オブジェクト指向言語・期末テスト問題用紙

（2015 年 7 月 31 日・ 10:30 〜 12:00）

（問題訂正適用済み）

訂正は赤字

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, Factorial.java, UpDownButton.java, UpDownButton4.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) `Foo-Bar` (B) `13T299` (C) `Chanel#5` (D) `Ans_1_2_1`

(ii) 次の文章のうち正しいものはどれか?

(A). Java は中間言語へコンパイルされ、コンパイルされたプログラムは Java がサポートする全ての機種で動作する。

(B). HTML 中に埋め込まれた JavaScript のプログラムは Web サーバー側で実行され、結果がクライアント側（Web ブラウザー）に送られる。

(C). Prolog は第五世代コンピュータープロジェクトで注目を浴びた言語で、関数型言語に分類される。

(D). Java はオブジェクト指向言語の特徴の一つである動的束縛を提供せず、その分、実行の高速化を図っている。

II. Java で次の実行例のようにコマンドライン引数として与えられた整数の 25 倍を表示したい。

`java Test2 40` の実行結果

```
40 の 25 倍は 1000 です。
```

`java Test2 25` の実行結果

```
25 の 25 倍は 625 です。
```

次のプログラムに対して、

ファイル名: Test2.java

```
public class Test2 {
    public static void main(String[] args) {
        String a = args[0];  // コマンドライン引数
        int n = (i);
        (ii);
    }
}
```

上記の出力をするために、空欄部分 (i), (ii) を式で埋めよ。ただし、(i) は、文字列 `a` を整数に変換する式である。(ii) は `n` と `n*25` を出力するため、`System.out.printf` または `System.out.println` を使う式である。なお、出力中の空白の有無は気にしなくてよい。

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("abc", "def");
8          ClassB b = new ClassB("ghi", "jkl");
9
10         System.out.println(a.x);
11         System.out.println(a.y);
12         System.out.println(b.x);  // エラー
13         System.out.println(b.y);  // エラー
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("mno", "pqr");
8          ClassB b = new ClassB("stu", "vwx");
9
10         System.out.println(a.x);
11         System.out.println(a.y);  // エラー
12         System.out.println(b.x);  // エラー
13         System.out.println(b.y);
14     }
15 }
```

packageA/Main.java の 12, 13 行目と packageB/Main.java の 11, 12 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.lang.String クラスの startsWith メソッドと endsWith 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。

java.lang
クラス **String**

...

public boolean **startsWith**(String prefix)

この文字列が、指定された接頭辞で始まるかどうかを判定します。

パラメータ:

prefix – 接頭辞。

戻り値:

引数によって表される文字シーケンスが、この文字列によって表される文字シーケンスの接頭辞である場合は true、そうでない場合は false。引数が空の文字列の場合や、equals(Object) メソッドによる判定においてこの String オブジェクトに等しい場合にも true が返される。

public boolean **endsWith**(String suffix)

この文字列が、指定された接尾辞で終るかどうかを判定します。

パラメータ:

suffix – 接尾辞。

戻り値:

引数によって表される文字シーケンスが、このオブジェクトによって表される文字シーケンスの接尾辞である場合は true、そうでない場合は false。引数が空の文字列の場合や、equals(Object) メソッドによる判定においてこの String オブジェクトに等しい場合の結果は true になる。

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: StringTest.java

```
public class StringTest {
    public static void main(String[] args) {
        String prefix = args[0];
        String suffix = args[1];
        int i;
        for (i=2; i<      (i)      ; i++) {
            String str = args[i];
            if (            (ii)            ) {
                System.out.print("!!");
            }
            System.out.print(str);
            if (            (iii)           ) {
                System.out.print("??");
            }
            System.out.println();
        }
    }
}
```

このプログラムは、java StringTest *prefix suffix str1 str2 ...strN* という形で使用し、*str1, str2, ..., strN* を 1 行ずつ

- prefix から始まるならば、先頭に!!を追加して、
- suffix で終了するならば、末尾に??を追加して、

表示する。実行例は次のようになる。

java StringTest in tion instructor inflation imagination illusion の実行結果

```
!!instructor
!!inflation??
imagination??
illusion
```

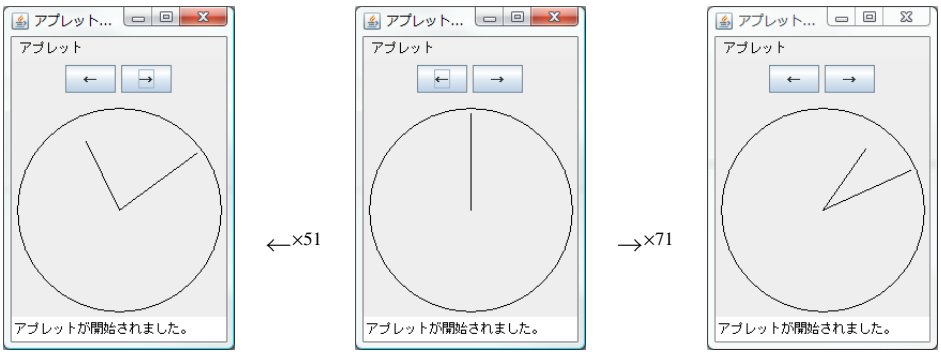
(i) 空欄 (i) にはコマンドライン引数の個数を表す式が入る。正しいものを以下の選択肢 (A)〜(D) から選べ。

(A) #(args)　(B) args.length　(C) length(args)　(D) #.args

(ii) 空欄 (ii) には、文字列 str の先頭が prefix で始まるかどうかを判定する式が入る。この空欄を埋めよ。

(iii) 空欄 (iii) には、文字列 str の末尾が suffix で終わるかどうかを判定する式が入る。この空欄を埋めよ。

V. 次のプログラム（ToyClock クラス）は、“←”, “→” という 2 つのボタンと、時計の針を表示し、“←” ボタンを押せば時計が一分戻り、“→” ボタンを押せば時計が一分進む、という “おもちゃの時計” アプレットである。



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。

ファイル名: ToyClock.java

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

public class ToyClock [        (i)        ] {
    JButton cBtn, aBtn;
    int m = 0;

    @Override
    public void init() {
        aBtn  = new JButton("←");
        cBtn  = new JButton("→");
        [              (ii)              ]
        setLayout(new FlowLayout());
        add(aBtn); add(cBtn);
    }

    public void actionPerformed(ActionEvent aEvt) {
        Object source = [      (iii)      ];
        if (source == aBtn) {
            m--;
        } else if (source == cBtn) {
            m++;
        }
        repaint();
    }
```

```
    @Override
    public void paint(Graphics g) {
        super.paint(g);
        int cx = 100, cy = 140;                    // 中心の座標
        g.drawOval(cx-95, cy-95, 190, 190);        // 縁
        double angle1 = m * 6 / 180.0 * Math.PI;   // 1分で長針は 6度動く
        int x1 = (int)(cx+90*Math.sin(angle1));
        int y1 = (int)(cy-90*Math.cos(angle1));
        g.drawLine(cx, cy, x1, y1);                // 長針
        double angle2 = m * 0.5 / 180 * Math.PI;   // 1分で短針は 0.5度動く
        int x2 = (int)(cx+70*Math.sin(angle2));
        int y2 = (int)(cy-70*Math.cos(angle2));
        g.drawLine(cx, cy, x2, y2);                // 短針
    }
}
```

また、同じ動作をするプログラムを、ラムダ式を利用して、ToyClock2 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。

ファイル名: ToyClock2.java

```
// import は ToyClock.java と同一なので省略する

public class ToyClock2 [      (iv)      ] {
    int m = 0;

    @Override
    public void init() {
        JButton aBtn  = new JButton("←");
        JButton cBtn  = new JButton("→");
        aBtn.addActionListener( [    (v-1)    ] );
        cBtn.addActionListener( [    (v-2)    ] );
        setLayout(new FlowLayout());
        add(aBtn); add(cBtn);
    }

    /* paintメソッドは ToyClockクラスと同一なので省略する */
}
```

VI. Point クラス（問題用紙の末尾のソース参照）を継承して、CoarsePoint クラスを定義する。CoarsePoint クラスは、1 つの新しい int 型のフィールド unit を持つ。print メソッドは再定義されていて、x, y の正確な値の代わりに、それよりも小さい unit の倍数のうち、正確な値にもっとも近いものを表示する。例えば unit が 4 で x が 9, かつ y が 15 の時、(8, 12) と表示する。（ここでは、x, y, unit が負の数の場合は考慮しない。）

(i)〜(ii) の空欄を埋めて CoarsePoint クラスの定義を完成させよ。

ファイル名: CoarsePoint.java

```
1  public class CoarsePoint [ (i) ] {
2      private int unit;
3
4      public CoarsePoint(int x0, int y0, int u0) {
5          super(x0, y0);  unit = u0;
6      }
7
8      @Override
9      public void print() {
           [ (ii) ]
10
11     }
12 }
```

さらに次のようなテスト用プログラム CoarsePointTest.java があるとする。

ファイル名: CoarsePointTest.java

```
1  public class CoarsePointTest {
2      public static void main(String[] args) {
3          Point p1       = new Point(2, 4);
4          CoarsePoint p2 = new CoarsePoint(13, 17, 5);
5          // p2.unit = 4; /* このままではコンパイルエラー */
6          Point[] pts = new Point[] {p1, p2};
7          int i;
8          for (i=0; i<pts.length; i++) {
9              pts[i].print();
10         }
11     }
12 }
```

これについて、以下の文章の (iii) 〜 (v) の空欄を埋めよ。

「CoarsePointTest.java を実行すると “ (iii) ” と出力する。さらに、CoarsePointTest.java の 5 行目のコメントアウトされた行を有効にする（つまり、// を消去する）と、コンパイルエラーになるが、これは、もし CoarsePoint.java の (iv-1) 行目の (iv-2) を (iv-3) とエラーではなくなる。このように変更したときは、CoarsePointTest.java を実行すると、“ (v) ” と出力する。」

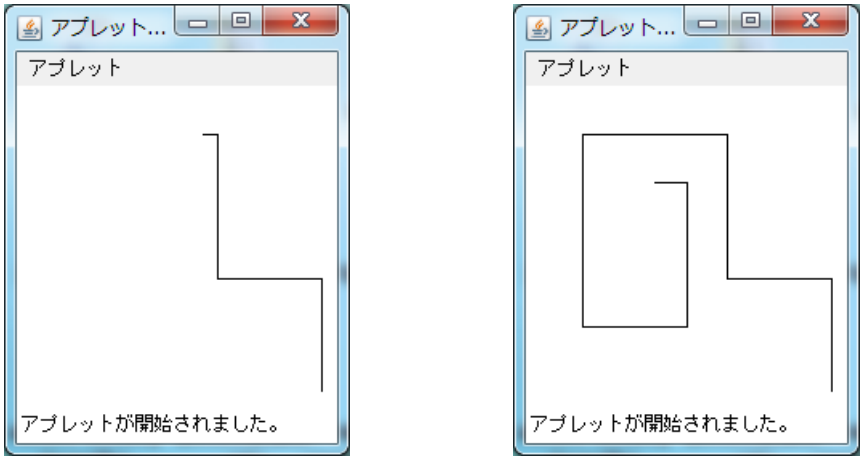
VII. 下のプログラムは、キーボードから打ち込まれた 'h'（左）, 'j'（下）, 'k'（上）, 'l'（右）キーに従って、向きを変え、時間とともにのびていく折れ線のアニメーションを描画する Java アプレットである。

ファイル: TronModoki.java

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4  import java.util.ArrayList;
5
6  public class TronModoki (i) {
7    private (ii) keys = new ArrayList<>();
8    private char last = 'k';
9    private Thread myThread = null;
10
11   @Override
12   public void start() {
13     if (myThread==null) {
14       myThread = new Thread(this);
15       myThread.start();
16     }
17   }
18
19   @Override
20   public void stop() {
21     myThread = null;
22   }
23
24   @Override
25   public void init() {
26     setFocusable(true);
27     (iii)
28   }
29
30   @Override
31   public void paint(Graphics g) {
32     int x = 190, y = 190;
33     final int step = 5;
34     for (char k : keys) {
35       int x1 = x, y1 =y;
36       switch(k) {
37       case 'h': x1-=step; break;
38       case 'j': y1+=step; break;
39       case 'k': y1-=step; break;
40       case 'l': x1+=step; break;
41       default: break;
42       }
43       g.drawLine(x, y, x1, y1);
44       x = x1; y = y1;
```

```
45      }
46    }
47
48    public void run() {
49      while (myThread==Thread.currentThread()) {
50        keys.add(last);
51        repaint();
52        try {
53          Thread.sleep(300);
54        } catch (InterruptedException e) {}
55      }
56    }
57
58    public void keyPressed(KeyEvent e) {}
59    public void keyReleased(KeyEvent e) {}
60    public void keyTyped(KeyEvent e) {
61      char c = e.getKeyChar();
62      if ("hjkl".indexOf(c) >= 0) {
63        last = c;
64      }
65    }
66  }
```

次にスクリーンショットを示す。



(1) 'h', 'k', 'h' と打ったところ　(2) さらに、いくつかのキーを打ったところ

(i)〜(iii) の空欄を埋めてプログラムを完成させよ。なお、(ii) は、フィールド keys の型が入る。keys は、34 行目、50 行目で使われている。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

以下に参考のために授業配布プリントの KeyTest.java, Factorial.java, UpDownButton.java, UpDownButton4.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) {}
}
```

Factorial.java

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

public class Factorial extends JApplet implements ActionListener {
  JTextField input;
  JLabel output;

  @Override
  public void init() {
    input=new JTextField("0", 8);
    output=new JLabel("␣␣␣1");
    input.addActionListener(this);
    setLayout(new FlowLayout());
    add(input);  add(new JLabel("の階乗は"));
    add(output); add(new JLabel("です。"));
  }

  static int factorial(int n) { // factorial -- 階乗のこと
    int r = 1;
    for (; n>0; n--) {
      r *= n;
    }
    return r;
  }

  public void actionPerformed(ActionEvent e) {
```

```
    try {
      int n = Integer.parseInt(input.getText());
      output.setText("␣␣"+factorial(n));
    } catch (NumberFormatException ex) {
      input.setText("数値!");
    }
  }
}
```

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();
  }
}
```

UpDownButton4.java

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

public class UpDownButton4 extends JApplet {
  int x=20;

  @Override
  public void init() {
    JButton lBtn = new JButton("Left");
    JButton rBtn = new JButton("Right");
    lBtn.addActionListener(e -> { x-=10; repaint(); });
    rBtn.addActionListener(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（2015年7月31日）

学籍番号		氏名	

I. （3 × 2）

(i).		(ii).	

II. （2 × 2）

(i).	
(ii).	

III. （2 × 4）

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

IV. （3 × 3）

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

V. （3, 4, 4, 3, 6）

(i).	
(ii).	
(iii).	
(iv).	
(v-1).	
(v-2).	

VI. （4, 4, 5, 5, 4）

(i).	
(ii).	
(iii).	
(iv).	行目の　　を
(v).	

VII. （4, 4, 3）

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

授業・テストの感想

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……