オブジェクト指向言語・中間テスト問題用紙

（2018 年 06 月 08 日・ 11:00 〜 12:00）

解答上、その他の注意事項

I. 問題は、問 I〜III まである。

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。（中間テスト 40 点・期末テスト 40 点）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。（必ず◯で囲むこと。）

Ⓐ `ActionListener`　ⒶⒶ `addActionListener`　ⒶⒺ `ActionEvent`
Ⓚ `KeyListener`　ⒶⓀ `addKeyListener`　ⓀⒺ `KeyEvent`
Ⓜ `MouseListener`　ⒶⓂ `addMouseListener`　ⓂⒺ `MouseEvent`
ⓟⓛ `System.out.println`　ⓟⓕ `System.out.printf`

また、参考のために問題用紙の末尾に授業配布プリントの MouseTest.java, LeftRightButton3.java, LeftRightButton4.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 Foo という、（無名パッケージに属して public な）クラスをコンパイルし、実行するためのコマンドの組み合わせはどれか？一つ選べ。

(A).
```
javac Foo.java
java Foo.class
```
(B).
```
javac Foo.java
java Foo
```
(C).
```
javac Foo
java Foo
```
(D).
```
javac Foo
java Foo.class
```

(ii) 次の文章のうち正しいものはどれか?

(A). Java コンパイラーは、JavaScript のプログラムもコンパイルすることが可能である。

(B). Java コンパイラーは、C のプログラムもコンパイルすることが可能である。

(C). Java は、メモリを自動的に回収するゴミ集めという技術を標準として取り入れている。

(D). Java は、グラフィックスに関する機能を言語に標準のライブラリーとしては持っていない。

II. 次の枠内の文章は java.awt.Polygon クラスのいくつかのメソッドと、関連する java.awt.Graphics クラスのメソッドの 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.awt
クラス **Polygon**

Polygon クラスは、座標空間内の閉じられた 2 次元領域をカプセル化します。...

コンストラクターの詳細
`public` **Polygon**()
空の多角形を作成します。

メソッドの詳細
`public void` **addPoint**(`int x, int y`)
この Polygon に指定された座標を追加します。
パラメーター:
x - 指定された X 座標
y - 指定された Y 座標

`public boolean` **contains**(`int x, int y`)
指定された座標がこの Polygon の内側にあるかどうかを判定します。
パラメーター:
x - テストされる指定された X 座標
y - テストされる指定された Y 座標
戻り値:
この `Polygon` に、指定された座標 `(x,y)` が含まれる場合は `true`、それ以外の場合は `false`。

java.awt
クラス **Graphics**

...

メソッドの詳細
`public void` **fillPolygon**(`Polygon p`)
指定された Polygon オブジェクトで定義された多角形をグラフィックス・コンテキストの現在の色で塗りつぶします。...
パラメーター:
p - 塗りつぶし対象の多角形。

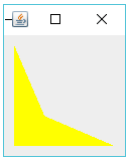
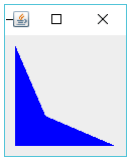
これらのメソッドを使用するプログラムを次のように作成する。

ファイル名: PolygonTest.java

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4
5  public class PolygonTest extends JPanel (i) {
6    private Polygon polygon;
7    private Color color = Color.YELLOW;
8
9    public PolygonTest() {
10     setPreferredSize(new Dimension(120, 120));
11
12     polygon = new Polygon();
13     (ii-1)
14     (ii-2)
15     (ii-3)
16     (ii-4)
17
18     addMouseListener(this);
19   }
20
21   @Override
22   protected void paintComponent(Graphics g) {
23     super.paintComponent(g);
24     g.setColor(color);
25     g.fillPolygon(polygon);
26   }
27
28   public void mouseClicked(MouseEvent e) {
29     int x = e.getX(), y = e.getY();
30
31     if ( (iii) ) {
32       color = Color.YELLOW;
33     } else {
34       color = Color.BLUE;
35     }
36     repaint();
37   }
38
39   /* 他の MouseListener のメソッドと main メソッドは割愛 */
40 }
```

このプログラムは (10, 10) – (10, 110) – (110, 110) – (40, 80) の 4 点を頂点とする多角形を描画し、マウスを多角形の内部でクリックすると、多角形の塗り潰しの色を黄色（YELLOW）に、外部でクリックすると青色（BLUE）に変更する。

実行例は次のようになる。

内部をクリックした時　外部をクリックした時

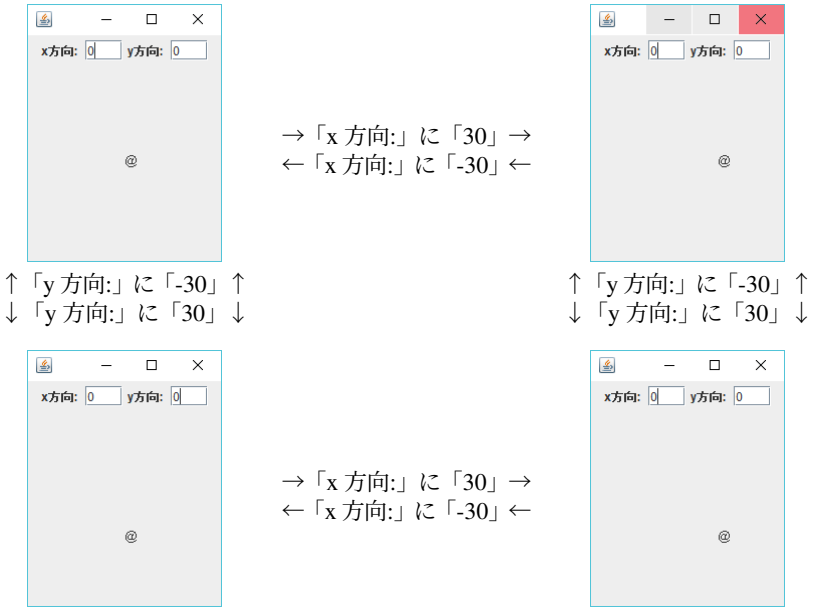
- 空欄 (i) を埋めよ。
- 空欄 (ii-1)〜(ii-4) には、`polygon` に (10, 10) – (10, 110) – (110, 110) – (40, 80) の 4 つの頂点を追加するための、4 つの文が入る。この空欄を埋めよ。
- 空欄 (iii) には、点 (x, y) が多角形 `polygon` の内側に入っているかどうかを判定する式が入る。この空欄を埋めよ。

III. 次はテキストフィールドを 2 つ持ち、最初は (96, 128) に “@” を表示していて、

- 「x 方向: 」というラベルの付いているほうのテキストフィールドに整数を入力すると、x 方向にその数値ぶんだけ移動する、
- 「y 方向: 」というラベルの付いているほうのテキストフィールドに整数を入力すると、y 方向にその数値ぶんだけ移動する、

という Java プログラムである。

実行例は次のようになる。



ファイル名: StringMover.java

```
1  import java.awt.*;
2  import java.awt.event.*;
3
4  import javax.swing.*;
5
6  public class StringMover extends JPanel implements ActionListener {
7    private JTextField xTF, yTF;
8    private int x = 96, y = 128;
9
10   public StringMover() {
11     setPreferredSize(new Dimension(192, 224));
12     add(new JLabel("x␣方向:␣"));
13     xTF = new JTextField("0", 3);
14     add(xTF);
15     add(new JLabel("y␣方向:␣"));
16     yTF = new JTextField("0", 3);
17     add(yTF);
18
```

```
19        (i-1)
20        (i-2)
21    }
22
23    public void actionPerformed(ActionEvent e) {
24      Object source = e.getSource();
25      if (source == xTF) {
26        int dx = Integer.parseInt(xTF.getText());
27        x += dx;
28        xTF.setText("0");
29      } else if (source == yTF) {
30        int dy = Integer.parseInt(yTF.getText());
31        y += dy;
32        yTF.setText("0");
33      }
34      repaint();
35    }
36
37    @Override
38    protected void paintComponent(Graphics g) {
39      super.paintComponent(g);
40      g.drawString("@", x, y);
41    }
42    /* main メソッドは割愛 */
43  }
```

- 空欄 (i-1)〜(i-2) にはあわせて 2 つの文が入る。この空欄を埋めよ。

さらにこの StringMover.java を匿名クラス・ラムダ式を用いて次のように同等のプログラム StringMover2.java に書き換える。

ファイル名: StringMover2.java

```
1   import java.awt.*;
2   import java.awt.event.*;
3
4   import javax.swing.*;
5
6   public class StringMover2 (ii) {
7     /* StringMover.java の 7〜8 行目と同一のため省略 */
8     public StringMover2() {
9       /* StringMover.java の 11〜17 行目と同一のため省略 */
10
11      xTF.addActionListener(
12          (iii)
13        );
14      yTF.addActionListener(
15          (iv)
16        );
```

```
17    }
18
19    /* paintComponent, main メソッドは割愛 */
20  }
```

- 空欄 (ii) を埋めよ。
- 空欄 (iii), (iv) を埋めて、StringMover2 のコンストラクターの定義を完成させよ。ただし、**空欄 (iii) には匿名クラスを、空欄 (iv) にはラムダ式を使用せよ。**適宜、`/* StringMover.java の○〜△行目と同じ */` のように省略してよい。

以下に参考のために授業配布プリントの MouseTest.java, LeftRightButton3.java, LeftRightButton4.java, のソースを掲載する。（main メソッドは省略している。）

MouseTest.java

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

public class MouseTest extends JPanel implements MouseListener {
  private int x = 50, y = 20;

  public MouseTest() {
    setPreferredSize(new Dimension(150, 150));
    addMouseListener(this);
  }

  public void mouseClicked(MouseEvent e) {
    x = e.getX();
    y = e.getY();
    repaint();
  }

  public void mousePressed(MouseEvent e) {}
  public void mouseReleased(MouseEvent e) {}
  public void mouseEntered(MouseEvent e) {}
  public void mouseExited(MouseEvent e) {}

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

  /* main メソッドは割愛 */
}
```

LeftRightButton3.java

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

public class LeftRightButton3 extends JPanel {
  private int x = 20;

  public LeftRightButton3() {
    setPreferredSize(new Dimension(200, 70));
    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 paintComponent(Graphics g) {
    super.paintComponent(g);
    g.drawString("HELLO␣WORLD!", x, 55);
  }

  /* main メソッドは割愛 */
}
```

LeftRightButton4.java

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

public class LeftRightButton4 extends JPanel {
  private int x = 20;

  public LeftRightButton4() {
    setPreferredSize(new Dimension(200, 70));
    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 paintComponent(Graphics g) {
    super.paintComponent(g);
    g.drawString("HELLO␣WORLD!", x, 55);
  }

  /* main メソッドは割愛 */
}
```

オブジェクト指向言語・中間テスト解答用紙（2018 年 06 月 08 日）

学籍番号		氏名	

I. （3 × 2）

(i).		(ii).	

II. （4, 6, 6）

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

III. （4, 4, 5, 5）

(i).	
(ii).	
(iii).	/* 匿名クラス使用 */ xTF.addActionListener( );</br>

（裏面に続く）

(iv).	/* ラムダ式使用 */ yTF.addActionListener( );

授業・テストの感想

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....
