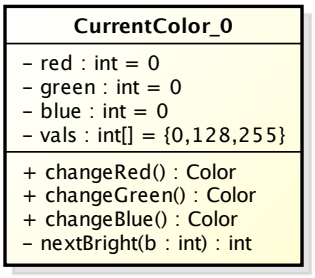
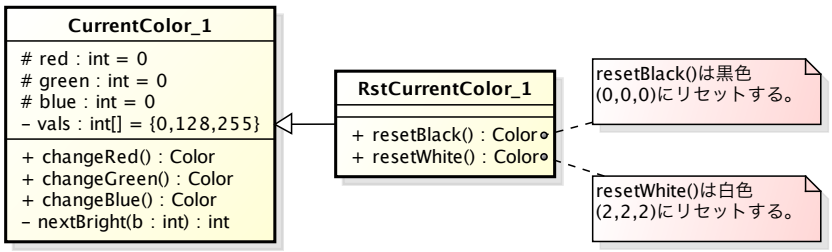
1 現在の色クラスの継承

「現在の色クラス」を継承して白色と黒色へのリセット機能を持った「リセット機能付き現在の色クラス」を作ります．CurrentColor_0 は「現在の色クラス」の一つの実装例です（黒色 (red=0,green=0,blue=0)，白色 (red=2,green=2,blue=2)）が，継承するには不向きな設計になっています．



1.1 属性の可視性を変更する（案１）

CurrentColor_1 は，属性 red，green，blue の可視性を protected に変更し，サブクラスからもアクセスできるように改造したものです．

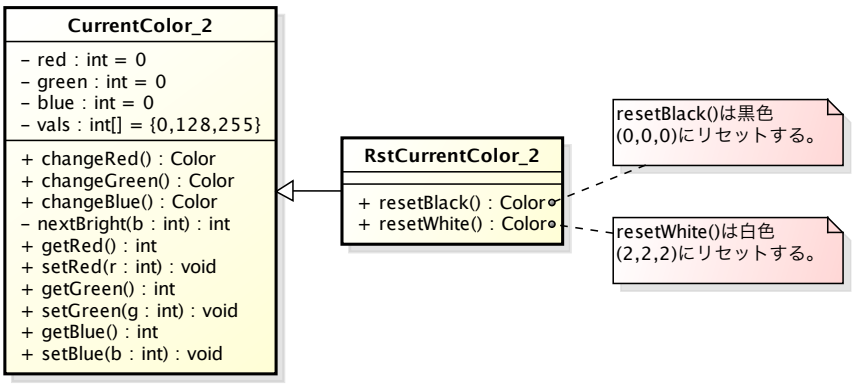


問題：CurrentColor_1 を継承して作った「リセット機能付き現在の色クラス」（RstCurretColor_1）の Java プログラムを書きなさい．（以下の全問題で import は省略して良い．）(10 点)

```
public class RstCurrentColor_1 extends CurrentColor_1 {
  public Color resetBlack() {
    red = green = blue = 0;
    return new Color(0,0,0);
  }
  public Color resetWhite() {
    red = green = blue = 2;
    return new Color(255,255,255);
  }
}
```

1.2 属性のアクセスメソッドを追加する（案２）

CurrentColor_2 は，属性 red，green，blue の可視性は変更しないで属性のアクセスメソッドを追加したものです．



問題：Java クラス RstCurrentColor_2 を書きなさい．(10 点)

```
public class RstCurrentColor_2 extends CurrentColor_2 {
  public Color resetBlack() {
    setRed(0);
    setGreen(0);
    setBlue(0);
    return new Color(0,0,0);
  }
  public Color resetWhite() {
    setRed(2);
    setGreen(2);
    setBlue(2);
    return new Color(255,255,255);
  }
}
```

問題：Java クラス `CurrentColor_2` **のメソッド** `getRed()` **を書きなさい．(5 点)**

```
public int getRed() {
  return red;
}
```

問題：Java クラス `CurrentColor_2` **のメソッド** `setRed()` **を書きなさい．(5 点)**

```
public void setRed(int r) {
  red = r;
}
```

1.3 外部仕様の適正化（案３）

案１，２のどちらも「現在の色クラス」の内部仕様である「属性（red，green，blue）の意味(vals 配列の添字)」がクラスの外に見えていました．そのため色の段階を３から４に変更した場合，「リセット機能付き現在の色クラス」にまで変更が影響します（４段階にした場合は，resetWhite() で値を２ではなく３にする必要があります.）．あるクラスの内部仕様変更が他のクラスに影響を与えるべきではありません．クラスの内部仕様は秘密にし（**カプセス化**し），常に一定の外部仕様を通じて操作できるようにすべきです．

RGB 各色 256 段階で表現するカラーモデルは，ディスプレイやデジタルカメラ等で一般的に用いられます．Java プログラムでもこの RGB カラーモデルを使用できます．そこで,「現在の色クラス」の RGB 各色が一般的な 256 段階であるように外部に見せ，内部が何段階なのかは分からないようにしましょう．このバージョンの「現在の色クラス」を CurrentColor_3 とします．内部表現が n 段階の時，外部表現（e）から内部表現（i）への変換を次の式を用いて行うことにします．

$$i = (e + b) \div a \text{ (但し, } a = 256 \div (n - 1), b = a \div 2)$$

この時，CurrentColor_3 クラスの外部表現と内部表現を変換する機能は次のように記述できます．

```
private int[] vals = {0,128,255};
private final int a = 256 / (vals.length - 1);
private final int b = a / 2;
// 外部表現から内部表現に
private int eToi(int e) {
  return (e + b) / a;
}
// 内部表現から外部表現に
private int iToe(int i) {
  return vals[i];
}
```

問題：Java クラス `CurrentColor_3` **のメソッド** `getRed()` **を書きなさい．(5 点)**

```
public int getRed() {
  return iToe(red);
}
```

問題：Java クラス `CurrentColor_3` **のメソッド** `setRed()` **を書きなさい．(5 点)**

```
public void setRed(int r) {
  red = eToi(r);
}
```

問題：`CurrentColor_3` **を継承する Java クラス** `RstCurrentColor_3` **を書きなさい．(10 点)**

```
public class RstCurrentColor_3 extends CurrentColor_3 {
  public Color resetBlack() {
    setRed(0);
    setGreen(0);
    setBlue(0);
    return new Color(0,0,0);
  }
  public Color resetWhite() {
    setRed(255);
    setGreen(255);
    setBlue(255);
    return new Color(255,255,255);
  }
}
```

2 スタックと FIFO

プログラムを読んで問題に答えなさい.

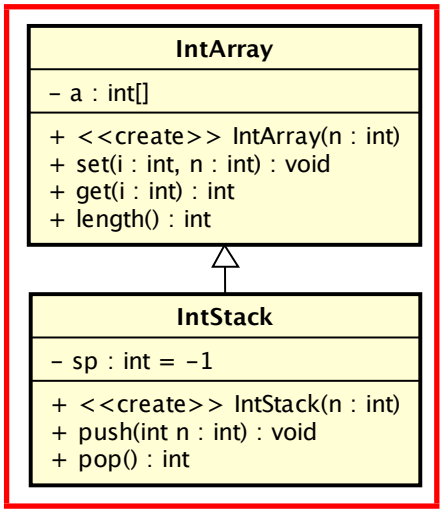
2.1 配列クラスと，配列クラスを継承したスタッククラス

```
1  public class IntArray {
2      private int[] a;
3      public IntArray(int n) {        // コンストラクタ
4          a = new int[n];
5      }
6      public void set(int i, int n) { a[i] = n; }
7      public int get(int i) { return a[i]; }
8      public int length() { return a.length; }
9  }
```

```
1  public class IntStack extends IntArray {
2      private int sp=-1;
3      public IntStack(int n) {    // コンストラクタ
4          super(n);               //   スーパークラスのコンストラクタを呼出す
5      }
6      public void push(int n) {   // スタックにデータを入れる
7          sp++;
8          set(sp, n);
9      }
10     public int pop() {          // スタックからデータを取出す
11         int n = get(sp);
12         sp--;
13         return n;
14     }
15 }
```

問題：上記の二つのクラスと，クラス間の関連を表すクラス図を描きなさい．(10 点)

（但し，コンストラクタは型を省略した通常操作のように書くものする.）



2.2 配列クラスを継承した FIFO クラス

```
1  public class IntFifo extends IntArray {
2      private int head = 0;
3      private int tail = 0;
4      public IntFifo(int n) {     // コンストラクタ
5          super(n);               //   スーパークラスのコンストラクタを呼出す
6      }
7      private int next(int p) {
8          return (p+1)%length();
9      }
10     public void in(int n) {     // FIFO にデータを追加する
11         set(head, n);
12         head = next(head);
13     }
14     public int out() {          // FIFO からデータを取出す
15         int n = get(tail);
16         tail = next(tail);
17         return n;
18     }
19 }
```

問題：以下のプログラムで (1)〜(10) の行について，エラーになる行は「×」，ならない行は「○」を記入しなさい．(各 2 点× 10 問=20 点)

```
// どこかのクラスのプログラムの一部
IntArray a = new IntArray(3);
IntFifo  f = new IntFifo(3);
int l;

a.in(1);          //(1)_____
l=a.next(1);      //(2)_____
a.set(1,1);       //(3)_____
a.a[1]=1;         //(4)_____
l=a.length();     //(5)_____
f.in(1);          //(6)_____
l=f.next(1);      //(7)_____
f.set(1,1);       //(8)_____
f.a[1]=1;         //(9)_____
l=f.length();     //(10)_____
```

2.3 メインクラス

```
1  public class IntMain {
2      IntStack s = new IntStack(3);  // スタック
3      IntFifo f = new IntFifo(3);    // FIFO
4
5      // コンストラクタ
6      private IntMain() {
7          s.push(1);                     // スタックに 1,2,3 を入れる
8          s.push(2);
9          s.push(3);
10
11         f.in(1);                       // FIFO に 1,2,3 を入れる
12         f.in(2);
13         f.in(3);
14
15         System.out.println("--Stack--");  // 順に取出して表示
16         System.out.println(s.pop());
17         System.out.println(s.pop());
18         System.out.println(s.pop());
19
20         System.out.println("--FIFO--");  // 順に取出して表示
21         System.out.println(f.out());
22         System.out.println(f.out());
23         System.out.println(f.out());
24     }
25
26     static public void main(String[] args) {
27         // IntMain のインスタンスを作る
28         new IntMain();
29     }
30 }
```

問題：「$ java IntMain 」の実行結果を書きなさい．(10 点)

```
--Stack--
3
2
1
--FIFO--
1
2
3
```

2.4 全体像

問題：`IntArray`, `IntStack`, `IntFifo`, `IntMain` **クラスの関連が分かるクラス図を描きなさい．(10 点)**

但し，クラスは名前のみ描き，属性と操作は省略して描くこととする．例えば，`IntMain` クラスは次のように描く．