การเขียนโปรแกรมคอมพิวเตอร์ 2

Computer Programming II

การเข้าถึงข้อมูลในอาเรย์

Data Access in Array

ภิญโญ แท้ประสาทสิทธิ์

ภาควิชาคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร

(taeprasartsit_p at silpakorn dot edu, pinyotae at gmail dot com)

Web site: webserv.cp.su.ac.th/~pinyotae/compro2

Facebook group: ComputerProgramming@CPSU

ปฏิบัติการที่ 2.1

เราจะเรียนอะไรในวันนี้

- ความรู้ทั่วไปเกี่ยวกับอาเรย์
  - ตัวแปรอาเรย์ ชนิดข้อมูลอาเรย์
  - คำสั่ง new และวัตถุสำหรับเก็บข้อมูลอาเรย์
  - ข้อดีที่จาวามีและซีไม่มีเกี่ยวกับอาเรย์
- การเติมค่าคงที่เข้าไปในอาเรย์ด้วย Arrays.fill

ทำความรู้จักและสร้างอาเรย์

- สำหรับภาษาจาวา อาเรย์จัดเป็นชนิดข้อมูลขั้นสูง (Advanced data type)
  - คือไม่ได้เก็บข้อมูลแต่เพียงอย่างเดียว
  - แต่มีคุณลักษณะและความสามารถที่ติดมาด้วยกับอาเรย์
  - ตัวอย่างคุณลักษณะที่สำคัญคือค่าความยาวของอาเรย์
- การสร้างอาเรย์จึงมีวิธีสร้างที่อาจจะดูซับซ้อนกว่าที่เราเคยพบในภาษาซี
  - หากอาเรย์มีขนาดเล็กและตายตัว แบบในภาษาซีจะเรียบง่ายมาก
  - แต่บางกรณีการสร้างอาเรย์ในจาวาก็ถือว่าง่ายกว่า โดยเฉพาะตอนที่เราต้องการสร้างอาเรย์ขนาดใหญ่และมีขนาดที่ไม่ตายตัว
  - ในจาวา เราสร้างอาเรย์ขนาดใหญ่ในเมธอด/ฟังก์ชันได้เลย ในภาษาซีเราจะต้องทำไว้นอกฟังก์ชัน หรือไม่ก็ใช้คำสั่ง malloc ซึ่งเข้าใจได้ยากมาก

ไม่พูดมากแล้ว สร้างอาเรย์กันเลยดีกว่า

- ตอนเขียนภาษาซี เราสร้างอาเรย์ของจำนวนเต็มที่มี 10 ช่องแบบนี้

```
int a[10];
```

- สำหรับจาวา โค้ดที่ให้ผลเหมือนกัน หน้าตาเป็นดังนี้

```
int[] a = new int[10];
```

- จุดที่เราควรสังเกตและรับทราบไว้แต่เนิ่น ๆ ก็คือว่า
  - การเขียนว่า `int[]` เป็นการบอกชนิดข้อมูลของตัวแปร `a` ว่า `a` นี้คืออาเรย์ของจำนวนเต็ม
  - การใช้ `new int[10]` เป็นการสั่งสร้างพื้นที่เก็บข้อมูล การเขียนว่า `int[] a;` จะได้แค่ตัวแปรอาเรย์ แต่พื้นที่เก็บจริงไม่ได้เกิดขึ้นตามมา
  - ตัวแปร `a` นี้เทียบได้กับตัวชี้ (pointer) ในภาษาซี คือตัวชี้กับที่เก็บข้อมูลจริงเป็นคนละส่วนกัน

สร้างอาเรย์ใหญ่ ๆ ได้ (ตราบใดที่หน่วยความจำเหลือ)

- การจะสร้างอาเรย์ใหญ่ ๆ ในฟังก์ชันภาษาซี เช่น เขียนว่า

```
int a[10000000];
```

เป็นสิ่งที่มักจะล้มเหลวด้วยข้อจำกัดด้านหน่วยความจำของฟังก์ชันและการสร้าง อาเรย์ที่มีขนาดตายตัว (static array)

- ในภาษาซี เราจะแก้เกมด้วยด้วยการใช้คำสั่ง malloc เพื่อให้สร้างอาเรย์ขนาดใหญ่หรืออาเรย์ที่กำหนดขนาดได้ขณะรันโปรแกรม
  - ขนาดของอาเรย์ข้างบนนี้ถูกกำหนดไว้ตายตัวในโค้ด
  - ถ้าเปลี่ยนตามค่า N ที่ผู้ใช้ใส่เข้ามาอะไรทำนองนั้น เราจึงจะเรียกว่ากำหนดขนาดขณะรันโปรแกรม
  - และนี่คือหน้าตาโค้ดอันน่าสะพรึงกลัวในภาษีซี

```
int* a = (int*) malloc(10000000 * sizeof(int));
```

เปรียบเทียบกันอีกรอบระหว่างซีกับจาวา

- ภาษาจาวา
```
int[] a = new int[10000000];
```
- ภาษาซี
```
int* a = (int*) malloc(10000000 * sizeof(int));
```
- นอกจากนี้ ในภาษาซียังมีปัญหาเรื่องที่โปรแกรมเมอร์ต้องตามเก็บหน่วยความจำที่มาจากการ `malloc` เองด้วย
  - คือถ้า `malloc` หน่วยความจำใดออกมาแล้ว เราก็จะต้องใช้คำสั่ง `free` เพื่อคืนหน่วยความจำเวลาที่เราใช้เสร็จแล้ว
  - นั่นคือถ้าโปรแกรมเรามีหลายส่วน และแต่ละส่วน `malloc` ที่เก็บข้อมูลมาแล้วไม่คืนหน่วยความจำ รวม ๆ แล้วก็อาจทำให้หน่วยความจำหมด
- ส่วนในภาษาจาวา ถ้าออกแบบโปรแกรมมาดีพอ จาวาเวอร์ชวลแมชชีนจะสามารถตามเก็บหน่วยความจำที่เรา `new` มาได้แบบอัตโนมัติ

ตัวอย่าง: ใส่เลข -1 เข้าไปในอาเรย์ที่สร้างขึ้นมา

- สมมติว่าเราต้องการใส่ค่าเริ่มต้นเป็น -1 ลงในอาเรย์ที่ยาว 10 ช่อง

```
int[] a = new int[10];
for(int i = 0; i < 10; ++i)
  a[i] = -1;
```

- จะเห็นได้ว่าจะต่างกับภาษาซีก็เฉพาะตอนสร้างอาเรย์ขึ้นมา
  - ช่องอาเรย์ยังเริ่มนับจากหมายเลข 0 เช่นเดิม

ข้อมูลในอาเรย์มีค่าเริ่มต้นเป็นศูนย์

- ในหน้าที่แล้ว เราใส่ -1 เข้าไปในอาเรย์ เมื่อเราสร้างอาเรย์เสร็จ คำถามมีอยู่ว่า ถ้าเราไม่ทำอะไรเลยหลังจากสร้างอาเรย์ ค่าจะเป็นอย่างไร
  - ใน**ภาษาซี** ค่าจะเป็นเลขที่คาดเดาได้ยากว่าจะได้เป็นเลขอะไร (ให้เราคิดว่ามันเป็นเลขที่เหมือนโดนสุ่มขึ้นมาอย่างนั้นก็พอได้)
  - ใน**ภาษาจาวา** เลขในอาเรย์จะมีค่าเป็นศูนย์ทันทีที่สร้างเสร็จจาก **new**
- ดังนั้นในตอนใช้งาน ถ้าหากค่าที่เราต้องการ มันเป็นศูนย์พอดี เราก็ไม่ต้องทำอะไรเพิ่มเติมก็ได้
  - แต่บางทีเราก็จะใส่ค่าเข้าไปโดยชัดแจ้งเพื่อให้เห็นชัดว่าเราต้องการให้ค่าข้างในเป็นเท่าใด

วิธีการเติมค่าคงที่เข้าไปในอาเรย์แบบรวดเดียวจบ

- สมมติว่าเราต้องการใส่ค่าเริ่มต้นเป็น -1 ลงในอาเรย์ที่ยาว 1 ล้านช่อง ด้วยความรู้ที่เรามี เราก็คงจะเขียนแบบนี้

```
int[] a = new int[1000000];
for(int i = 0; i < 1000000; ++i)
  a[i] = -1;
```

- ทว่าในยุคสมัยนี้ การเขียนเช่นนั้นถือว่าเยิ่นเย้อ และถ้าอาเรย์ยาวกว่านี้อาจจะใช้เวลาในการประมวลผลนานกว่าที่ควรเป็น
  - ในภาษาจาวาเรามีวิธีที่ดีกว่านั้น คือใช้เมธอด fill ใน java.util.Arrays
  - ซึ่งลูปสองบรรทัดนั้นจะถูกรวบลงเป็นคำสั่งสั้น ๆ ที่เข้าใจแสนง่ายดังนี้

```
int[] a = new int[1000000];
java.util.Arrays.fill(a, -1);
```

ตอนนี้เรารู้อะไรแล้วบ้าง

- อาเรย์ในภาษาจาวาจัดเป็นชนิดข้อมูลแบบหนึ่ง ทำให้เราต้องประกาศชนิดอย่างชัดแจ้งในตอนสร้างตัวแปร
  - ลองเปรียบเทียบเหตุการณ์สองอย่างนี้
  1. สร้างตัวแปรจำนวนเต็ม `x` เราเขียนว่า **`int x;`**
  2. สร้างตัวแปรอาเรย์เก็บจำนวนเต็ม เราเขียนว่า **`int[] x;`**
- ตัวแปรอาเรย์เป็นเหมือนตัวชี้ ไม่ใช่ที่เก็บข้อมูลที่แท้จริง เราจึงมักจะสร้างทั้งตัวแปรอาเรย์และที่เก็บข้อมูลควบคู่กันไปด้วยคำสั่ง `new`

```
int[] x = new int[10];
```

- ในโค้ดข้างบนเรากล่าวว่า ส่วน `int[] x` คือการประกาศตัวแปรอาเรย์ ส่วนตรง `new int[10]` คือการสร้างวัตถุสำหรับเก็บข้อมูลอาเรย์
- เราใช้เมธอด `fill` ใน `java.util.Arrays` เพื่อเติมค่าเริ่มต้นได้

ตอนนี้เรารู้อะไรแล้วบ้าง (2)

- ค่าเริ่มต้นในอาเรย์ตัวเลข (ทั้ง int, float, double, ...) จะเป็นเลขศูนย์
- ถ้าเราอยากได้เลขอื่น เราใช้เมธอด `fill` ใน `java.util.Arrays` เพื่อเติมค่าเริ่มต้นที่ต้องการได้

```
int[] a = new int[1000000];
java.util.Arrays.fill(a, -1);
```