オブジェクト指向言語・期末テスト問題用紙

（2014 年 8 月 01 日・ 10:30 〜 12:00）

解答上、その他の注意事項

I. 問題は、問 I〜VII までである。

II. 解答用紙の右上の欄に学籍番号・名前を記入すること。

III. 解答欄を間違えないよう注意すること。

IV. 解答中の文字 (特に a と d) がはっきりと区別できるよう注意すること。

V. 持ち込みは不可である。筆記用具・時計・学生証以外のものは、かばんの中などにしまうこと。

VI. テストの配点は 80 点である。合格はレポートの得点を加点して、100 点満点中 60 点以上とする。

すべての問に対する補足:
プログラムの空欄を埋める問題では、解答が長くなる可能性があるので、下の省略形（◯囲み文字）を用いても良い。例えば `this==null` と書く代わりに, Ⓣ==Ⓝと書いて良い。（必ず◯で囲むこと。）

Ⓐ `ActionListener` Ⓐ `addActionListener` Ⓒ `class` Ⓓ `actionPerformed`
Ⓔ `ActionEvent` Ⓖ `getSource` Ⓘ `implements` Ⓙ `JApplet` Ⓚ `KeyListener`
Ⓚ `addKeyListener` Ⓜ `MouseListener` Ⓜ `addMouseListener` Ⓝ `null`
Ⓟ `public` Ⓠ `equals` Ⓡ `Runnable` Ⓢ `System.out.println` Ⓣ `this`
Ⓥ `void` Ⓦ `new` Ⓧ `extends`

また、参考のために問題用紙の末尾に授業プリントの KeyTest.java, UpDownButton.java, UpDownButton3.java, BubbleSort1.java, BubbleSort2.java, Point.java, ColorPoint.java のソースを掲載する。

I. 次の各多肢選択問題に答えよ。解答は各問の指示する選択肢から選べ。ただし、特に指定しない限り、選ぶべき選択肢は必ずしも **一つとは限らない**。

(i) 次のうち Java のクラス名として使うことができるのは、どれか?

(A) `A'`　(B) `X_0_7`　(C) `7eleven`　(D) `Ver.2`

(ii) 次の文章のうち正しいものはどれか?

(A). Java アプレットも Java サーブレットも Web ブラウザー上でプログラムが実行されるという特徴は共通している。

(B). Java 仮想機械は Java 専用に設計されており、Java 以外の言語から Java 仮想機械へのコンパイラーは存在しない。

(C). Java では public なクラスの名前と、そのクラスを定義しているファイルの名前は一致させなければならない。

(D). Java と異なり、JavaScript は変数の型宣言を必要としない。型エラーは実行中に検出される。

II. Java で次の実行例のように 1 から 5 までの整数の二乗を出力したい。

`java Test2` の実行結果

```
1 の二乗は 1 です。
2 の二乗は 4 です。
3 の二乗は 9 です。
4 の二乗は 16 です。
5 の二乗は 25 です。
```

次のプログラムに対して、

```
public class Test2 {
    public static void main(String[] args) {
        int n;
        for (n=1 n<=5; n++) {
            (?);
        }
    }
}
```

上記の出力をするために、空欄部分を `System.out.printf` か `System.out.println` で始まる一つの式で埋めよ。

III. packageA.ClassA クラス（packageA パッケージの ClassA クラス）、packageB.ClassB クラス（packageB パッケージの ClassB クラス）、packageA.Main クラス（packageA パッケージの Main クラス）、および packageB.Main クラス（packageB パッケージの Main クラス）をそれぞれ次のように定義する

パス: packageA/ClassA.java

```
package packageA;

public class ClassA {
    (i)  String x;
    (ii) String y;

    public ClassA(String a, String b) {
        x = a; y = b;
    }
}
```

パス: packageB/ClassB.java

```
package packageB;

public class ClassB {
    (iii) String x;
    (iv)  String y;

    public ClassB(String a, String b) {
        x = a; y = b;
    }
}
```

パス: packageA/Main.java

```
1  package packageA;
2
3  import packageB.*;
4
5  public class Main {
6      public static void main(String[] args) {
7          ClassA a = new ClassA("い", "ろ");
8          ClassB b = new ClassB("は", "に");
9
10         System.out.println(a.x);  // エラー
11         System.out.println(a.y);
12         System.out.println(b.x);  // エラー
13         System.out.println(b.y);
14     }
15 }
```

パス: packageB/Main.java

```
1  package packageB;
2
3  import packageA.*;
4
5  public class Main {
6      public static void main(String[] args) {
7          ClassA a = new ClassA("い", "ろ");
8          ClassB b = new ClassB("は", "に");
9
10         System.out.println(a.x);  // エラー
11         System.out.println(a.y);  // エラー
12         System.out.println(b.x);
13         System.out.println(b.y);
14     }
15 }
```

packageA/Main.java の 10, 12 行目と packageB/Main.java の 10, 11 行目（「// エラー」というコメントのある行）がコンパイル時にエラーになるという。(i)〜(iv) の空欄に当てはまる修飾子（あるいは修飾子なし）を以下の (A)〜(C) から選べ。

(A) public　　(B) private　　(C)　　（修飾子なし）

IV. 次の枠内の文章は java.util.ArrayDeque クラスのコンストラクター、push メソッド、pop メソッドと add メソッドの説明の Java™ API 仕様からの抜粋（問題を解くのに関係ない部分は割愛）である。

（参考）両端キューとは両端（先頭と末尾）で挿入と削除をサポートするコレクションで、“deque” は “double ended queue”（両端キュー）の省略形であり、「デック」と発音される。ArrayDeque は両端キューのサイズ変更可能な配列による実装である。この問では ArrayDeque を主にスタックとして使用する。

java.util
クラス **ArrayDeque<E>**

…
public ArrayDeque()
…空の配列両端キューを作成します。

public boolean **add**(E e)
指定された要素をこの両端キューの末尾に挿入します。
…
パラメータ:
e – 追加する要素
…
public void **push**(E e)
この両端キューで表されるスタックに要素を入れます。つまり、要素をこの両端キューの先頭に挿入します。
…
パラメータ:
e –プッシュする要素
…
public E **pop**()
この両端キューで表されるスタックから要素をポップします。つまり、この両端キューの最初の要素を削除して返します。
…
戻り値:
この両端キューの先頭の要素（この両端キューによって表されるスタックの先頭）
…

このメソッドを使用し、テストするプログラムを次のように作成する。

ファイル名: SHanoi.java

```
import java.util.ArrayDeque;

public class SHanoi {
    private static [(i)] a, b, c;

    public static void main(String[] args) {
        a = new [(i)]();
        b = new [(i)]();
        c = new [(i)]();

        int n = [(ii)] > 0 ? [(iii)] : 5;
        initialize(n, a);
        showStatus();
        hanoi(n, a, b, c);
    }

    private static void showStatus() {
        System.out.printf("%s\t%s\t%s%n", a, b, c);
    }

    private static void initialize(int n, [(i)] q) {
        for (int i=1; i<=n; i++) {
            [(iv)]
        }
    }

    private static void hanoi(int n, [(i)] x,
                              [(i)] y, [(i)] z) {
        if (n>0) {
            hanoi(n-1, x, z, y);
            move(x, y);
            showStatus();
            hanoi(n-1, z, y, x);
        }
    }

    private static void move([(i)] x, [(i)] y) {
        [(v)]
    }
}
```

このプログラムは、ハノイの塔 (Tower of Hanoi) のアルゴリズムを、実行途中の３つの塔（スタック）の内容を表示しながら実行する。

実行例は次のようになる。

java SHanoi 4 の実行結果（途中まで）

```
[1, 2, 3, 4]    []      []
[2, 3, 4]       []      [1]
[3, 4]  [2]     [1]
[3, 4]  [1, 2]  []
[4]     [1, 2]  [3]
[1, 4]  [2]     [3]
[1, 4]  []      [2, 3]
[4]     []      [1, 2, 3]
...
```

(i) 上のプログラムの空欄 (i) には、整数型を要素とする ArrayDeque の型（またはコンストラクター）を表す、共通の式（または型の式）が入る。あてはまるものを以下の選択肢 (A)〜(D) から選べ。

(A) `ArrayDeque[int]`　(B) `ArrayDeque[Integer]`
(C) `ArrayDeque<int>`　(D) `ArrayDeque<Integer>`

(ii) 空欄 (ii) にはコマンドライン引数の個数を表す式が入る。正しいものを以下の選択肢 (A)〜(D) から選べ。

(A) `args_length`　(B) `args.length`
(C) `length(args)`　(D) `args[length]`

(iii) 空欄 (iii) には、最初のコマンドライン引数を整数に変換した結果を表す式が入る。正しいものを以下の選択肢 (A)〜(D) から選べ。

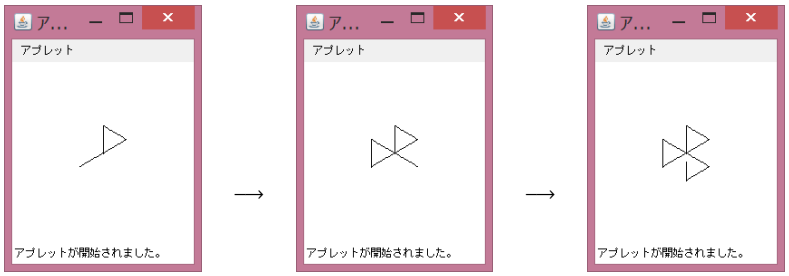
(A) `Integer.parseInt(args[0])`　(B) `atoi(args[0])`
(C) `args[0].toInteger`　(D) `toInteger(args[0])`

(iv) 空欄 (iv) には、両端キュー q の 末尾 に整数 i を追加する文（または文のならび）が入る。この空欄を埋めよ。

(v) 空欄 (v) には、両端キュー x（の先頭）からポップした要素を両端キュー y（の先頭）にプッシュする文（または文のならび）が入る。この空欄を埋めよ。

V. 以下は、 キー入力を受け取り、 “k” キーが押されれば右下（時計盤の 4 時の方向）、 “h” キーが押されれば左下（8 時の方向）、“u” キーが押されれば上（12 時の方向）に、約 10 ピクセル分の線を描画して現在位置を移動する、という Java アプレットのプログラムである。

次のスクリーンショットは “uuukkkhhhhhhuuukkkkkkhhhuu” というキーが入力されたときの画面である。



スクリーンショット

(i)〜(iii) の空欄を埋めて、プログラムを完成させよ。

なお、(ii) のヒントとして Java の主なプリミティブ型とラッパークラスとの対応を以下に挙げる。

プリミティブ型	ラッパークラス
int	Integer
char	Character
double	Double
boolean	Boolean

```
1  import java.util.ArrayList;
2  import java.awt.*;
3  import java.awt.event.*;
4  import javax.swing.*;
5
6  public class KeyDraw [      (i)      ] {
7      ArrayList[  (ii)  ] keys = new ArrayList[ (ii) と同じ ]();
8
9      public void keyPressed(KeyEvent e) {
10         char k = e.getKeyChar();
11         keys.add(k);
12         repaint();
13     }
14     public void keyTyped(KeyEvent e) {}
15     public void keyReleased(KeyEvent e) {}
16
17     @Override
18     public void init() {
19         setFocusable(true);
20         [      (iii)      ];
21     }
```

```
22
23      @Override
24      public void paint(Graphics g) {
25          int i;
26          double x0 = 100, y0 = 100;
27          for (i=0; i < keys.size(); i++) {
28              double x1 = x0, y1 = y0;
29              char k = keys.get(i);
30              switch (k) {
31              case 'h': // left
32                  x1 = x0 - 8.66; y1 = y0 + 5; break;
33              case 'u': // up
34                  y1 = y0-10; break;
35              case 'k': // right
36                  x1 = x0 + 8.66; y1 = y0 + 5;  break;
37              }
38              g.drawLine((int)x0, (int)y0, (int)x1, (int)y1);
39              x0 = x1; y0 = y1;
40          }
41      }
42  }
```

また、同じ動作をするプログラムを、匿名（無名）クラスを利用して、KewDraw2 クラスとして実装する。(iv)〜(v) の空欄を埋めて、プログラムを完成させよ。(v) については労力の節約のため、必要なら「KeyDraw.java の○行めから△行めと同じ」というような書き方をしても良い。

```
1   // import は同一なので省略する
2
3   public class KeyDraw2 [        (iv)        ] {
4       ArrayList [ (ii) と同じ ] keys = new ArrayList [ (ii) と同じ ]();
5
6       @Override
7       public void init() {
8           setFocusable(true);
9           addKeyListener(
                [                                ]
                [              (v)               ]
                [                                ]
10
11          );
12      }
13
14      // paint メソッドは KeyDraw.java と同一なので省略する
15  }
```

VI. 次に定義されるクラス Animal を継承して、

ファイル: animals/Animal.java

```
1  package animals;
2
3  public class Animal {
4      public int age;
5
6      public Animal() { age = 0; }  // 注 1
7
8      public void say() {
9          System.out.print("...");
10         age++;
11     }
12 }
```

3 つのクラス Dog, Cat, Pika を定義する。（pika – ナキウサギ）

ファイル: animals/Dog.java

```
1  package animals;
2
3  public class Dog (i) {
4      public Dog() { super(); }  // 注 1
5
6      @Override
7      public void say() {
8          if (age<1) {
9              System.out.print("Kyan␣");
10         } else {
11             System.out.print("Wan␣");
12         }
13         age++;
14     }
15 }
```

ファイル: animals/Cat.java

```
1  package animals;
2
3  public class Cat (i) {
4      public Cat() { super(); }  // 注 1
5
6      @Override
7      public void say() {
8          System.out.print("Nyaa␣" );
9          age++;
10     }
11 }
```

ファイル: animals/Pika.java

```
1  package animals;
2
3  public class Pika (i) {
4      private boolean female; // false -- 雄, true -- 雌
5
6      public Pika(boolean f) {
7          super();
8          female = f;
9      }
10
11     @Override
12     public void say() {
13         if (female) {
14             System.out.print("Pyuu␣");
15         } else {
16             System.out.print("Kiii␣");
17         }
18         age++;
19     }
20 }
```

注 1: これらのコンストラクターは、実際には定義する必要はない。(デフォルトで定義されるコンストラクターと同等である。)

また、AnimalTest クラスは、これらのクラスのテスト用の main メソッドを持つ。

ファイル: AnimalTest.java

```
1  public class AnimalTest {
2      public static void foo(Animal a) {
3          int i;
4          for(i=0; i<3; i++) {
5              a.say();
6          }
7          System.out.println();
8      }
9
10     public static void main(String[] args) {
11         Dog d  = new Dog();
12         Cat c  = new Cat();
13         Pika p = new Pika(true);
14         p.age++;
15         p.female = false;
16         System.out.println("--␣Dog␣--");
17         foo(d);
18         System.out.println("--␣Cat␣--");
19         foo(c);
20         System.out.println("--␣Pika␣--");
21         foo(p);
```

```
22      }
23  }
```

(i) の空欄（3 箇所で共通）を埋めて、クラスの定義を完成させよ。

(ii) AnimalTest.java の 14 行目「`p.age++;`」と 15 行目「`p.female = false;`」について、当てはまるものを (A) 〜 (C) の選択肢から一つ選べ。

(A) どちらの行もコンパイル時にはエラーにならない。
(B) 「`p.age++;`」のみがコンパイル時にエラーになる。
(C) 「`p.female = false;`」のみがコンパイル時にエラーになる。

(iii) AnimalTest.java の 14, 15 行目どちらの行もコンパイル時にエラーになるようにするためには、どのファイルの何行めをどのように変更すれば良いか？
（行の左端の数字が行番号を表す。）

ヒント： `age` フィールドは `Dog`, `Cat`, `Pika` クラスからはアクセスできる必要がある。なお、この問で定義されるクラスは `AnimalTest` クラス以外、すべて同一パッケージ (`animals`) にある。（`AnimalTest` クラスだけは無名パッケージにある。）

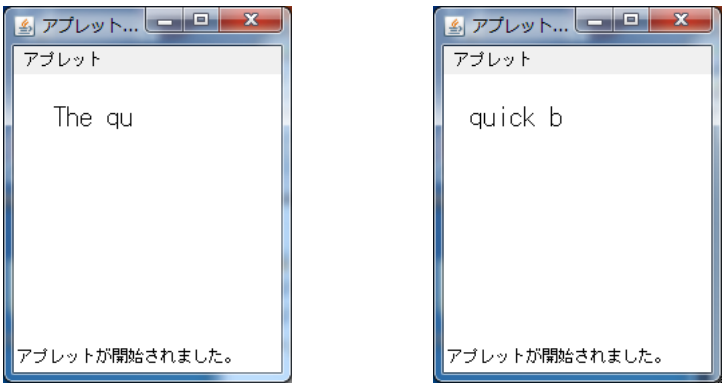
(iv) 上の問でエラーになるようにした AnimalTest.java の「`p.age++;`」と「`p.female = false;`」の 2 行を コメントアウト し、このクラスの `main` メソッドを実行するとき、出力はどうなるか？（解答用紙に記入するとき、空白の有無は気にしなくて良い。）

VII. 下のプログラムは、マウスをクリックするたびに、“　　The quick brown fox jumps over the lazy dog.　　”という文字列から、8 文字の部分文字列を先頭から順番に取り出して表示する “電光掲示板” アプレットである。

ファイル: Denko.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Denko extends JApplet implements MouseListener {
    String message = "␣␣␣␣␣␣␣The␣quick␣brown␣fox␣jumps␣over␣the␣lazy␣dog.␣␣␣␣␣␣␣";
    int i=0;

    @Override
    public void init() {
        addMouseListener(this);
    }

    @Override
    public void paint(Graphics g) {
        g.setFont(new Font("Monospaced", Font.PLAIN, 20));
        g.drawString(message.substring(i, i+8), 10, 40);
    }

    public void mouseClicked(MouseEvent e) {
        i++;
        if (i > message.length()-8) { i=0; }
        repaint();
    }
    public void mousePressed(MouseEvent e) {}  /* 5 */
    public void mouseReleased(MouseEvent e) {} /* 5 */
    public void mouseEntered(MouseEvent e) {}  /* 5 */
    public void mouseExited(MouseEvent e) {}   /* 5 */
}
```

次にスクリーンショットを示す。



(1) 5 回クリックしたところ　(2) さらに、5 回クリックしたところ

このプログラムを参考にして、マウスクリックではなく時間経過で表示が変化するようなアニメーションのアプレットを作成する。スレッドを用いて、約 100 ミリ秒間隔で表示が更新されるようにする。以下の (i)〜(iii) の空欄を埋めてプログラムを完成させよ。

ファイル: Denko2.java

```
import java.awt.*;
import javax.swing.*;

public class Denko2 [      (i)      ] {
    Thread myThread = null;
    String message = "␣␣␣␣␣␣␣The␣quick␣brown␣fox␣jumps␣over␣the␣lazy␣dog.␣␣␣␣␣␣␣";
    int i=0;

    @Override
    public void start() {
        if (myThread==null) {
            myThread = [      (ii)      ];
            myThread.start();
        }
    }

    @Override
    public void stop() {
        myThread = null;
    }

    @Override
    public void paint(Graphics g) {
        g.setFont(new Font("Monospaced", Font.PLAIN, 20));
        g.drawString(message.substring(i, i+8), 10, 40);
    }

    public void run() {
        Thread thisThread = Thread.currentThread();
        while([      (iii)      ]) {
            i++;
            if (i > message.length()-8) { i=0; }
            repaint();
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {}
        }
    }
}
```

以下に参考のために授業配布プリントの KeyTest.java, UpDownButton.java, UpDownButton3.java, BubbleSort1.java, BubbleSort2.java, Point.java, ColorPoint.java のソースを掲載する。

KeyTest.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class KeyTest extends JApplet implements KeyListener {
  int x=50, y=20;

  @Override
  public void init() {
    setFocusable(true);
    addKeyListener(this);
  }

  @Override
  public void paint(Graphics g) {
    super.paint(g);
    g.drawString("HELLO␣WORLD!", x, y);
  }

  public void keyTyped(KeyEvent e) {
    int k = e.getKeyChar();
    if (k=='u') {
      y-=10;
    } else if (k=='d') {
      y+=10;
    }
    repaint();
  }
  public void keyReleased(KeyEvent e) {}
  public void keyPressed(KeyEvent e) {}
}
```

UpDownButton.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class UpDownButton extends JApplet implements ActionListener {
  int x=20;
  JButton lBtn, rBtn;

  @Override
  public void init() {
    lBtn = new JButton("Left");   rBtn = new JButton("Right");
    lBtn.addActionListener(this); rBtn.addActionListener(this);
    setLayout(new FlowLayout());
    add(lBtn);                    add(rBtn);
  }

  @Override
  public void paint(Graphics g) {
    super.paint(g);
    g.drawString("HELLO␣WORLD!", x, 55);
  }

  public void actionPerformed(ActionEvent e) {
    Object source = e.getSource();
    if (source == lBtn) {         // lBtnが押された
      x-=10;
    } else if (source == rBtn) {  // rBtnが押された
```

```
      x+=10;
    }
    repaint();
  }
}
```

UpDownButton3.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class UpDownButton3 extends JApplet {
  int x=20;

  @Override
  public void init() {
    JButton lBtn = new JButton("Left");
    JButton rBtn = new JButton("Right");
    lBtn.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        x-=10;
        repaint();
      }
    });
    rBtn.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        x+=10;
        repaint();
      }
    });
    setLayout(new FlowLayout());
    add(lBtn); add(rBtn);
  }

  @Override
  public void paint(Graphics g) {
    super.paint(g);
    g.drawString("HELLO␣WORLD!", x, 55);
  }
}
```

BubbleSort1.java

```
import javax.swing.*;
import java.awt.*;

public class BubbleSort1 extends JApplet implements Runnable {
  int[] args = { 10, 3, 46, 7, 23, 34, 8, 12, 4, 45, 44, 52};
  Color[] cs = { Color.RED, Color.ORANGE, Color.GREEN, Color.BLUE};
  Thread thread = null;

  @Override
  public void start() {
    if (thread == null) {
      thread = new Thread(this);
      thread.start();
    }
  }

  @Override
  public void stop() {
    thread = null;
  }

  @Override
```

```
  public void paint(Graphics g) {
    int i;
    super.paint(g);
    for(i=0; i<args.length; i++) {
      g.setColor(cs[args[i]%cs.length]);
      g.fillRect(0, i*10, args[i]*5, 10);
    }
  }

  public void run() {
    int i, j;
    Thread thisThread = Thread.currentThread();

    for (i=0; i<args.length-1; i++) {
      for (j=args.length-1; thread == thisThread && j>i; j--) {
        if (args[j-1]>args[j]) { // スワップする。
          int tmp=args[j-1]; args[j-1]=args[j]; args[j]=tmp;
        }
        repaint();
        try { // repaint の後でしばらく止まる
          Thread.sleep(500);
        } catch (InterruptedException e) {}
      }
    }
  }
}
```

BubbleSort2.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class BubbleSort2 extends JApplet implements Runnable, ActionListener {
  int[] args = { 10, 3, 46, 7, 23, 34, 8, 12, 4, 45, 44, 52};
  Color[] cs = {Color.RED, Color.ORANGE, Color.GREEN, Color.BLUE};
  Thread thread = null;
  private boolean threadSuspended = true;
  // start, stop, paint メソッドは BubbleSort1.java と同一なので省略する。

  @Override
  public void init() {
    JButton step = new JButton("Step");
    step.addActionListener(this);
    setLayout(new FlowLayout());
    add(step);
  }

  public synchronized void actionPerformed(ActionEvent e) {
    threadSuspended = false;
    notify();
  }

  public void run() {
    int i, j;
    for (i=0; i<args.length-1; i++) {
      for (j=args.length-1; j>i; j--) {
        if (args[j-1]>args[j]) { // スワップする。
          int tmp=args[j-1]; args[j-1]=args[j]; args[j]=tmp;
        }
        repaint();
        try {  // repaint の後で止まる
          synchronized(this) {
            while (threadSuspended) {
              wait();
```

```
            }
            threadSuspended = true;
          }
        } catch (InterruptedException e) {}
      }
    }
    thread = null;
  }
}
```

Point.java

```
public class Point {
  public int x, y;

  public void move(int dx, int dy) {
    x += dx;  y += dy;
  }

  public double distance() {
     return Math.sqrt(x*x+y*y);
  }

  public void print() {
    System.out.printf("(%d,␣%d)", x, y);
  }

  public void moveAndPrint(int dx, int dy) {
    print(); move(dx, dy); print();
  }

  public Point(int x0, int y0) {
    x = x0; y = y0;
  }
}
```

ColorPoint.java

```
public class ColorPoint extends Point {
  private String[] cs = {"black", "red", "green", ..., "white"};
  private String color;

  @Override
  public void print() {
    System.out.printf("<font␣color='%s'>", getColor());  // 色の指定
    super.print();
    System.out.print("</font>");                          // 色を戻す
  }

  public void setColor(String c) {
    int i;
    for (i=0; i<cs.length; i++) {
      if (c.equals(cs[i])) {
        color = c; return;
      }
    } // 対応する色がなかったら何もしない。
  }

  public ColorPoint(int x, int y, String c) {
    super(x, y);
    setColor(c);
    if (color==null) color = "black";
  }

  public String getColor() { return color; }
}
```

オブジェクト指向言語・期末テスト解答用紙（2014年8月01日）

学籍番号		氏名	

I. （3 × 2）

(i).		(ii).	

II. （4）

III. （2 × 4）

(i).		(ii).		(iii).		(iv).	

IV. （3, 3, 3, 4, 5）

(i).		(ii).		(iii).	
(iv).					
(v).					

V. （3, 3, 3, 3, 5）

(i).	
(ii).	
(iii).	
(iv).	
(v).	

VI. （4, 4, 4, 6）

(i).	
(ii).	
(iii).	
(iv).	

VII. （3 × 3）

(i).	
(ii).	
(iii).	

授業・テストの感想

……

……

……

……

……

……

……

……

……

……