

Project Array Part II

COS2103

Data structure and Algorithm

1 / 2568

Present to

Asst.Prof.DR.Chouvalit Khancome

By Krittaya Tantichaiyakul

6705500269

หัวข้อ

อุปกรณ์กีฬา



Assignment

COS2103 เทอม 1/68

K

งาน 1.2 ArrayLists

Asst.Prof.Dr.Chouvalit Khancome • 5 ส.ค. (แก้ไข 16 ต.ค.)

50/50

ครบกำหนด 18 ต.ค.

ให้พัฒนา แอปพลิเคชันที่ออกแบบเหมือนกับ งาน 1.1 แต่ใช้ java.util.ArrayList แทนโครงสร้างข้อมูลที่เรียนกับอาจารย์ และใช้อัลกอริทึมได้อย่างอิสระ
ส่งในลักษณะเดียวกัน

👤

ความคิดเห็นในชั้นเรียน


💬

เพิ่มความคิดเห็น

งานของคุณ


ให้คะแนนแล้ว

ProjectArrayPar...
Compressed Archive



×

การทำงานของโปร...
PDF



×

+

เพิ่มหรือสร้าง

ส่งอีกครั้ง

👤

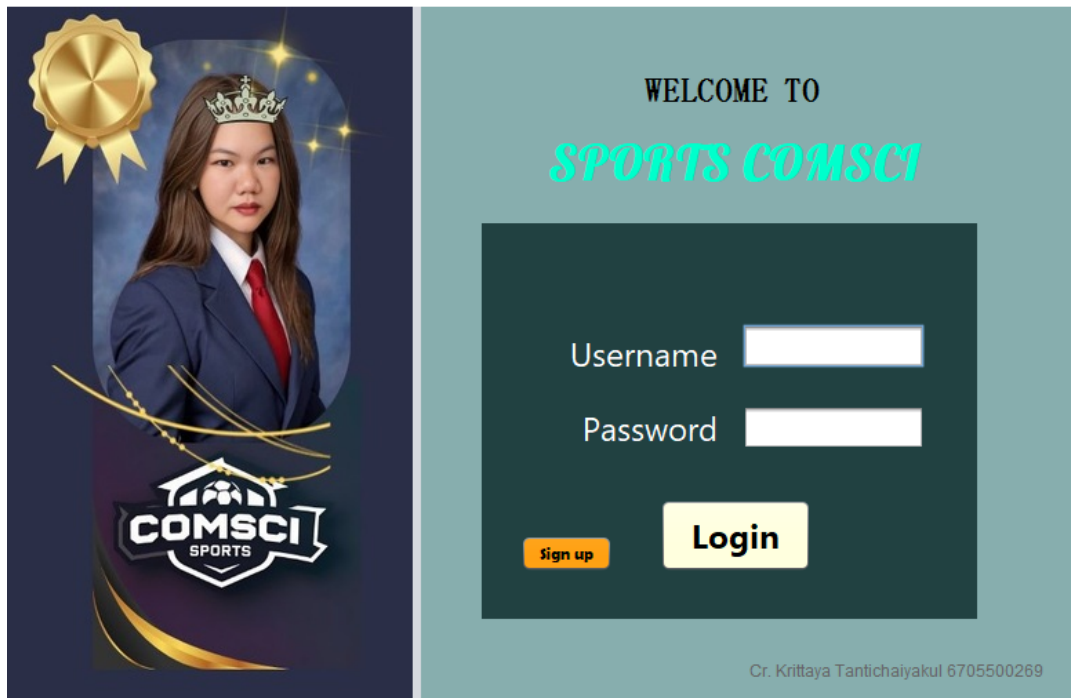
ความคิดเห็นส่วนตัว

เพิ่มความคิดเห็นถึง

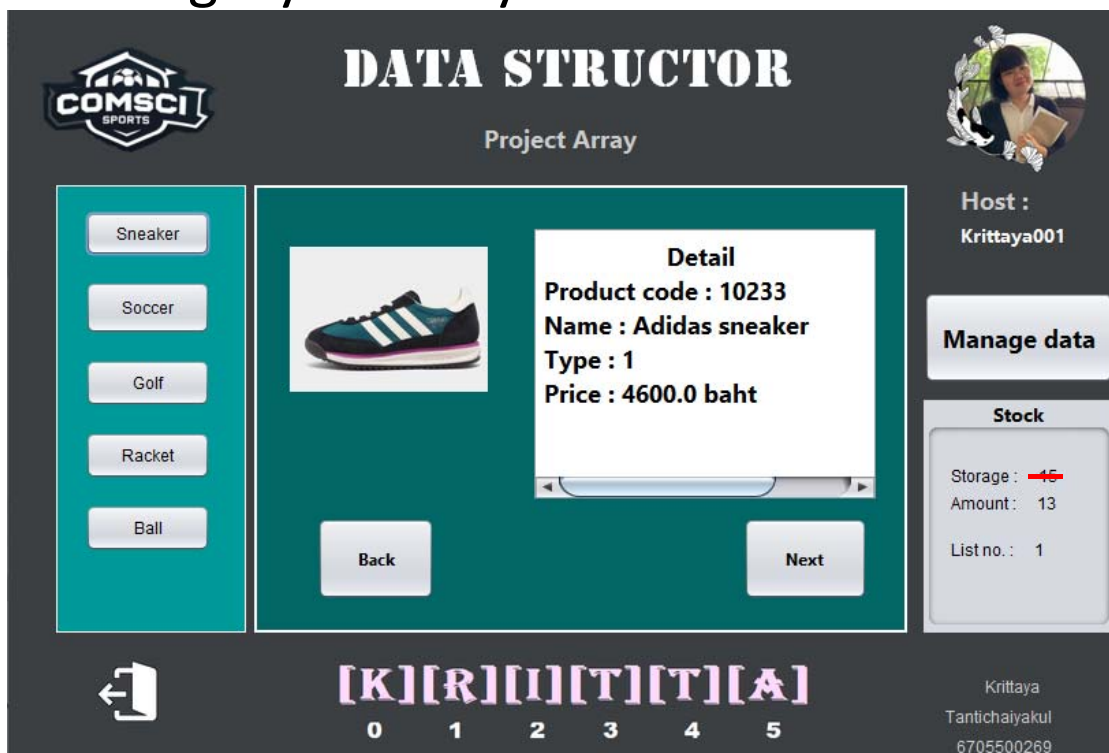
Asst.Prof.Dr.Chouvalit Khancome

Program Interface Preview

1. Login page - Login



2. Category of array - Menu



3. list of products – Manage

DATA STRUCTOR AND ALGORITHM

Delete Insert

search code :

Host :
Krittaya001

Delete code

Status : delete complete!

Code : 10841

Name : Lacoste speed turbo

Price : 3500.0

Type : 1

Delete

No.	Code	Name	Price	Type	Address
1	10233	Adidas sne...	4600.0	1	/sc/sneake...
2	10472	Nike sneaker	4200.0	1	/sc/sneake...
3	10785	Skecher sn...	2500.0	1	/sc/ske.jpg
4	20096	Umbro Glove	800.0	2	/sc/ub.jpg
5	20504	Nike footba...	1500.0	2	/sc/shirtNik...
6	30319	Ping stick g...	1300.0	3	/sc/golfstic...
7	30873	Anyday golf ...	5000.0	3	/sc/golfbag...
8	40152	Yonex strike	2400.0	4	/sc/yon.png
9	40197	Kawasaki s...	1700.0	4	/sc/kawa.jpg
10	40601	Victor shuttl...	200.0	4	/sc/shuttle.j...
11	50290	Puma footb...	800.0	5	/sc/puma.jpg
12	50475	Mikasa voll...	1200.0	5	/sc/volley.jpg
13	50778	Wilson bas...	1500.0	5	/sc/wilson.j...

Stock

Storage : ~~15~~

Amount : 13

Present to Asst.Prof.DR.Chouvalit Khancome
 by.. Krittaya Tantichaiyakul 6705500269

4. ไฟล์กลางจัดเก็บอาร์เรย์ – DataStore

```

11 public class DataStore {
12     public static int n=13;
13     final int UB=15;
14     int LB=0;
15     public static class Products {
16         public int code;
17         public String name;
18         public int type;
19         public float price;
20         public String pic;
21     }
22     public void setAll(int a, String b, int c, float d, String e){
23         code = a;
24         name = b;
25         type = c;
26         price = d;
27         pic = e;
28     }
29     static int getN(){
30         return n;
31     }
32 }
33
34
35 public static Products[] list = new Products[15];
36
37 static {
38     for (int i = 0; i < list.length; i++) {
39         list[i] = new Products();
40     }
41 }
42
43 // กำหนดข้อมูลตั้งแต่แรก
44 list[0].setAll(10233,"Adidas sneaker",1,4600,"/sc/sneakerAdi.jpg");
45 list[1].setAll(10472,"Nike sneaker",1,4200,"/sc/sneakerNike.jpg");
46 list[2].setAll(10785,"Skecher sneaker",1,2500,"/sc/ske.jpg");
47 list[3].setAll(20096,"Umbro Glove",2,800,"/sc/ub.jpg");
48 list[4].setAll(20504,"Nike football shirt",2,1500,"/sc/shirtNike.jpg");
49 list[5].setAll(30319,"Ping stick golf",3,1300,"/sc/golfstick.jpg");
50 list[6].setAll(30873,"Anyday golf bag",3,5000,"/sc/golfbag.jpg");
51 list[7].setAll(40152,"Yonex strike",4,2400,"/sc/yon.png");
52 list[8].setAll(40197,"Kawasaki strike",4,1700,"/sc/kawa.jpg");
53 list[9].setAll(40601,"Victor shuttlecock",4,200,"/sc/shuttle.jpg");
54 list[10].setAll(50290,"Puma football",5,800,"/sc/puma.jpg");
55 list[11].setAll(50475,"Mikasa volleyball",5,1200,"/sc/volley.jpg");
56 list[12].setAll(50778,"Wilson basketball",5,1500,"/sc/wilson.jpg");
57
58 }

```

ความแตกต่างระหว่าง

Part I และ Part II

1. สำหรับ Part I จะมีการใช้งานอาร์เรย์แบบปกติ array[] และมีการใช้งานคู่กับอัลกอริทึมค่อนข้างมาก ส่วน Part II จะมีการนำเข้า Java.util.ArrayList มาใช้งาน ซึ่งแพ็คเกจนี้จะอำนวยความสะดวกในการใช้งานมากขึ้น มีการใช้งานอัลกอริทึมที่น้อยลง สั้นลง รวดเร็วขึ้น ทำให้ผู้เขียนโค้ดได้ง่าย และโปรแกรมทำงานได้ดียิ่งขึ้น

2. การสร้างอาร์เรย์แบบปกติ จะบังคับให้ผู้ใช้เก็บข้อมูลได้ตามขนาดของมัน หากเต็มแล้วต้องการจะเพิ่มข้อมูลอีก จำเป็นต้องลบหรือสร้างอาร์เรย์ขึ้นมาใหม่ที่มีขนาดใหญ่กว่าเดิม ซึ่งเป็นปัญหาสำหรับใครหลายคน แพ็คเกจ Java.util.ArrayList นี้เองจึงถูกสร้างมาเพื่อแก้ปัญหาดังกล่าว จะทำให้อาร์เรย์ของเรามีความยืดหยุ่นมากขึ้น กล่าวคือ สามารถเพิ่มข้อมูลได้เรื่อยๆ นั่นเอง

DATA STRUCTOR AND ALGORITHM

Delete

Insert

search code :

Code

66666

Name

Ari

Price

6000

Type

6

Instead No.

19

Insert

No.	Code	Name	Price	Type	Address
3	10185	Skecher s...	2500.0	1	/sc/ske.jpg
4	20096	Umbro Glo...	800.0	2	/sc/sub.jpg
5	20504	Nike footb...	1500.0	2	/sc/shirtNi...
6	30319	Ping stick ...	1300.0	3	/sc/golfstic...
7	30873	Anyday golf...	5000.0	3	/sc/golfbag...
8	40152	Yonex strike	2400.0	4	/sc/yon.png
9	40197	Kawasaki ...	1700.0	4	/sc/kawa.jpg
10	40601	Victor shutt...	200.0	4	/sc/shuttle....
11	50290	Puma foot...	800.0	5	/sc/puma.j...
12	50475	Mikasa voll...	1200.0	5	/sc/volley.jpg
13	50778	Wilson ba...	1500.0	5	/sc/wilson....
14	11111	Fila	1000.0	1	/sc/home....
15	22222	Asics	2000.0	2	/sc/home....
16	33333	New Balan...	3000.0	3	/sc/home....
17	44444	Baoji	4000.0	4	/sc/home....
18	55555	Crocs	5000.0	5	/sc/home....
19	66666	Ari	6000.0	6	/sc/home....

Host :

Krittaya001

Stock

Storage : -

Amount : 19

Present to Asst.Prof.DR.Chouvalit Khancome
by.. Krittaya Tantichaiyakul 6705500269

3. เนื่องจากอาร์เรย์ชนิด arraylist สามารถยืดหยุ่นได้ และมีคำสั่ง size() เพื่อตรวจสอบขนาดของอาร์เรย์ได้ เราจึงไม่จำเป็นต้องใช้ตัวแปร n อีกต่อไป เพียงแต่คำสั่งอาจพิมพ์ยาวนิดนึง เนื่องจากเราประกาศ arraylist ไว้ในไฟล์กลาง DataStore จึงต้องมีการเข้าถึง ก็คือ DataStore.list.size() แต่แตกต่างกับการที่เวลาเรา insert delete หรืออัปเดตกระดานสติก เราไม่จำเป็นต้องกังวลเรื่องความถูกต้องของ n++ หรือ n— อีกแล้ว

```
int insertCode=Integer.parseInt(jTextField1.getText());
int insertIndex=Integer.parseInt(jTextField7.getText());
insertIndex=1;
for(int i=0;i<DataStore.list.size();i++){
    if(insertCode==DataStore.list.get(i).code){
        System.out.println("This code exist already!");
        JOptionPane.showMessageDialog(this, "This code exist already!");
        return;
    }
}
if(insertIndex<0){
```

4. ปกติเราจะสร้างตัวแปร UB ซึ่งเป็นขนาดของอาร์เรย์ แต่การนำแฟ็กเกจมาใช้ อาร์เรย์ list ของเราจะไม่มีวันเต็ม ดังนั้นตัวแปร UB จึงไม่จำเป็นสำหรับการใช้อัลกอริทึมอีกต่อไป เวลากำหนดสเกล เราก็สามารถตั้งว่า 0-DataStore.list.size() หรือพูดง่าย ๆ ก็คือ ตั้งแต่ 0 ถึง จำนวนสมาชิกทั้งหมดได้เลย

5. ในหน้า Menu คำสั่งปุ่ม next back ปกติเราจะต้องสร้างเงื่อนไขแบบเลือกอย่างใดอย่างหนึ่ง คือ ค่า current อยู่ภายในจำนวนสินค้าที่มีอยู่ไหม เพื่อโชว์ข้อมูลภายในสินค้าที่มีอยู่ ต่อมา ค่า current อยู่ภายในขนาดของอาร์เรย์ไหม เพื่อโชว์ข้อมูลช่องว่างที่เหลือด้วย แต่สำหรับการนำแฟ็กเกจมาใช้ เราจะไม่มีความกังวล ดังนั้นไม่จำเป็นต้องโชว์ตำแหน่งข้อมูลที่ว่างแล้ว สามารถทำให้ current ไม่เกินจำนวนสินค้าหรือ .size() ได้เลย

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent e) {
    if(current<DataStore.list.size()-1){
        current++;
        jLabel1.setIcon(new javax.swing.ImageIcon(
            getClass().getResource(DataStore.list.get(current).pic)
        ));
    }else{
        JOptionPane.showMessageDialog(this, "Overflow!");
        System.out.println("Overflow!");
    }
}
```

6. สำหรับการ insert ถือเป็นหัวข้อที่เห็นความแตกต่างระหว่างอาร์เรย์แบบปกติกับอาร์เรย์ลิสมากที่สุด ปกติการที่เราจะแทรกข้อมูลลงอาร์เรย์แบบปกติ เราจะใช้อัลกอริธึมมาก เริ่มตั้งแต่ตรวจสอบว่าโค้ดซ้ำไหม อาร์เรย์เต็มหรือยังและตำแหน่งที่ต้องการแทรกอยู่ภายในอาร์เรย์ไหม ซึ่งสำหรับ arraylist เราไม่มีเพดาน จึงไม่จำเป็นต้องใช้ if ตรวจสอบว่าอาร์เรย์เต็มหรือยัง แทรกข้อมูลเข้าไปได้เลย ต่อมา if ตรวจสอบว่าตำแหน่งที่ต้องการแทรกเกินอาร์เรย์ลิสไหม จากเดิม $insertIndex \geq (UB-1)$ เราสามารถแทรกลงไปตรงไปตำแหน่งไหนของอาร์เรย์ก็ได้ รวมถึงช่องว่างด้วย แต่ครั้งนี้เราจะไม่มีการช่องว่างแล้ว จึงต้องเปลี่ยน UB เป็น size() และที่สำคัญ ผู้ใช้ยังมีโอกาสที่จะแทรกข้อมูลเข้าไปเป็นตัวสุดท้ายได้ด้วย เราจะต้องขยายสเกล +1 โดยเปลี่ยนจาก \geq เป็น $>$ หรือให้ $size()+1$ ก็ได้เหมือนกัน

```
530 int insertCode=Integer.parseInt(jTextField1.getText());
531 int insertIndex=Integer.parseInt(jTextField7.getText());
532 insertIndex-=1;
533 1 for(int i=0;i<DataStore.list.size();i++){
534     if(insertCode==DataStore.list.get(i).code){
535         System.out.println("This code exist already!");
536         JOptionPane.showMessageDialog(this, "This code exist already!");
537         return;
538     }
539 }
540 2 if(insertIndex<0){
541     System.out.println("Underflow!");
542     JOptionPane.showMessageDialog(this, "Underflow!");
543     return;
544 }
545 3 if(insertIndex>(DataStore.list.size())){
546     System.out.println("Overflow!");
547     JOptionPane.showMessageDialog(this, "Overflow!");
548     return;
549 }
550 }
551 DataStore.Products obj2 = new DataStore.Products();
552 obj2.code=Integer.parseInt(jTextField1.getText());
553 obj2.name=jTextField2.getText();
554 obj2.price=Integer.parseInt(jTextField3.getText());
555 obj2.type=Integer.parseInt(jTextField4.getText());
556 obj2.pic="/sc/home.png";
557 DataStore.list.add(insertIndex,obj2);
558 System.out.println("Status : insert complete!");
559 jLabel21.setText("Amount : "+DataStore.list.size());
560 loadTableData();
561 }
```

} Temp

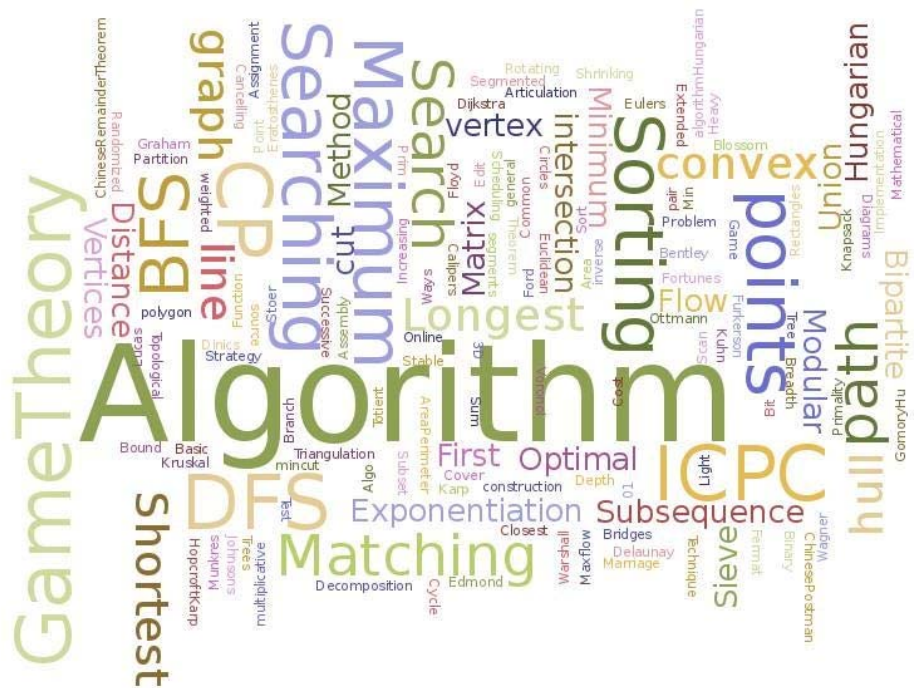
การใช้ฟังก์ชัน add เป็นคำสั่งที่จะเพิ่มข้อมูลลงอาเรย์ลิสต์อัตโนมัติ โดยที่เราไม่จำเป็นต้องขยับอีลิเมนต์ตัวข้างหลังออกทุกตัวอีกแล้ว ซึ่งตรงนี้ก็ถือว่าเป็นการลดขั้นตอนที่ดีมาก ตอบโจทย์วิชานี้ก็คือ การออกแบบอัลกอริทึมอย่างมีประสิทธิภาพ ทำให้จำนวน Big O ลดลง แต่คำสั่งนี้มีเงื่อนไขว่า ต้องแนบอาทิวเม้นต์ตัวที่ 2 เป็นอ็อบเจกต์ที่เป็นข้อมูลใหม่ลงไปด้วย เราจะต้องสร้างอ็อบเจกต์ใหม่สำหรับเซตค่าที่ได้รับจาก TextField ไว้ก่อน หลังจากที่อ็อบเจกต์ตัวนั้นมีหน้าตาของมันเรียบร้อยแล้ว แล้วจึงสามารถใส่ลงไปบนฟังก์ชัน add(int, object) เพื่อเพิ่มข้อมูลได้แล้ว

2. สำหรับการ delete จะมีคำสั่งที่อำนวยความสะดวกก็คือ remove() มาพร้อมใช้งาน เราไม่จำเป็นต้องใช้ลูป for แล้วขยับตัวหลังมาแทนตัวหน้าอีกเช่นกัน ใช้คำสั่ง remove() ก็จบ แต่คำสั่งนี้มีเงื่อนไขคือ ต้องใส่อาทิวเม้นต์เป็นค่า index เพื่อลบตำแหน่งนั้นด้วย ดังนั้น เราจึงต้อง search หาดำแหน่งดังกล่าวก่อน แล้วจึงนำค่านั้นใส่ลงบนอาทิวเม้นต์ได้เลย เช่น remove(index)

```
569 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
570     int delCode = Integer.parseInt(jTextField5.getText());  
571     int index=SearchData(delCode);  
572     if(index<0){  
573         System.out.println("Status : not found");  
574         jLabel11.setText("Status : not found..");  
575         return;  
576     }  
577     jLabel12.setText("Code : "+DataStore.list.get(index).code);  
578     jLabel13.setText("Name : "+DataStore.list.get(index).name);  
579     jLabel14.setText("Price : "+DataStore.list.get(index).price);  
580     jLabel15.setText("Type : "+DataStore.list.get(index).type);  
581     DataStore.list.remove(index);  
582     System.out.println("Status : delete complete!");  
583     jLabel11.setText("Status : delete complete!");  
584     jLabel21.setText("Amount : "+DataStore.list.size());  
585     loadTableData();  
586 }  
587  
588
```

3. สำหรับการค้นหา search นั้นยังไม่มีคำสั่งที่ค้นหาตำแหน่งโดยตรง จะมีที่ใช้แทนกันได้บ้างก็คือ contains() แต่คำสั่งนี้จะให้ค่า Boolean ว่าเจอหรือไม่เจอแค่นั้น แต่โปรแกรมของเราจำเป็นต้องใช้คำสั่งที่ค้นหาตำแหน่งด้วย เพื่อนำมาประยุกต์ใช้กับการลบและแทรกข้อมูล หนูจึงยังคงต้องใช้ลูปเดิมในการค้นหาเด้ออยู่ค่ะ

รวมถึงการ sort ด้วยเช่นกัน แพ็คเกจอาเรย์ลิสต์ยังไม่มีคำสั่งในการจัดเรียงข้อมูลค่ะ



ขอบคุณค่ะ

A decorative flourish consisting of symmetrical, swirling lines that frame the text. At the bottom center, there is a small, stylized floral or leaf-like motif.