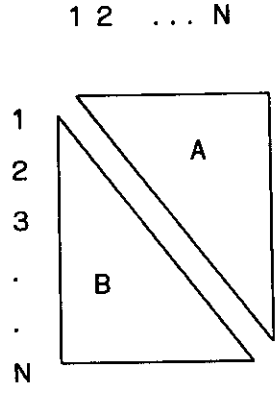
บางครั้งในโปรแกรมต้องใช้แถวลำดับแบบสามเหลี่ยมมากกว่าหนึ่งชุด ถ้าแถวลำดับสองชุดมีมิติเหมือนกัน ดังนั้นค่อนข้างจะเป็นงานตรงไปตรงมาเพื่อเก็บแถวลำดับในลักษณะประหยัดเนื้อที่สมมติว่าแถวลำดับ A เป็นแบบสามเหลี่ยมบนขนาด N × N และแถวลำดับ B เป็นแบบสามเหลี่ยมล่างขนาด (N-1) × (N-1) ดังนั้น A และ B สามารถเก็บด้วยกันในแถวลำดับ C ขนาด N×N ดังแสดงในรูป 2-13

C (I,J) เมื่อ I ≤ J คือสมาชิกของ A

และ C (I,J) เมื่อ I > J คือสมาชิกของ B

ที่จริง A (I,J) เก็บเป็น C (I,J) สำหรับ I ≤ J

ที่จริง B (I,J) เก็บเป็น C (I+1,J) สำหรับ I ≥ J

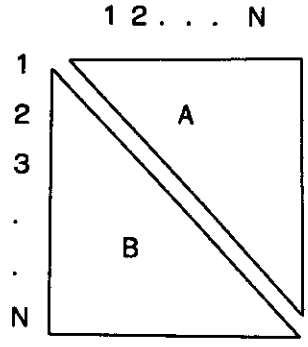


รูป 2-13 แถวลำดับแบบสามเหลี่ยมบนและแบบสามเหลี่ยมล่างใช้เนื้อที่ร่วมกันขนาด N × N

ให้แถวลำดับ A เป็นสามเหลี่ยมบน แถวลำดับ B เป็นสามเหลี่ยมล่างทั้งคู่มีขนาด N × N ดังนั้นแถวลำดับ C ต้องมี ขนาด N × (N+1) แสดงในรูป 2-14

ในที่นี้ A (I,J) เก็บเป็น C (I,J+1) สำหรับ I ≤ J

และ B (I,J) เก็บเป็น C (I,J) สำหรับ I ≥ J



รูป 2-14 แถวลำดับแบบสามเหลี่ยมบนและแบบสามเหลี่ยมล่างใช้เนื้อที่ร่วมกันขนาด N × (N+1)

ต่อไปพิจารณาแถวลำดับแบบสามเหลี่ยมบนสองชุดชื่อ A และ AA ขนาด N × N ให้ใช้เนื้อที่แถวลำดับร่วมกัน วิธีหนึ่งคือสลับเปลี่ยน (transpose) แถวลำดับแบบสามเหลี่ยมหนึ่งชุด ชื่อ AA คือ แทนที่จะเป็นแบบสามเหลี่ยมบน ให้เป็นสามเหลี่ยมล่างแทน นั่นคือ แถวลำดับ AA สลับเปลี่ยนเป็นแถวลำดับ AA^T สมาชิก AA(I,J) กลายเป็น AA^T (I,J) เพราะฉะนั้น A และ AA^T สามารถเก็บด้วยกันในแถวลำดับ C มีมิติเป็น N × (N+1) ดังแสดงในพารากราฟก่อนหน้านี้

A(I,J) เก็บเป็น C(I,J+1) และ AA(I,J) เก็บเป็น C(I,J)

2.7 แถวลำดับสพาช (Sparse Arrays)

แถวลำดับชนิดพิเศษอีกชนิดหนึ่งซึ่งเกิดขึ้นในการประยุกต์ คือ แถวลำดับสพาช แถวลำดับสพาชหมายถึง แถวลำดับซึ่งสมาชิกที่เป็นศูนย์มีความหนาแน่นค่อนข้างสูง(An array is called sparse if it has a relative high density of zero elements.)

ตัวอย่าง แถวลำดับขนาด 8X10 รูป 2-15 ซึ่งสมาชิกที่ไม่ใช่ศูนย์มี 8 ตัวจากสมาชิกทั้งหมดของแถวลำดับ 80 ตัวซึ่งเท่ากับมีสมาชิกที่เป็นศูนย์จำนวน 90 %

0	0	0	1	0	0	0	2	0	0
0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	4	0	0	0	0	0	0
0	0	0	0	0	0	0	2	0	0
0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0

รูป 2-15 ตัวอย่างแถวลำดับสพาซ

การทำให้เป็นเชิงเส้น (linearization)

แถวลำดับสพาซหลายมิติทำให้เป็นเชิงเส้นผ่านเทคนิคที่นำเสนอมาก่อนหน้านี้แล้ว แต่อย่างไรก็ตามมีเนื้อที่สูญเปล่ามาก (much wasted space)

ขณะนี้ให้พิจารณาการแทนที่อีกสองทางเลือกซึ่งจะเก็บเฉพาะสมาชิกที่ไม่ใช่ศูนย์เท่านั้น

การแทนที่แบบเวกเตอร์ (Vector Representation)

สมาชิกแต่ละตัวที่ไม่ใช่ศูนย์ในแถวลำดับสพาซสองมิติถูกแทนที่เป็นสามส่วน (triple) ด้วยรูปแบบ (ดรรชนีแถว , ดรรชนีสดมภ์ , ค่า)

ชุดสามส่วนเหล่านี้สามารถเรียงลำดับโดยดรรชนีแถวเป็นหลักและดรรชนีสดมภ์เป็นรอง เรียงจากน้อยไปหามาก และเก็บเป็นเวกเตอร์

ตัวอย่างเช่น ใช้วิธีนี้ แถวลำดับสพาซ 8X10 ในรูป 2-15 ถูกแทนที่โดยเวกเตอร์ V แสดงในรูป 2-16

	row	column	value
V(1):	1	5	1
V(2):	1	8	2
V(3):	2	2	1
V(4):	3	1	1
V(5):	5	4	4
V(6):	6	8	2
V(7):	8	1	2
V(8):	8	2	1

รูป 2-16 การแทนที่เวกเตอร์ของแถวลำดับสพาซของรูป 2-15

การแปลงส่งหน่วยเก็บนี้ใช้เนื้อที่มากกว่าเพื่อแทนสมาชิกที่ไม่ใช่ศูนย์แต่ไม่มีการแทนที่สมาชิกที่เป็นศูนย์ ถ้าแถวลำดับสพาซเป็นหนึ่งมิติ สมาชิกไม่ใช่ศูนย์แต่ละตัวจะถูกแทนที่เป็นคู่กรณีทั่วไป สมาชิกที่ไม่ใช่ศูนย์แต่ละตัวของแถวลำดับ N- มิติคือถูกแทนที่ด้วย entry N+1 ค่า

ตัวอย่าง ถ้าแถวลำดับข้างต้นมีการ update เพื่อให้สมาชิกที่มีดรรชนีล่าง (1,8) มีค่าเป็นศูนย์ ดังนั้นสมาชิกเวกเตอร์ V(3) จนถึง V(8) จะต้องย้ายไปเป็น V(2) ถึง V(7)
ในทำนองเดียวกันถ้าสมาชิกตัวที่มีดรรชนีล่าง (4,6)ปรับค่าเป็น 9 ดังนั้นเวกเตอร์ V(5) จนถึง V (8) จำเป็นต้องย้ายไปเป็น V(6) ถึง V(9) และ V(3) คือ triple (4,6,9)

การแทนที่แบบรายการโยง (Linked – list Representation)
การแทนที่อีกวิธีหนึ่งสำหรับแถวลำดับสพาซคือใช้รายการโยงซึ่งเป็นทางเลือกที่สำคัญจะอภิปรายรายละเอียดในบทที่ 6 เรื่องรายการโยง

แบบฝึกหัด

1. แถวลำดับที่มีสดมภ์เป็นตัวเลขจาก 4 ถึง 13 และมีเลขแถวจาก 6 ถึง 12 แถวลำดับชุดนี้เก็บสมาชิกได้มากที่สุดกี่ตัว
2. แถวลำดับ space (A:B, C:D) มีสมาชิกกี่ตัว
3. แถวลำดับ A (1: N) และแถวลำดับ B(-N : 0,3)
4. แถวลำดับ Test (1: 10 , 1:5) เก็บในหน่วยความจำทีละสดมภ์ ถ้าเลขที่อยู่ฐาน B=0 และขนาดสมาชิก S = 1 จงคำนวณเลขที่อยู่ของ Test (7,2)
5. กำหนดให้ ARRAY (10 : 30, 10 : 100) มีเลขที่อยู่ฐาน = 50 และขนาดสมาชิก = 8 ไบต์
   (a) ARRAY มีสมาชิกกี่ตัว
   (b) ถ้า ARRAY เก็บทีละแถว จงคำนวณหาเลขที่อยู่เริ่มต้นของ ARRAY (21,75) จงคำนวณหาเลขที่อยู่เริ่มต้นของ ARRAY (21,75)
6. กำหนดแถวลำดับ A (50 : 100, 50 : 75) จงคำนวณเลขที่อยู่เริ่มต้นของ A (62,56)
7. จงเขียนนิพจน์คำนวณเลขที่อยู่ของสมาชิกตัวที่ j ในแถวที่ i ของแถวลำดับ IVAN (B: A , D:C) เมื่อสมาชิกแต่ละตัวในแถวลำดับใช้เนื้อที่ n words ของหน่วยความจำสมมติว่า ตำแหน่งเริ่มต้นคือ 0
8. จงพิจารณาแถวลำดับ Q (A:B , C:D , E:F) เลขที่อยู่ฐาน = x , ความยาวของสมาชิก= L
   (a) จงคำนวณหาจำนวนสมาชิกทั้งหมดของแถวลำดับ Q
   (b) จงคำนวณหาตำแหน่งเริ่มต้นของ Q (i,j,k) ทั้งนี้แถวลำดับถูกทำให้เป็นเชิงเส้นและเก็บโดยวิธีอะไร
9. จงเขียนอัลกอริทึมสลับเปลี่ยน (transpose) แถวลำดับ
10. จงเขียน transpose สำหรับแถวลำดับสพาซข้างล่างนี้เมื่อเก็บแถวลำดับเป็นเวกเตอร์ ของชุดสามส่วน (ดรรชนีแถว , ดรรชนีสดมภ์ , ค่า)

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 9 & 0 & 1 \\ 0 & 16 & 0 & 0 & 0 & 0 \end{bmatrix}$$

11. จงพิจารณาแถวลำดับ Q (-1:1 , 1: 2 , 1:3)

(a) Q มีสมาชิกกี่ตัว

(b) ถ้าเก็บ Q ทีละแถวจงเขียนรายชื่อลำดับสมาชิกของ Q

(c) ถ้าเก็บ Q ทีละสดมภ์จงเขียนรายชื่อลำดับสมาชิก

ทุกกรณีวาดรูปประกอบด้วย