Chapter 2 Array

ผศ.ดร.สิลดา อินทรโสธรฉันท์

สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยขอนแก่น

2

รูปแบบการจัดเก็บข้อมูล

- โครงสร้างที่ใช้การจัดสรรหน่วยความจำแบบลำดับ (Sequential Allocation)

วิธีการจัดสรรแบบนี้จะเหมาะกับข้อมูลที่ทราบปริมาณข้อมูลที่ค่อนข้างชัดเจน เนื่องจากต้องกำหนดขนาดข้อมูลทั้งหมดไว้ล่วงหน้า เนื่องจากต้องกำหนดขนาดข้อมูลทั้งหมดไว้ล่วงหน้า ทำให้ข้อมูลถูกจัดเก็บต่อเนื่องกันเป็นกลุ่ม ทำให้สามารถเข้าถึงได้ง่ายและรวดเร็ว โครงสร้างที่ใช้การจัดสรรหน่วยความจำแบบลำดับ คือ อาเรย์

รูปแบบการจัดเก็บข้อมูล

- โครงสร้างที่ใช้การจัดสรรหน่วยความจำแบบพลวัต (Dynamic Allocation)

ทุกครั้งที่มีข้อมูลใหม่เพิ่มเข้ามา จะมีการจัดสรรหน่วยความจำที่มีปริมาณมากพอสำหรับข้อมูลใหม่ วิธีการจัดสรรแบบนี้จะเหมาะกับข้อมูลที่ไม่ทราบปริมาณแน่ชัด อาจปรับเปลี่ยนได้ตลอดเวลา โครงสร้างที่ใช้การจัดสรรหน่วยความจำแบบพลวัต คือ ลิงก์ลิสต์แบบต่าง ๆ

โครงสร้างอาเรย์

- อาเรย์เป็นโครงสร้างข้อมูลที่ใช้ในการเก็บข้อมูลประเภทเดียวกันหลาย ๆ ค่าไว้เป็นกลุ่มเดียวกัน
- เป็นการจองเนื้อที่ในหน่วยความจำแบบต่อเนื่องตามขนาดของชนิดข้อมูลและจำนวนสมาชิกที่ระบุไว้
- อาเรย์ไม่ว่าจะเป็น 1 มิติ 2 มิติ หรือหลายมิติ ก็จะถูกนำมาเก็บในหน่วยความจำต่อเนื่องกันในรูปแบบ 1 มิติ โดยตำแหน่งในการเก็บข้อมูลในหน่วยความจำจะคำนวณจากขนาดของข้อมูลแต่ละตัวและตำแหน่งข้อมูลในอาเรย์

โครงสร้างอาเรย์

- สำหรับการระบุตำแหน่งสมาชิกแต่ละตัวในอาเรย์จะใช้เลขดัชนี (index) กำกับ
- โดยอาเรย์ 1 มิติ จะใช้เลขดัชนี 1 จำนวนในการระบุตำแหน่ง
- หากเป็นอาเรย์ 2 มิติจะใช้เลขดัชนี 2 จำนวนสำหรับระบุแถวและคอลัมน์ที่ข้อมูลอยู่
- ซึ่งการอ้างอิงถึงสมาชิกแต่ละตัวสามารถเขียนได้หลายรูปแบบ ที่นิยมใช้กันจะมีดังนี้

โครงสร้างอาเรย์

- แบบที่ 1 เขียนเลขดัชนีห้อยไว้ด้านล่างของชื่ออาเรย์ เช่น A1, A2, A3, ..., An
- แบบที่ 2 เขียนเลขดัชนีไว้ในเครื่องหมาย “( )” ต่อท้ายชื่ออาเรย์ เช่น A(1), A(2), A(3), ..., A(n)
- แบบที่ 3 เขียนเลขดัชนีไว้ในเครื่องหมาย “[ ]” ต่อท้ายชื่ออาเรย์ เช่น on A[1], A[2], A[3], ..., A[n]

ในที่นี้จะเลือกใช้การเขียนแบบที่ 3 เพื่อให้สอดคล้องกับการเขียนโปรแกรมภาษาจาวา

โครงสร้างอาเรย์

อาเรย์ A

ตำแหน่ง	A[0]	A[1]	A[2]	A[3]	A[4]
ข้อมูล	S	T	A	M	P

- หากต้องการตำแหน่งที่เก็บค่า T ก็จะเป็นตำแหน่ง A [1]
- หากอ้างถึงค่าในตำแหน่ง A[0] ก็จะได้ค่า S กลับมา

โครงสร้างอาเรย์

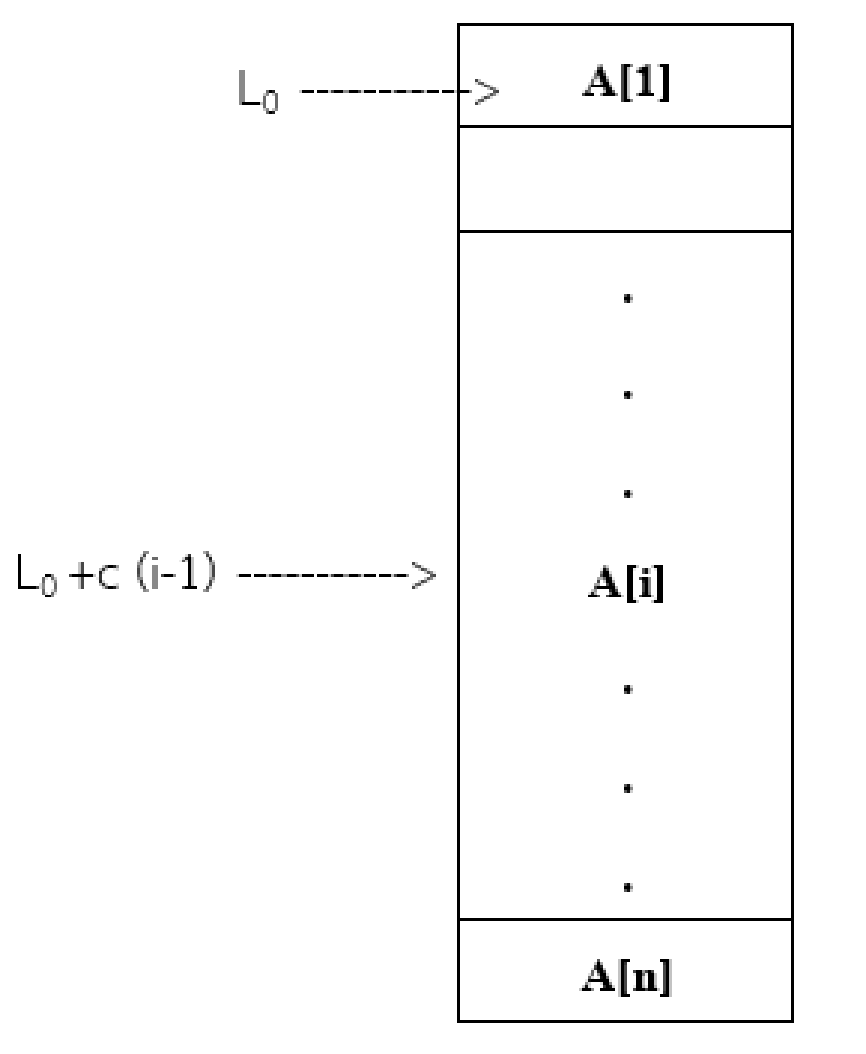
อาเรย์ B

ตำแหน่ง	B[0][0]	B[0][1]	B[0][2]	B[0][3]	B[0][4]
ข้อมูล	10	15	8	18	12
ตำแหน่ง	B[1][0]	B[1][1]	B[1][2]	B[1][3]	B[1][4]
ข้อมูล	7	9	16	13	11

- หากอ้างถึงตำแหน่ง B[0][2] จะได้ค่า 8 กลับมา
- หากต้องการตำแหน่งที่เก็บค่า 16 ก็จะเป็นตำแหน่ง B[1][2]

โครงสร้างอาเรย์

- การเก็บข้อมูลด้วยโครงสร้างอาเรย์จะเป็นการจองเนื้อที่เก็บข้อมูลให้สมาชิกแต่ละตัวแบบต่อเนื่องกัน
- ถ้าสมาชิกแต่ละตัวของอาเรย์ใช้เนื้อที่ในการจัดเก็บ c บิท มีจำนวนสมาชิก n ตัว เนื้อที่ในหน่วยความจำที่จองไว้จะเป็น c * n บิทต่อเนื่องกัน

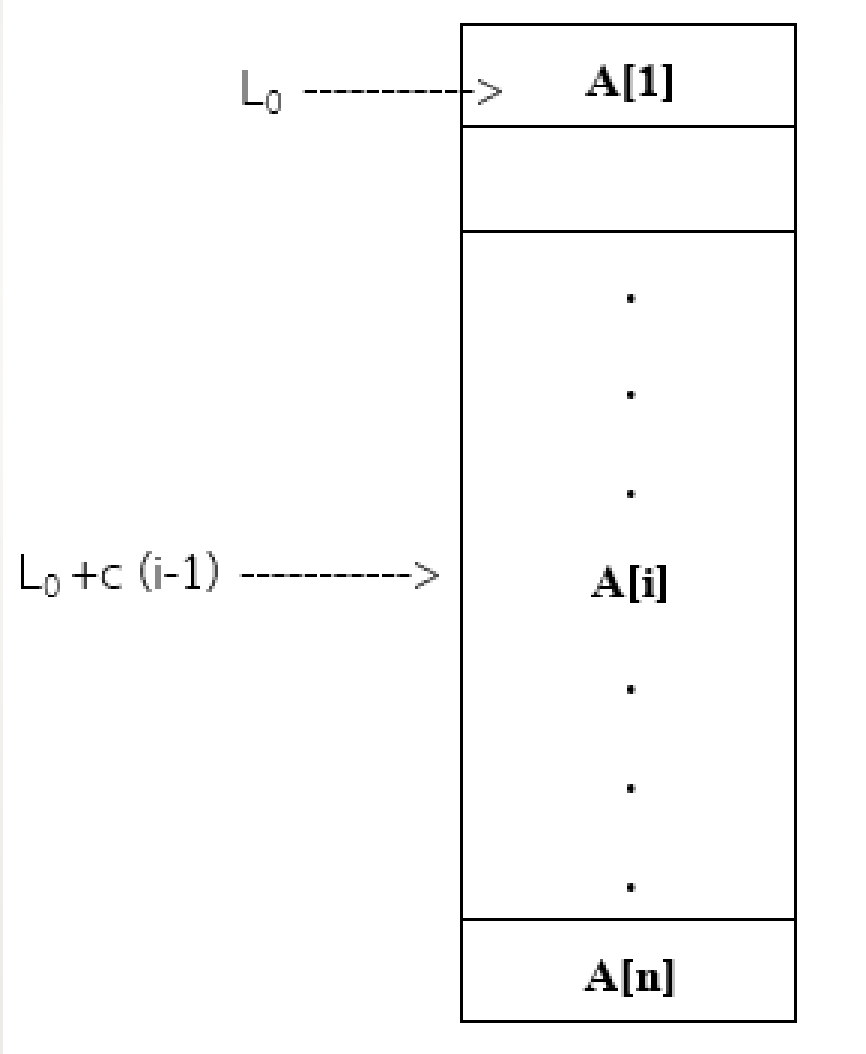


โครงสร้างอาเรย์

ตำแหน่งที่อยู่ของ Ai คำนวณได้จาก

$$LOC (A[i]) = L_0 + c * (i - 1)$$

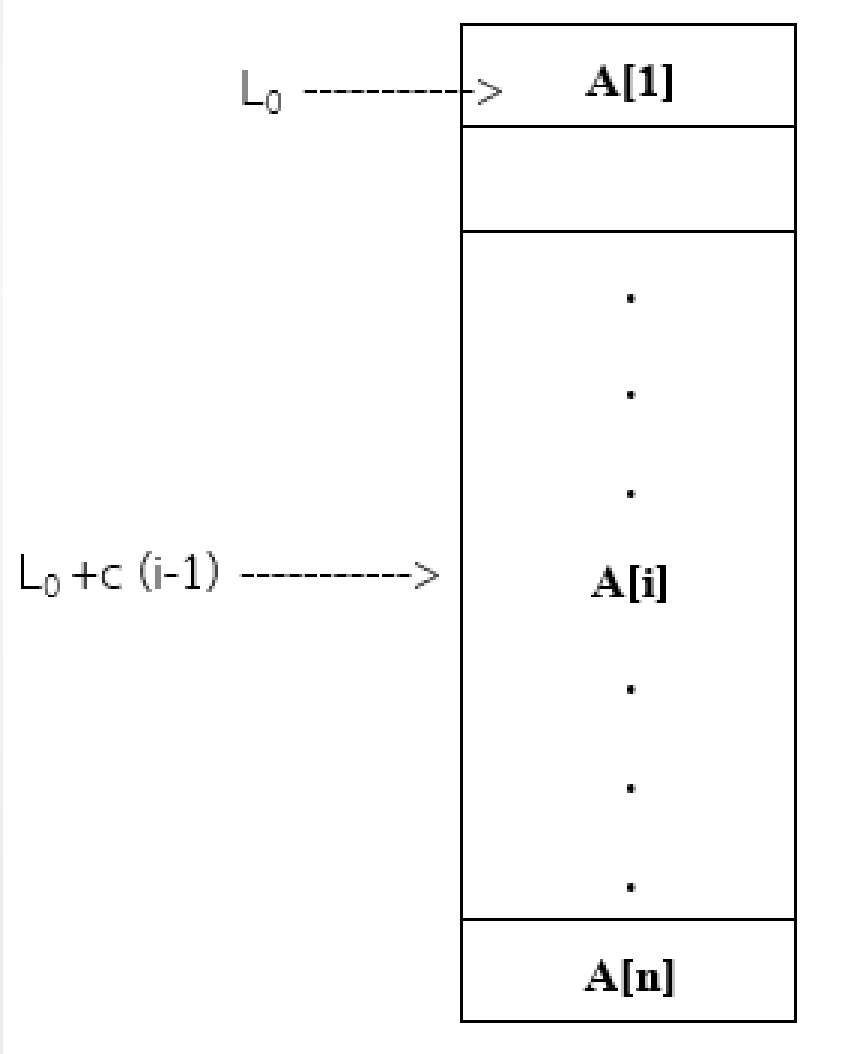
- L_0 แทนค่าตำแหน่งที่อยู่ของบิทแรกซึ่งจัดสรรให้สมาชิกตัวแรกของ A
- c แทนจำนวนบิทที่จัดสรรให้แก่แต่ละสมาชิก



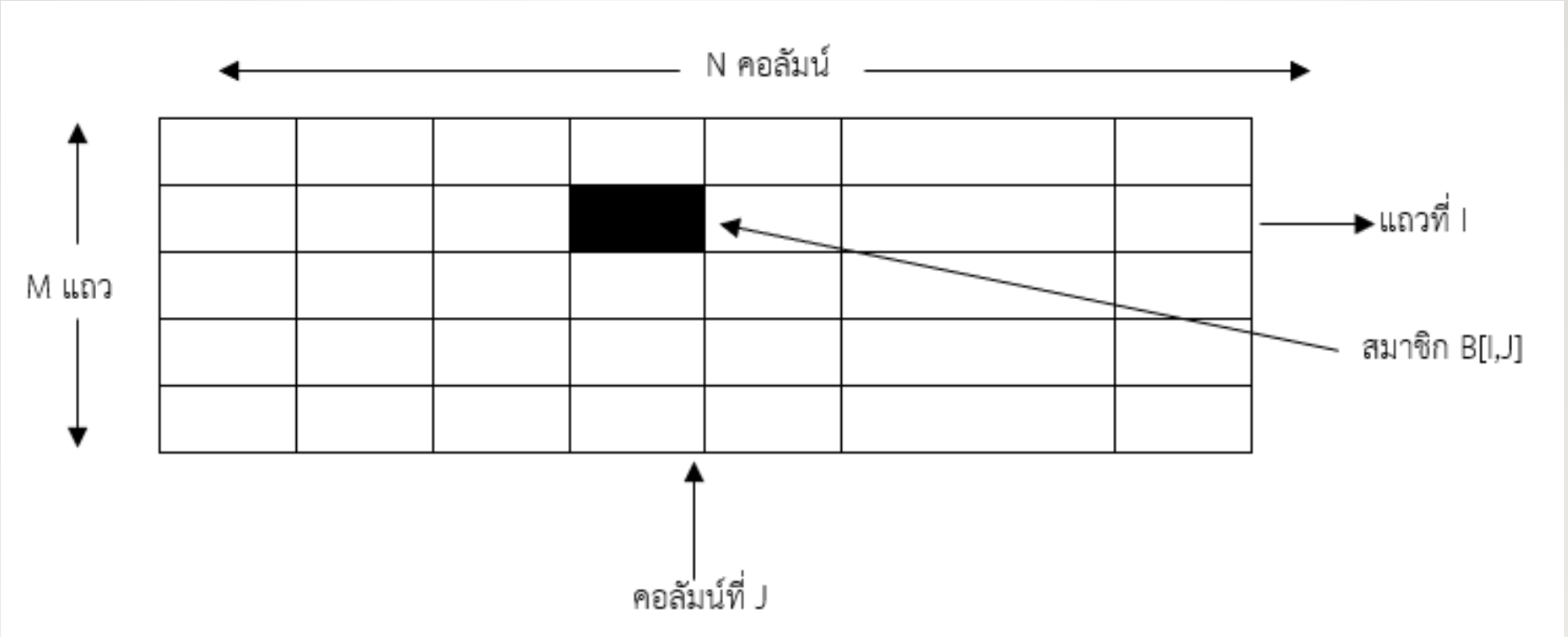
โครงสร้างอาเรย์

ในกรณีทั่ว ๆ ไป ถ้าตำแหน่งที่ทราบค่า (L_0) มีค่าดัชนีเป็นตัวแปร b จะสามารถหาตำแหน่งที่อยู่ของ A[i] จาก

$$LOC (A[i]) = L_0 + c * (i - b)$$



โครงสร้างอาเรย์

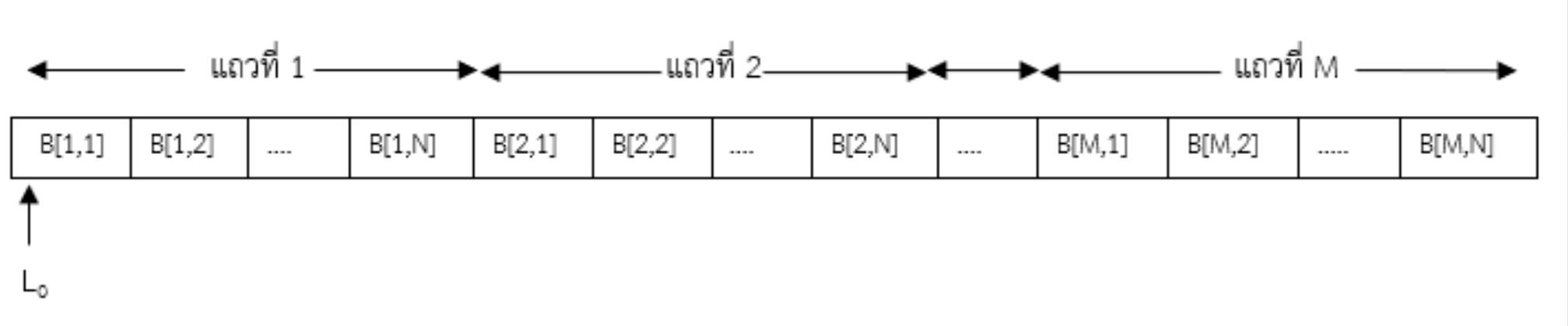


โครงสร้างอาเรย์

- กรณีอาเรย์ 2 มิติ เช่น B[I,J] โดย 1 ≤ I ≤ M และ 1 ≤ J ≤ N
- ถ้า c เป็นจำนวนบิทของสมาชิกแต่ละตัวจะต้องใช้หน่วยความจำทั้งหมดในการเก็บอาเรย์ 2 มิตินี้เป็นจำนวน c * M * N บิท
- ในการจัดเก็บอาเรย์ 2 มิติลงในหน่วยความจำจะเป็นการเก็บข้อมูลในรูปแบบ 1 มิติโดยนำค่าในแต่ละแถว/คอลัมน์มาวางต่อเนื่องกันขึ้นอยู่กับรูปแบบการจัดเรียง โดยมี 2 รูปแบบดังนี้

โครงสร้างอาเรย์

1) เรียงลำดับแบบแถวเป็นหลัก (Row Major or Lexicographic Order of Indices) โดยการนำสมาชิกจากอาเรย์ 2 มิติ มาบรรจุลงในอาเรย์ 1 มิติ ทีละแถวดังนี้



โครงสร้างอาเรย์

- ถ้า L_0 เป็นตำแหน่งที่อยู่บิทแรกของอาเรย์ 1 มิติ จะสามารถหาตำแหน่งที่อยู่ของสมาชิก B[I,J] ได้ดังนี้
- <u>หาตำแหน่งสมาชิกในแต่ละแถว</u> ให้ Y เป็นตำแหน่งที่อยู่ของสมาชิกแรกในแถวที่ I จะสามารถหาตำแหน่งที่อยู่ของสมาชิกตัวที่ J ในแถว ได้จากสูตร

$$LOC (B[I,J]) = Y + c * (J - 1)$$

โครงสร้างอาเรย์

- หาตำแหน่งแรกของแถวที่ I (Y) จะพิจารณาจากจำนวนบิทที่ใช้ในการเก็บข้อมูลทั้งหมดตั้งแต่แถวที่ 1 ถึง แถวที่ I – 1 ซึ่งแต่ละแถวจะใช้จำนวนบิทเป็น c * N

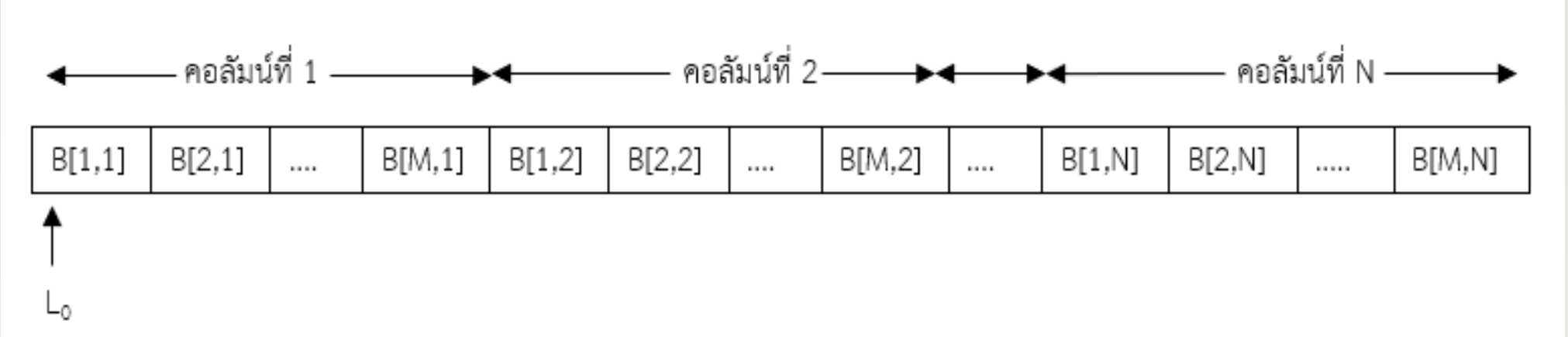
$$Y = L_0 + (I - 1) * c * N$$

- นำค่า Y แทนลงในสมการจะได้สมการในการหาตำแหน่งของ B[I,J] โดยการจัดเรียงแบบแถวเป็นหลัก ดังนี้

$$LOC(B [I , J]) = L_0 + (I - 1) * c * N \quad + \quad c * (J - 1)$$

โครงสร้างอาเรย์

2) เรียงลำดับแบบคอลัมน์เป็นหลัก (Column Major Order) โดยการนำสมาชิกจากอาเรย์ 2 มิติ มาบรรจุลงในอาเรย์ 1 มิติ ทีละคอลัมน์ดังนี้



โครงสร้างอาเรย์

- ถ้า L_0 เป็นตำแหน่งที่อยู่บิทแรกของอาเรย์ 1 มิติ จะสามารถหาตำแหน่งที่อยู่ของสมาชิก B[I,J] ได้ดังนี้
- หาตำแหน่งสมาชิกในแต่ละคอลัมน์ ให้ Y เป็นตำแหน่งที่อยู่ของสมาชิกแรกในคอลัมน์ที่ J จะสามารถหาตำแหน่งที่อยู่ของสมาชิกตัวที่ I ในคอลัมน์ ได้จากสูตร

$$LOC (B[I,J]) = Y + c * (I - 1)$$

โครงสร้างอาเรย์

- <u>หาตำแหน่งแรกของคอลัมน์ที่ J</u> (Y) จะพิจารณาจากจำนวนบิทที่ใช้ในการเก็บข้อมูลทั้งหมดตั้งแต่คอลัมน์ที่ 1 ถึง คอลัมน์ที่ J – 1 ซึ่งแต่ละคอลัมน์จะใช้จำนวนบิทเป็น c * M

$$Y = L_0 + (J - 1) * c * M$$

- นำค่า Y แทนลงในสมการจะได้สมการในการหาตำแหน่งของ B[I,J] โดยการจัดเรียงแบบคอลัมน์เป็นหลักดังนี้

$$LOC (B [I , J]) = L_0 + (J - 1) * c * M + c * (I - 1)$$

ข้อสังเกต

- ชนิดข้อมูลและจำนวนสมาชิกในอาเรย์จะมีผลโดยตรงต่อเนื้อที่ที่ใช้ในหน่วยความจำ
- หากใช้ชนิดข้อมูลที่มีขนาดใหญ่ (ใช้เนื้อที่ในการเก็บสมาชิกแต่ละจำนวนมาก) แต่เก็บข้อมูลขนาดเล็ก ก็จะทำให้สิ้นเปลืองเนื้อที่ไปโดยเปล่าประโยชน์
- การระบุจำนวนสมาชิกในอาเรย์ที่มากเกินไปแต่ไม่ได้ใช้ก็เช่นเดียวกัน
- ดังนั้น ควรเลือกใช้ชนิดข้อมูลและจำนวนสมาชิกให้เหมาะสมกับพฤติกรรมของข้อมูลที่จะเก็บ

การสร้างอาเรย์ 1 มิติ

```
dataType  [ ] arrayName = new dataType [arraySize] ;
```

หรือ

```
dataType  arrayName[ ] = new dataType [arraySize] ;
```

- ตัวอย่างเช่น

```
String stdName[] = new String [10] ;   // เก็บข้อมูลชนิด String จำนวน 10 ตัว

double grade [] = new double [10];   // เก็บข้อมูลชนิด double จำนวน 10 ตัว
```

การสร้างอาเรย์ 2 มิติ

```
dataType [ ][ ] arrayName = new dataType [row][col];
```

หรือ

```
dataType arrayName [ ][ ] = new dataType [row][col];
```

ตัวอย่างเช่น

```
int matrixA[][] = new int [5][4] ;
```

- อาเรย์ matrixA ที่สร้างขึ้น จะมีชนิดข้อมูลเป็น int และมีจำนวนข้อมูลเป็น 5 แถว x 4 คอลัมน์

การกำหนดค่าให้กับอาเรย์

การกำหนดค่าให้กับตัวแปรสามารถทำได้หลายแบบ ขึ้นอยู่กับลักษณะการใช้งาน

แบบที่ 1 สร้างพร้อมกับกำหนดค่าให้กับสมาชิกแต่ละตัว เช่น

```
String grade [ ] = {“A”, “B”, “C”, “D”, “F”} ;

int [ ][ ] m = { {1,2,3,4,5}, {6,7,8,9,10}, {11,12,13,14,15}};
```

- การสร้างอาเรย์พร้อมกับทำการกำหนดค่าให้กับสมาชิกแต่ละตัว ลักษณะนี้ไม่ต้องทำการระบุขนาด
- การกำหนดค่าอาเรย์ 2 มิติ จะต้องใส่เครื่องหมาย { } ครอบข้อมูลในแต่ละแถวเอาไว้ ซึ่งข้อมูลจะต้องเท่ากันทุกแถวและเป็นชนิดเดียวกัน

การกำหนดค่าให้กับอาเรย์

แบบที่ 2 กรณีต้องการกำหนดค่าให้สมาชิกทุกตัวมีค่าเดียวกัน สามารถใช้เมธอด fill ของคลาส Arrays ซึ่งคำสั่งนี้จะใช้ได้เฉพาะกับอาเรย์ 1 มิติเท่านั้น

- โดยในการใช้งานเมธอด fill ต้องทำการ import java.util.Arrays; หรืออาจใช้ import java.util.*;
- การใช้งานจะต้องทำการสร้างอาเรย์ไว้ก่อนแล้วจึงเรียกใช้เมธอด fill ตัวอย่างเช่น

```
int[] a = new int[5];

Arrays.fill (a,15);
```

การกำหนดค่าให้กับอาเรย์

แบบที่ 3 ต้องการกำหนดค่าใส่สมาชิกแต่ละตัวโดยการรับค่าทางคีย์บอร์ด

```
Scanner kb = new Scanner(System.in);
int size = 10;       // กำหนดขนาดอาเรย์เป็น 10
int num [] = new int [size];
for(int i = 0 ; i < size ; i++){
        num [i] = kb.nextInt();
}
```

การกำหนดค่าให้กับอาเรย์

```
Scanner kb = new Scanner(System.in);
int row = 5 , col = 4 ;
int num [][] = new int [row][col];
for(int i = 0 ; i < row ; i++){
        for(int j = 0 ; j < col ; j++){
            num [i][j] = kb.nextInt();
       }
}
```

การดำเนินการเกี่ยวกับอาเรย์

- การเข้าถึงสมาชิกแต่ละตัวในอาเรย์
- การแทรกข้อมูล
- การลบข้อมูล

การเข้าถึงสมาชิกแต่ละตัวในอาเรย์

- กำหนดให้ A เป็นอาเรย์ที่มีดัชนีเริ่มต้นเป็น LB และดัชนีสูงสุดเป็น UB
- ในการเดินทางบนอาเรย์ K จะทำการอ้างอิงถึงดัชนีตั้งแต่ LB จนถึง UB ซึ่งมีขั้นตอนการทำงานดังนี้

1. Repeat for K = LB to UB

   Apply PROCESS to A[K]

   [End of Loop]

2. Exit

การแทรกข้อมูล

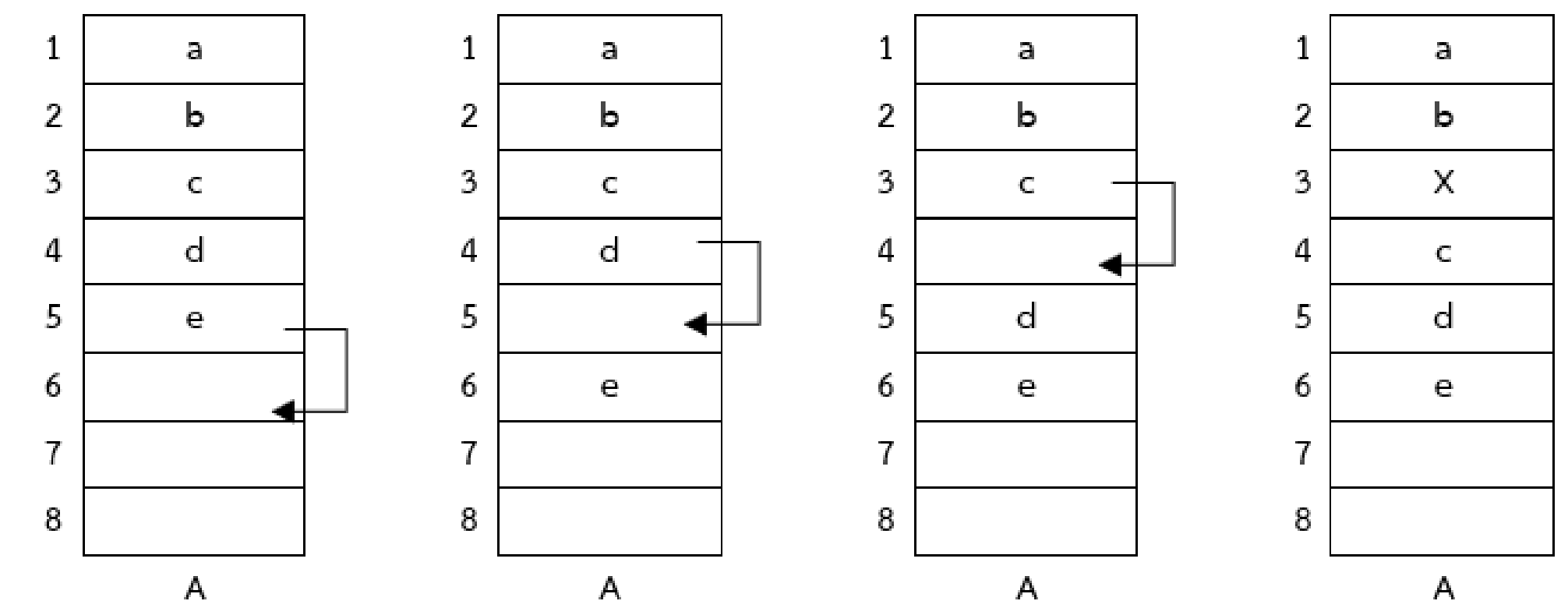
- การแทรก/เพิ่มข้อมูลลงในอาเรย์ หากตำแหน่งที่ต้องการใส่ข้อมูลใหม่ลงไปยังไม่มีข้อมูลเดิมอยู่ก็สามารถใส่ข้อมูลลงตำแหน่งนั้นได้โดยไม่ทำให้ข้อมูลเดิมสูญหาย แต่ถ้าตำแหน่งที่ต้องการแทรกมีข้อมูลเดิมอยู่แล้วหากนำข้อมูลใหม่ใส่ลงไปทันทีย่อมทำให้ข้อมูลเดิมหายไป
- ดังนั้นในการแทรกข้อมูลใหม่ลงในตำแหน่งที่มีข้อมูลเดิมอยู่จะต้องมีการเตรียมอาเรย์เพื่อให้ตำแหน่งดังกล่าวสามารถใส่ข้อมูลใหม่ลงไป โดยที่ข้อมูลเดิมทั้งหมดยังคงอยู่

การแทรกข้อมูล

- การเตรียมอาเรย์ก่อนทำการแทรกข้อมูลสามารถทำได้หลายแบบ ในที่นี้ขอยกตัวอย่าง 2 วิธีที่นิยมใช้

แบบที่ 1 จะทำการขยับข้อมูลเดิมที่อยู่ตั้งแต่ตำแหน่งที่ต้องการนำข้อมูลใหม่ใส่ลงไปจนถึงตำแหน่งสุดท้ายไป 1 ตำแหน่ง เพื่อเตรียมตำแหน่งสำหรับการใส่ข้อมูลใหม่ลงไป ดังตัวอย่าง

การแทรกข้อมูล



การแทรกข้อมูล

แบบที่ 1

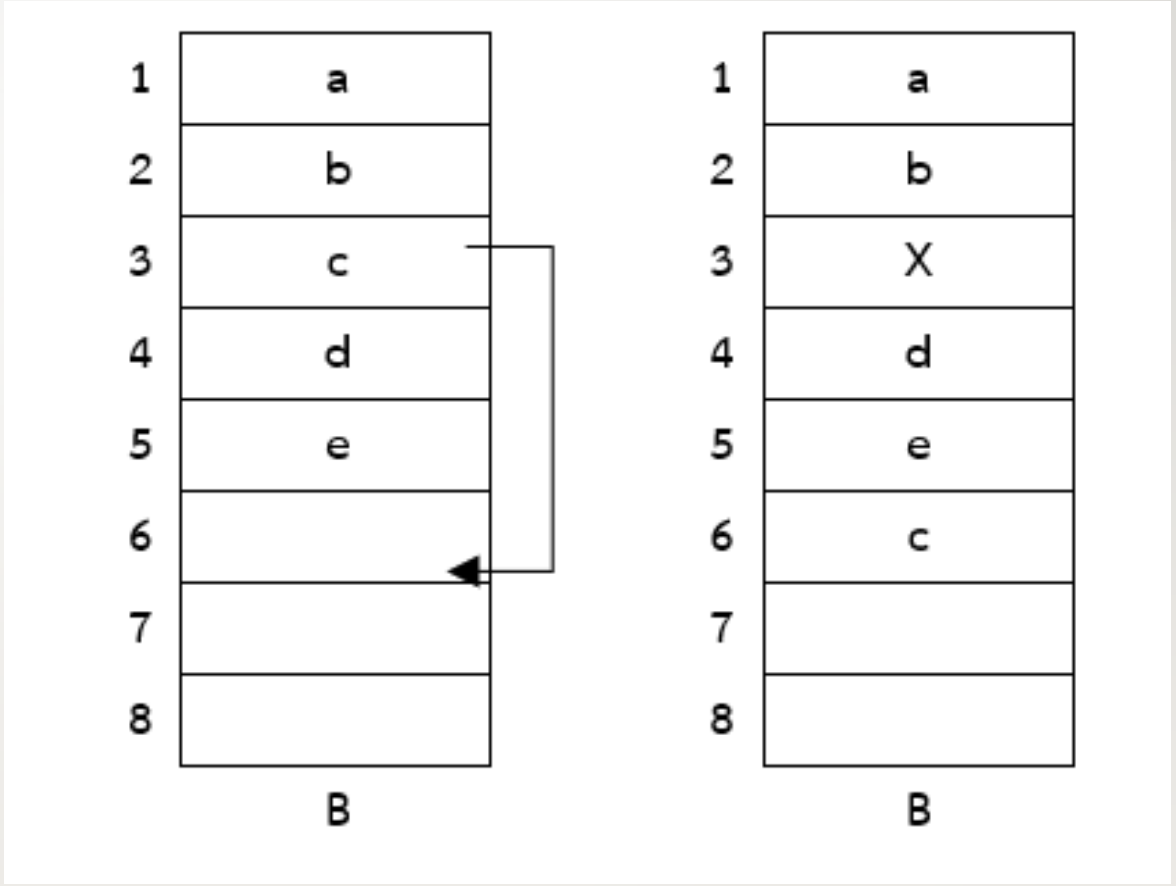
- ให้ A เป็นอาเรย์ที่มีข้อมูลอยู่ N ตำแหน่ง
- ตัวแปร ITEM จะทำการเก็บข้อมูลที่ต้องการแทรกลงไปในตำแหน่ง K
- โดยที่ K ≤ N อัลกอริทึมสำหรับแทรกข้อมูลเป็นดังนี้

INSERT1 (A, N, K, ITEM)

1. [Initialize counter] Set J := N
2. Repeat step 3 and 4 while J ≥ K
3. [Move the J^{th} element downward]
   Set A[J+1] := A[J]
4. [Decrease counter] Set J := J – 1
5. [Insert element] Set A[K] := ITEM
6. [Reset N] Set N := N + 1
7. Exit

การแทรกข้อมูล

แบบที่ 2 จะทำการคัดลอกข้อมูลเดิมในตำแหน่งที่ต้องการใส่ข้อมูลใหม่ไปไว้ในตำแหน่งท้ายสุด แล้วจึงนำข้อมูลใหม่ใส่ลงไป ดังตัวอย่าง



การแทรกข้อมูล

แบบที่ 2

- ให้ B เป็นอาเรย์ที่มีข้อมูลอยู่ N ตำแหน่ง
- ตัวแปร ITEM จะทำการเก็บข้อมูลที่ต้องการแทรกลงไปในตำแหน่ง K
- โดยที่ K ≤ N อัลกอริทึมสำหรับแทรกข้อมูลเป็นดังนี้

INSERT2 (B, N, K, ITEM)

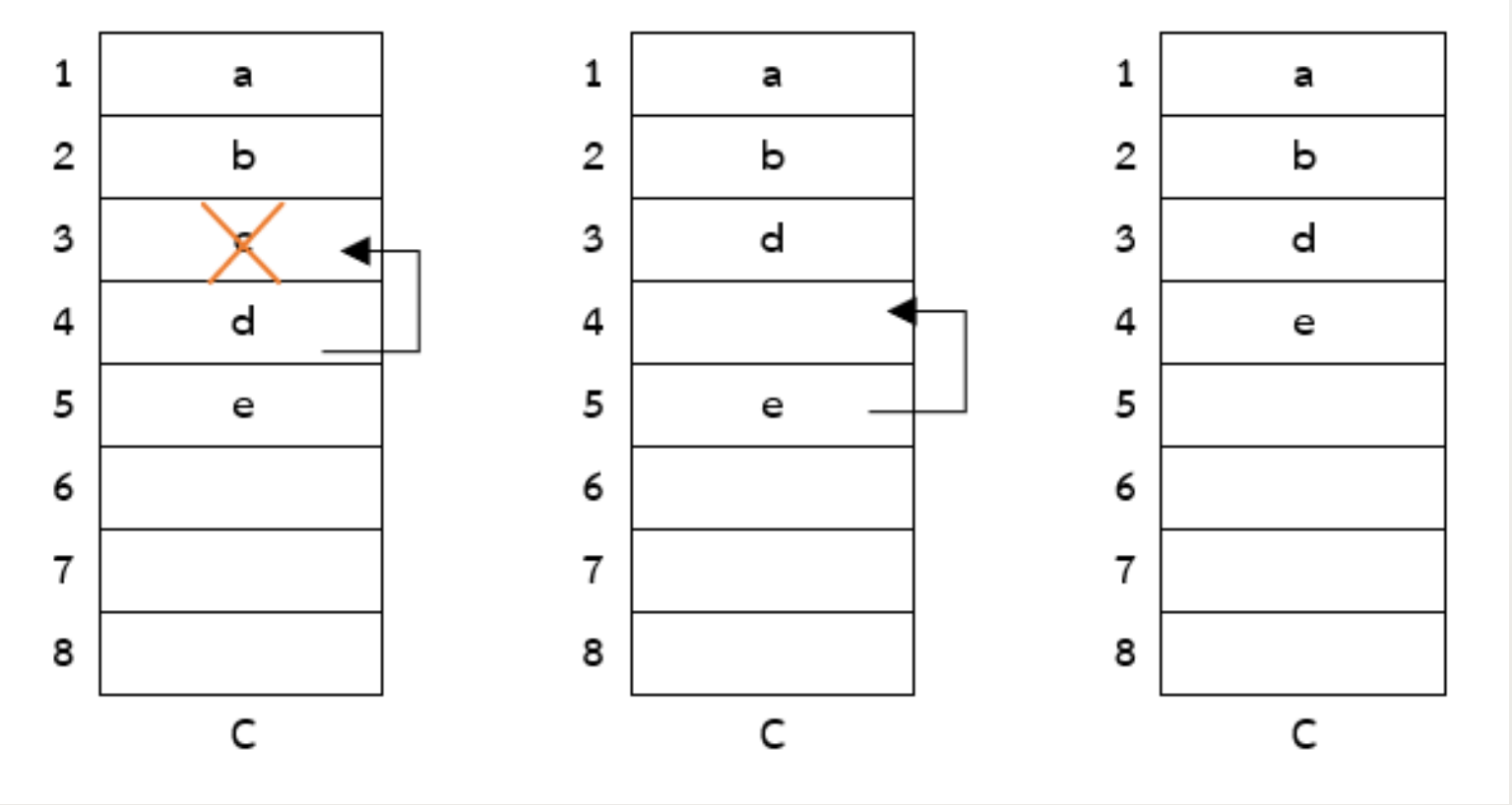
1. [Move the K^{th} element]
   Set B[N+1] := B[K]
2. [Insert element]
   Set B[K] := ITEM
3. [Reset N] Set N := N + 1
4. Exit

การลบข้อมูล

- การลบข้อมูลจากอาเรย์จะเป็นการคัดลอกค่าสมาชิกที่มีอยู่เดิมมาแทนที่ แล้วลดจำนวนข้อมูลที่เก็บไว้ในอาเรย์ลง 1 ค่า
- โดยในการคัดลอกค่าที่มีอยู่เดิมมาแทนที่สามารถทำได้หลายแบบ ในที่นี้ขอยกตัวอย่าง 2 วิธีที่นิยมใช้

แบบที่ 1 จะทำการขยับข้อมูลเดิมที่อยู่ตั้งแต่หลังตำแหน่งที่ต้องการลบข้อมูลจนถึงตำแหน่งสุดท้าย ลดลง 1 ตำแหน่ง ดังตัวอย่าง

การลบข้อมูล



การลบข้อมูล

แบบที่ 1

- ให้ C เป็นอาเรย์ที่มีข้อมูลอยู่ N ตำแหน่ง
- K เป็นตำแหน่งที่ต้องการลบข้อมูล
- โดยที่ K ≤ N อัลกอริทึมสำหรับลบข้อมูลเป็นดังนี้

DELETE1 (C, N, K, ITEM)

1. Set ITEM := C[K]
2. Repeat for J = K to N – 1 :

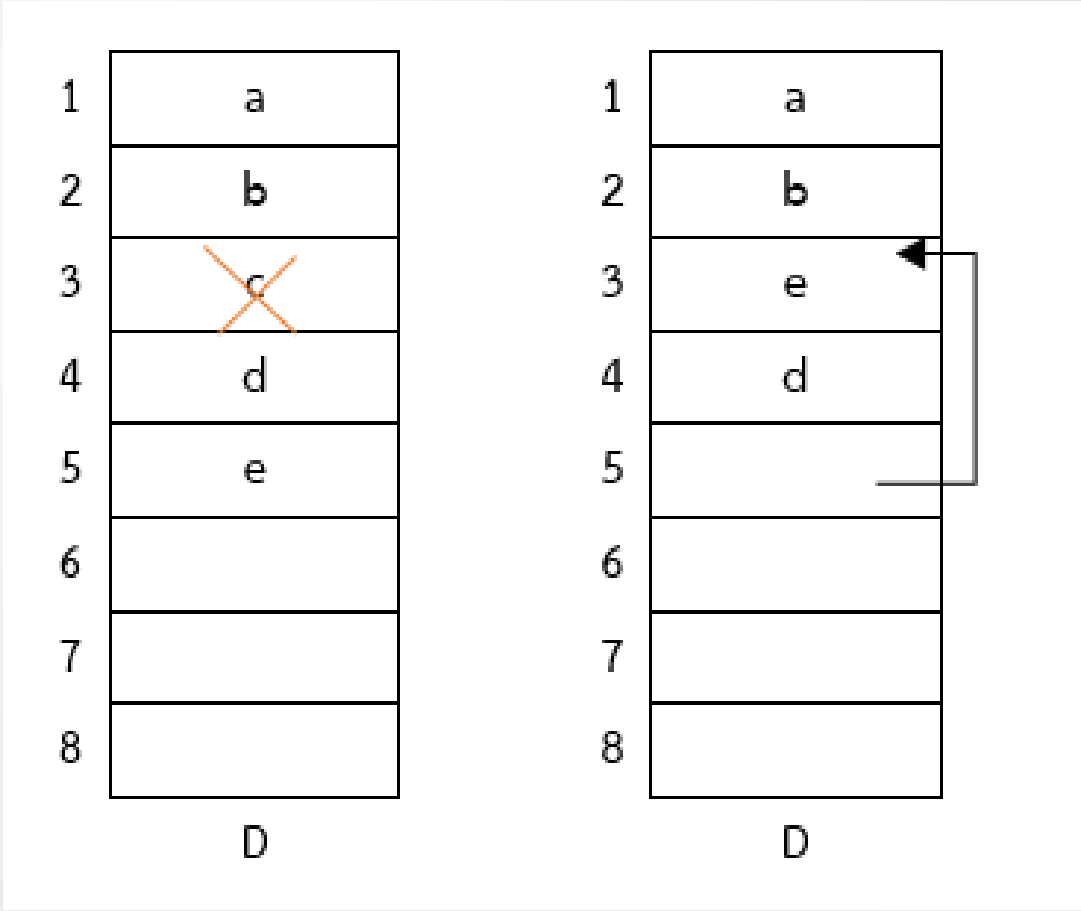
   [Move the J + 1st element upward]

   Set C[J] := C[J + 1]
3. [Reset the number N of elements]

   Set N := N – 1
4. Exit

การลบข้อมูล

<u>แบบที่ 2</u> จะทำการคัดลอกข้อมูลเดิมในตำแหน่งสุดท้ายมาไว้แทนที่ตำแหน่งที่ลบ ดังตัวอย่าง



การลบข้อมูล

แบบที่ 2

- ให้ D เป็นอาเรย์ที่มีข้อมูลอยู่ N ตำแหน่ง
- K เป็นตำแหน่งที่ต้องการลบข้อมูล
- โดยที่ K ≤ N อัลกอริทึมสำหรับลบข้อมูลเป็นดังนี้

DELETE2 (D, N, K, ITEM)

1. Set ITEM := D[K]
2. [Move the N^{st} element]
   Set D[K] := D[N]
3. [Reset the number N of elements]
   Set N := N – 1
4. Exit