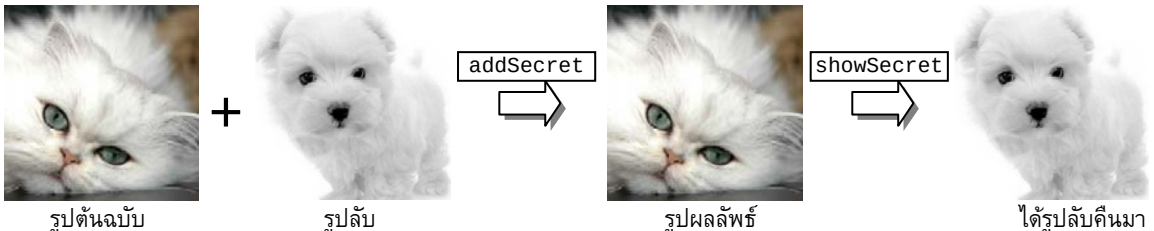
การบ้านครั้งที่ 2 : ก่อนสอบปลายภาค

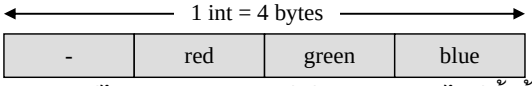
การบ้านนี้มี 4 ข้อ ให้เขียนทุกคลาสเก็บในแฟ้ม jlab เดียว ตั้งชื่อขึ้นต้นด้วย HW2- ตามด้วยรหัสนิสิต (เช่น **HW1-5230123421.jlab**) แนบมากับ email พร้อมกับแนบภาพผลลัพธ์ (ของข้อ 1) มาที่ **java2110101@gmail.com** มีหัวเรื่องของ email เป็น **2110101-HW2 ตามด้วยรหัสนิสิต** (เช่น 2110101-HW2-5230123421) ก่อนเที่ยงคืนวันที่ 21 กุมภาพันธ์นี้ (ขอเน้นว่าให้ตั้งชื่อแฟ้ม และหัวเรื่องของ email ให้ตรงตามข้อกำหนด และครั้งนี้จะไม่รับการบ้านสายอย่างเด็ดขาด)

1. จอภาพคอมพิวเตอร์สามารถแสดงสีของแต่ละจุดภาพได้ถึง 16 ล้านสี แต่ถ้าสีของจุดภาพเพี้ยนไปเล็กน้อย ตามนุษย์ทั่วไปมักแยกไม่เห็นความแตกต่าง เราจะอาศัยข้อเท็จจริงนี้ในการขโมยที่เก็บข้อมูลของแต่ละจุดภาพมาเก็บข้อมูลลับลงไปในแฟ้มภาพ ซึ่งเมื่อเปิดแฟ้มภาพนี้ผู้ชมจะมองไม่ออกว่า มีการปนข้อมูลลับในภาพ แต่เราสามารถอ่านข้อมูลลับนี้กลับคืนออกมาได้ การบ้านนี้ให้เขียนเมท็อด **addSecret** เพื่อแทรกรูปลับขาวดำ (สีเทา) เข้าในแฟ้มภาพ และเมท็อด **showSecret** เพื่ออ่านรูปลับกลับคืนมาได้ด้วย ดังตัวอย่างข้างล่างนี้



รูปต้นฉบับ | รูปลับ | รูปผลลัพธ์ (แทบไม่เห็นความแตกต่างกับรูปต้นฉบับ) | ได้รูปลับคืนมา

ขอทวนเล็กน้อยว่า เราใช้คำสั่ง **w.loadImage(filename)** เพื่ออ่านแฟ้มภาพมาแสดงในวินโดว์ **w** จากนั้นสามารถอ่านแผนที่จุดภาพได้ด้วยคำสั่ง **w.getPixmap()** ผลที่ได้คืออาเรย์สองมิติแบบ **int** แต่ละ **int** เก็บสีของจุดภาพหนึ่งจุดมีขนาด 4 ไบต์ (หนึ่ง **int** มีสี่ไบต์) แบ่งเป็นสามไบต์ขวาเก็บแยกแม่สี แดง เขียว น้ำเงิน ดังรูปข้างล่างนี้ (ไบต์ซ้ายสุดแทนระดับความโปร่งที่ไม่ขออธิบาย)



แม่สีหนึ่งสีใช้ที่เก็บ 1 ไบต์ = 8 บิต จึงเก็บแต่ละแม่สีได้ 2^8 ระดับ (สามสี สีละ 2^8 ระดับ ได้สีทั้งสิ้น $2^8 2^8 2^8 = 2^{24} \approx 16$ ล้านสี) ถ้าเราขโมยบิตทางขวาของแต่ละแม่สี มาใช้เก็บข้อมูลลับ สีเดิมของรูปจะเพี้ยนไปเล็กน้อย แต่สังเกตไม่เห็น (เพราะบิตขวามีนัยสำคัญต่ำ) จะขอขโมย 2 บิตขวาของแม่สีแดง และ 3 บิตขวาของแม่สีเขียวและน้ำเงิน รวมเป็น 8 บิตต่อจุดภาพมาเก็บระดับสีเทาของรูปภาพลับ ตัวอย่างเช่น จุดภาพที่พิกัด (10, 12) ในรูปต้นฉบับเป็น | 10101001 | 11111111 | 11011110 | (แสดงเป็นฐานสอง) ส่วนจุดภาพของรูปลับที่พิกัดดียวกัน (10,12) มีระดับความเทาเป็น **10011010** (เขียนฐานสองเช่นกัน) เรานำระดับเทานี้มาแยกเป็นสามส่วนคือ **10** **011** และ **010** นำไปวางแทนบิตขวาของแต่ละแม่สีของจุดภาพต้นฉบับ จะได้ | 10101010 | 11111011 | 11011010 | กระทำเช่นนี้กับทุก ๆ จุดภาพของรูปต้นฉบับและรูปลับ จะได้รูปต้นฉบับที่เก็บรูปลับที่มองไม่ค่อยเห็นข้อมูลที่ปนเข้าไป

อาจสงสัยว่า จะแยกข้อมูลออกเป็นบิต ๆ และเปลี่ยนแปลงเป็นบิต ๆ ได้อย่างไร จาวามีคำสั่งจัดการระดับบิต แต่เนื่องจากเราไม่ได้เรียนคำสั่งเหล่านี้ จึงขอใช้การคำนวณแทน ถ้ายังจำได้ หากเราต้องการเลขหลักพันของจำนวนเต็ม **a** ก็ทำได้ด้วยนิพจน์ **a / 1000 % 10** (การหาร **1000** เพื่อตัดหลักร้อย สิบ และหน่วยทิ้ง จากนั้น **% 10** เพื่อดึงหลักหน่วยของผลลัพธ์ ซึ่งคือหลักพันของ **a** ที่ต้องการนั่นเอง) ปัญหาของเราคือ ดึงออกมา 3 บิต สามบิตแทนค่าได้ตั้งแต่ 000 ถึง 111 (ฐานสอง) ซึ่งคือ 0 ถึง 7 (ฐานสิบ) จึงใช้การ **/ 8** และ **% 8** เช่น ต้องการดึง 3 บิตขวาสุดของ **a** ก็ใช้ **a % 8** ต้องการดึง 3 บิตถัดมาทางซ้าย ก็ใช้ **a / 8 % 8** และต้องการสองบิตถัดไปทางซ้าย ก็ใช้ **a / 64 % 4** (การ mod 4 คือการดึงแค่ 2 บิต)

คลาส **SWindow** ในหน้าถัดไปแสดงเมท็อด **mixSecret(sourcePixel, secretPixel)** ที่คืนผลลัพธ์ที่ได้จากการนำสีของ **secretPixel** มาแปลงเป็นระดับสีเทา แล้วเติมเข้าไปในค่าของ **sourcePixel** และแสดงเมท็อด **getSecret(onePixel)** ที่คืนค่าสีเทาที่ได้จากการสกัดข้อมูลลับใน **onePixel**

จงเขียนคลาส **SWindow** ให้เป็นคลาสลูกของ **DWindow** มีเมท็อด **addSecret** และ **showSecret** ให้บริการเพิ่มรูปลับ และแสดงรูปลับ ดังแสดงโครงของคลาสข้างล่างนี้ (ตัวอย่างการใช้งานแสดงในเมท็อด **main**) สำหรับกรณีที่รูปต้นฉบับและรูปลับมีขนาดไม่เท่ากัน ให้ใช้ขนาดของรูปต้นฉบับเป็นสำคัญ (หมายเหตุ : ต้องตั้งสีของจอภาพของคอมพิวเตอร์ที่ใช้เป็นแบบ 32 bit จึงจะทำงานถูกต้อง)

หมายเหตุ : นอกจากเราใช้ **w.loadImage(…)** ในการอ่านแฟ้มภาพมาแสดงในวินโดว์ได้แล้ว เราสามารถใช้ **w.loadImage(…)** อ่านภาพจาก web ได้ด้วย เช่น **w.loadImage("http://www.eng.chula.ac.th/files/u1/Building2.jpg")**

- เมื่อเขียนเมท็อด **showSecret** เสร็จแล้ว สามารถทดลองอ่านรูปดังต่อไปนี้จาก web ดูว่า มีรูปลับอะไรซ่อนอยู่ **http://www.cp.eng.chula.ac.th/~somchai/2110101/showme.png**
- เมื่อเขียนเมท็อด **addSecret** เสร็จแล้ว ให้ลองหาภาพหนึ่งภาพมาเพิ่มใส่ในอีกภาพหนึ่ง **นำผลที่ได้แนบมากับ email ด้วย**

2. นิสิตคงเคยเห็นรายการโทรทัศน์ที่ในบางฉากต้องทำภาพเบลอในบางบริเวณ เช่น ข่าวอาชญากรรมที่มีภาพน่าสะเทือนใจ หรือโฆษณาแฝงที่ต้องปิดชื่อสินค้า เป็นต้น เขาเรียกการทำเช่นนี้ว่า pixelation เสมือนเป็นการขยายจุดภาพให้มีขนาดใหญ่ขึ้น ดังแสดงในรูปข้างล่างนี้ เป็นการทำ pixelate เฉพาะบริเวณโลโก้ของสินค้า โดยกำหนดขนาด (size) ของการทำ pixelate ที่ต่างกัน

   จงเขียนเมท็อด **pixelate** เพิ่มในคลาส **SWindow** (Pixelation ทำได้ด้วยการตั้งสีให้กับจุดภาพของบริเวณสี่เหลี่ยมขนาด size×size ด้วยสีตัวแทนที่ได้จากค่าเฉลี่ยของสีในบริเวณนั้น ต้องหาค่าเฉลี่ยของแต่ละแม่สี แล้วค่อยนำมา mixRGB เป็นสีตัวแทน)

size = 1 size = 5 size = 10 size = 20

```
import jlab.graphics.DWindow;
public class SWindow extends DWindow {
  //----  constructors -----------------------------

    // เขียน constructors ตามความเหมาะสม

  // ----  public methods -------------------------
  public void showSecret() {
    // เขียนคำสั่งที่นี่ เรียกใช้ getSecret ให้เป็นประโยชน์
    // เมื่อได้แผนที่จุดภาพของรูปลับแล้ว ก็ให้แสดงผลที่วินโดว์เลย

  }
  public void addSecret(SWindow secret) {
    // เขียนคำสั่งที่นี่ เรียกใช้ mixSecret ให้เป็นประโยชน์

  }
  public void pixelate(int size) {
    // ทำ pixelate ทั้งรูป

  }
  public void pixelate(int size, int upperLeftX, int upperLeftY,
                       int width, int height) {
    // ทำ pixelate เฉพาะจุดภาพในบริเวณสี่เหลี่ยมผืนผ้าที่มีจุดซ้ายบนที่ (upperLeftX, upperLeftY)
    // ขนาดกว้าง width สูง height ค่า size กำหนดขนาดของการทำ pixelate

  }
  //----  private methods  ------------------------
  private static int mixSecret(int sourcePixel, int secretPixel) {
    int r = DWindow.getR(secretPixel);
    int g = DWindow.getG(secretPixel);
    int b = DWindow.getB(secretPixel);
    int gray = (r + g + b) / 3;                                    // นำแม่สีทั้งสามมาแปลงเป็นสีเทา
    r = DWindow.getR(sourcePixel) / 4 * 4 + gray / 64 % 4;         // 2 บิตซ้ายของเทาแทนของสีแดง
    g = DWindow.getG(sourcePixel) / 8 * 8 + gray / 8 % 8;          // 3 บิตกลางของเทาแทนของสีเขียว
    b = DWindow.getB(sourcePixel) / 8 * 8 + gray % 8;              // 3 บิตขวาของเทาแทนของสีน้ำเงิน
    return DWindow.mixRGB(r, g, b);                                // ผสมกลับเป็นผลลัพธ์
  }
  private static int getSecret(int onePixel) {
    int right2 = DWindow.getR(onePixel) % 4;    // ดึง 2 บิตขวาของแม่สีแดง
    int mid3   = DWindow.getG(onePixel) % 8;    // ดึง 3 บิตขวาของแม่สีเขียว
    int left3  = DWindow.getB(onePixel) % 8;    // ดึง 3 บิตขวาของแม่สีน้ำเงิน
    return right2 * 64 + mid3 * 8 + left3;      // รวมกันเป็นระดับสีเทาของรูปลับ
  }
  //-----------------------------------------
  // เขียน main สำหรับทดสอบการทำงาน
  public static void main(String[] args) {
    SWindow secret = new SWindow("http://www.eng.chula.ac.th/files/u1/Building1Hall.jpg");
    SWindow source = new SWindow("http://www.eng.chula.ac.th/files/u1/Building2.jpg");
    source.addSecret(secret);
    source.saveImage("c:/temp/test.png");    // อย่า save เป็น jpg หรือ gif เพราะข้อมูลลับในภาพจะสูญหาย
    source.loadImage("c:/temp/test.png");    // ให้ save เป็นแบบ png หรือ bmp
    source.showSecret();
    source.sleep(1000);  // ให้เห็นภาพจริง 1 วินาที
    source.pixelate(10); // แล้วค่อยทำ pixelate
  }
}
```

3. อาเรย์ที่ใช้ในจาวานั้นขยายขนาดไม่ได้ จองไว้ 10 ช่องก็ใช้ช่องที่ index 0 ถึง index 9 เท่านั้น คงจะดีไม่น้อย ถ้ามีอาเรย์ที่ขยายขนาดให้เราใช้ index ใดก็ได้ (ที่ไม่ติดลบ) การบ้านข้อนี้ต้องการให้เขียนคลาสชื่อ **ArrayOfDouble** เพื่อให้ผู้ใช้สร้างอ็อบเจกต์ที่ทำตัวเสมือนเป็นอาเรย์ที่เก็บจำนวนแบบ **double** มีเมท็อดให้บริการตั้งค่า (**set**) ขอดูค่า (**get**) ค้นค่า (**indexOf**) และคืนความยาวอาเรย์ (**length**) ขอให้นิสิตศึกษาความหมาย วิธีการใช้งาน และผลการทำงานของแต่ละเมท็อด จากตัวอย่างที่เขียนให้ในเมท็อด **main** ของคลาส **ArrayOfDouble** ข้างล่างนี้

```
public class ArrayOfDouble {
  private double[] theArray;
  // เพิ่มเติมตัวแปรประจำอ็อบเจกต์อื่นๆ ได้ตามความเหมาะสม


  //----  constructors ----------------------------
  public ArrayOfDouble(int c) {
    // สร้างอาเรย์ภายในที่พร้อมเก็บข้อมูล c ช่อง

  }
  //----  object methods ----------------------------
  public void set(int index, double value) {
    // ให้ช่องที่ index ของอาเรย์มีค่าเท่ากับ value
    // ในกรณีที่ index มีค่ามากกว่าหรือเท่ากับขนาดของ theArray ให้ขยายขนาดของ theArray ให้เพียงพอกับการใช้ช่องที่ index ที่ได้รับ
    // ถ้า index ที่ได้รับเป็นจำนวนลบ ให้แสดงข้อความผิดพลาดทางจอภาพด้วย

  }
  public double get(int index) {
    // คืนค่าที่เก็บในช่องที่ index ของอาเรย์

  }
  public int indexOf(double value) {
    // คืนค่า index ของช่องที่เก็บค่าที่เท่ากับ value (ถ้ามีหลายช่องที่เก็บ value ให้คืนช่องที่มี index น้อยสุด)
    // หาไม่พบ คืน -1

  }
  public int length() {
    // คืนขนาดของอาเรย์

  }
  public void print() {
    // แสดงค่าต่าง ๆ ของอาเรย์นี้ทางจอภาพ

  }
  //-------------------------------------------------------------------------
  public static void main(String[] args) {
    // main นี้มีไว้ทดสอบความถูกต้องของคลาสที่เขียน
    ArrayOfDouble d = new ArrayOfDouble(5);  // จอง 5 ช่อง ช่องที่ 0 ถึง ช่องที่ 4
    d.print();  // [0.0, 0.0, 0.0, 0.0, 0.0]
    for(int i=1; i<d.length(); i+=2) d.set(i, i+0.5);
    d.print();  // [0.0, 1.5, 0.0, 3.5, 0.0]
    d.set(7, 7.0);   // ต้องการให้ค่า 7.0 กับอาเรย์ช่องที่ 7 ทำให้ภายในอ็อบเจกต์ d มีการขยายอาเรย์เป็น 8 ช่อง
    d.set(4, 1.5);   // ต้องการให้ค่า 7.0 กับอาเรย์ช่องที่ 4 อาเรย์ที่มี มีช่องที่ 4 อยู่แล้ว จึงไม่จำเป็นต้องขยายอาเรย์แต่อย่างใด
    d.print();  // [0.0, 1.5, 0.0, 3.5, 1.5, 0.0, 0.0, 7.0]
    System.out.println( d.get(3) );  // ได้ 3.5
    System.out.println( d.indexOf(1.5) );  // ได้ 1
    System.out.println( d.indexOf(9.9) );  // ได้ -1
    System.out.println( d.length() );      // ได้ 8
  }
}
```

ดูวิธีการขยายอาเรย์ในแผ่นใสของบทอาเรย์ 1 มิติ

4. ในข้อนี้เรามีคลาส **ListOfDoubles** เขียนให้เรียบร้อยแล้วดังนี้

```
public class ListOfDoubles {  // เก็บรายการของ double มีบริการในการเพิ่มต่อท้ายรายการ, ขอข้อมูล และค้นข้อมูล
  private ArrayOfDouble d;    // ภายในใช้อ็อบเจกต์ของ ArrayOfDouble ในการเก็บข้อมูล

  public ListOfDoubles()       { d = new ArrayOfDouble(0); }  // สร้างรายการว่าง ๆ
  public void add(double v)    { d.set(d.length(), v);     }  // นำ v ไปต่อท้ายของรายการ
  public boolean isEmpty()     { return d.length() == 0;   }  // คืนจริงเมื่อเป็นรายการว่าง ๆ ไม่มีข้อมูลเลย
  public int length()          { return d.length();        }
  public double get(int index) { return d.get(index);      }
  public void print()          { d.print();                }  // แสดงข้อมูลต่าง ๆ ในรายการออกทางจอภาพ
  public boolean contains(double v) { return d.indexOf(v) >= 0; } // คืนจริง ถ้ามี v เก็บในรายการ
  //-------------------------------------------------------------------------
  public static void main(String[] args) {
    ListOfDoubles list = new ListOfDoubles();
    System.out.println(list.isEmpty());   // true
    list.add(9); list.add(8); list.add(7);
    System.out.println(list.isEmpty());   // false
    list.print();                         // [9.0, 8.0, 7.0]
    System.out.println(list.contains(9 ) + ", " + list.contains(2)); // true, false
  }
}
```

จงเขียนคลาสใหม่ชื่อ **SetOfDoubles** เป็น subclass ของ **ListOfDoubles** มีรายละเอียดดังแสดงข้างล่างนี้

```
public class SetOfDoubles extends ListOfDoubles {

  public SetOfDoubles() {
  }
  public SetOfDoubles(SetOfDoubles s) {
    for (int i = 0; i < s.length(); i++)
      add(s.get(i));
  }
  public SetOfDoubles(double[] v) {
    // นำข้อมูลทั้งหมดของ v เพิ่มในเซต (อย่าลืม ตัวใดซ้ำกันเกินหนึ่งตัว ให้เก็บแค่ดังเดียว)
  }
  public void add(double v) {
    // เพิ่ม v ในเซต  (ไม่เก็บ v ถ้ามี v อยู่แล้วในเซต)
  }
  public boolean has(double v) {
    return contains(v);
  }
  public SetOfDoubles unionWith(SetOfDoubles s) {
    // คืนเซตใหม่ที่เป็นผลจากการนำเซตเรา union กับเซต s
    SetOfDoubles out = new SetOfDoubles(s);
    for (int i = 0; i < length(); i++)
      out.add(get(i));
    return out;
  }
  public SetOfDoubles intersectionWith(SetOfDoubles s) {
    // คืนเซตใหม่ที่เป็นผลจากการนำเซตเรา intersection กับเซต s

  }
  public boolean equals(SetOfDoubles s) {
    // คืนผลการเปรียบเทียบว่าเซตเรามีสมาชิกเหมือนกับเซต s หรือไม่
  }
  public boolean isSubsetOf(SetOfDoubles s) {
    // คืนจริง ถ้าเซตเราเป็นเซตย่อยของเซต s
  }
  //-----------------------------------------------------------------------------------
  public static void main(String[] args) {
    SetOfDoubles s1 = new SetOfDoubles(new double[] { 1, 2, 3, 1, 2, 4, 5 } );
    SetOfDoubles s2 = new SetOfDoubles(new double[] { 3, 2, 4, 1, 5 } );
    System.out.println(s1.has(4) + ", " + s1.has(-9));         // true, false
    System.out.println(s1.equals(s2) + ", " + s2.equals(s1)); // true, true
    SetOfDoubles result = s1.unionWith(s2);
    result.print();  // [3.0, 2.0, 4.0, 1.0, 5.0]   หมายเหตุ : ลำดับของข้อมูลไม่จำเป็นต้องเหมือนที่แสดงนั้นก็ได้
    result = s1.intersectionWith(s2);
    result.print();  // [1.0, 2.0, 3.0, 4.0, 5.0]   หมายเหตุ : ลำดับของข้อมูลไม่จำเป็นต้องเหมือนที่แสดงนั้นก็ได้
    result = s2.unionWith(new SetOfDoubles(new double[] { 9, 8, 2 } ));
    result.print();  // [9.0, 8.0, 2.0, 3.0, 4.0, 1.0, 5.0]   หมายเหตุ : ลำดับของข้อมูลไม่จำเป็นต้องเหมือนก็ได้
    result = s2.intersectionWith(new SetOfDoubles(new double[] { 9, 8, 2 }));
    result.print();  // [2.0]
    System.out.println( result.isSubsetOf(s2) + ", " + s2.isSubsetOf(result) ); // true, false
  }
}
```