```java
////////////////////////////////////////////////////////////
// 課題１　次のようにマウスのカーソルに同期しメッセージを…
////////////////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

class Assignment12_1 extends Frame implements MouseListener
{
  // メッセージ番号
  private int msgno=0;
  // メッセージ内容
  private String[] msgs={
      "クリックされました",
      "画面上に入りました",
      "画面から出ました",
      "ボタンが押されました",
      "ボタンが離されました"};

  // コンストラクタ
  public Assignment12_1(){
    // タイトル設定
    super("課題１");
    // リスナー登録
    addMouseListener(this);
    addWindowListener(new SimpleWindowAdapter());
    // ウィンドウ設定
    setSize(250, 250);
    setVisible(true);
  }
  // マウスがクリックされたら
  public void mouseClicked(MouseEvent e){
    msgno=0;
    repaint();
  }
  // マウスが画面に入ったら
  public void mouseEntered(MouseEvent e){
    msgno=1;
    repaint();
  }
  // マウスが画面から出たら
  public void mouseExited(MouseEvent e){
    msgno=2;
    repaint();
  }
  // マウスのボタンが押されたら
  public void mousePressed(MouseEvent e){
    msgno=3;
    repaint();
  }
  // マウスのボタンが離されたら
```

```
52    public void mouseReleased(MouseEvent e){
53      msgno=4;
54      repaint();
55    }
56    // 描画メソッド
57    public void paint(Graphics g){
58      g.setColor(Color.BLACK);
59      g.drawString(msgs[msgno], 15, 50);
60    }
61    // アプリ起動
62    public static void main(String[] args){
63      Assignment12_1 obj = new Assignment12_1();
64    }
65
66    // ウィンドウイベント処理
67    private class SimpleWindowAdapter extends WindowAdapter{
68      public void windowClosing(WindowEvent e){
69        System.exit(0);
70      }
71    }
72  }
73
```

```
////////////////////////////////////////////////////////////////////////
// 課題２　星の画像がマウスカーソルを追従するコードを作成しなさい。次…
////////////////////////////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

class Assignment12_2 extends Frame implements MouseMotionListener
{
  // 表示画像
  private Image img;
  // 画像表示位置
  private int mx,my;

  // コンストラクタ
  public Assignment12_2(){
    // タイトル設定
    super("課題２");
    // 画像読み込み
    Toolkit tk = getToolkit();
    img = tk.getImage("star0.gif");
    // 画像の初期位置設定
    mx=0;
    my=0;
    // リスナー登録
    addMouseMotionListener(this);
    addWindowListener(new SimpleWindowAdapter());
    // ウィンドウ設定
    setSize(400, 400);
    setVisible(true);
  }
  // マウスモーションリスナーメソッド
  public void mouseDragged(MouseEvent e){}
  public void mouseMoved(MouseEvent e){
    mx = e.getX();
    my = e.getY();
    repaint();
  }
  // 描画メソッド
  public void paint(Graphics g){
    g.drawImage(img, mx, my,this);
  }
  // アプリ起動
  public static void main(String[] args){
    Assignment12_2 obj = new Assignment12_2();
  }

  // ウィンドウイベント処理
  private class SimpleWindowAdapter extends WindowAdapter{
    public void windowClosing(WindowEvent e){
      System.exit(0);
    }
```

```
53    }
54 }
55
```

```
////////////////////////////////////////////////////////////////////////
// 課題３　色の異なる複数の星の画像がマウスカーソルを追従するコードを作…
////////////////////////////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

class Assignment12_3 extends Frame implements MouseMotionListener
{
  // 表示画像配列
  private Image[] img=new Image[3];
  // 画像表示位置配列
  private int[][] ary = new int[50][2];

  // コンストラクタ
  public Assignment12_3(){
    // タイトル設定
    super("課題３");
    // 画像読み込み
    String name;
    Toolkit tk = getToolkit();
    for(int i=0;i<img.length;i++){
      name="star"+i+".gif";
      img[i] = tk.getImage(name);
    }
    // 画像の初期位置設定
    for(int i=0;i<ary.length;i++)
      for(int j=0;j<ary[i].length;j++)
        ary[i][j]=0;
    // リスナー登録
    addMouseMotionListener(this);
    addWindowListener(new SimpleWindowAdapter());
    // ウィンドウ設定
    setSize(500, 400);
    setVisible(true);
  }
  // マウスモーションリスナーメソッド
  public void mouseDragged(MouseEvent e){}
  public void mouseMoved(MouseEvent e){
    // 配列の値を一つ後ろへ移動
    for(int i=0;i<ary.length-1;i++){
      ary[ary.length-1-i][0] = ary[ary.length-2-i][0];
      ary[ary.length-1-i][1] = ary[ary.length-2-i][1];
    }
    // 最初の配列要素に現座標を保存
    ary[0][0] = e.getX();
    ary[0][1] = e.getY();
    // 再描画
    repaint();
  }
  // 描画メソッド
```

```java
////////////////////////////////////////////////////////////////////////
// 課題３　色の異なる複数の星の画像がマウスカーソルを追従するコードを作…
////////////////////////////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

public class Assignment12_3d extends Frame implements MouseMotionListener
{
  // 現在のウィンドウサイズ
  private int width, height;
  // 表示画像配列
  private Image[] img=new Image[3];
  // 画像表示位置配列
  private int[][] ary = new int[50][2];
  // ウィンドウイメージバッファ
  private Image offImage;

  // コンストラクタ
  public Assignment12_3d(){
    // タイトル設定
    super("課題３ - ダブルバッファ");
    // 初期のウィンドウサイズ設定
    width = 500;
    height = 400;
    // 画像読み込み
    String name;
    Toolkit tk = getToolkit();
    for(int i=0;i<img.length;i++){
      name="star"+i+".gif";
      img[i] = tk.getImage(name);
    }
    // 画像の初期位置設定
    for(int i=0;i<ary.length;i++)
      for(int j=0;j<ary[i].length;j++)
        ary[i][j]=0;
    // リスナー登録
    addMouseMotionListener(this);
    addWindowListener(new SimpleWindowAdapter());
    addComponentListener(new SimpleComponentAdapter());
    // ウィンドウ設定
    setSize(width, height);
    setVisible(true);
    // ウィンドウイメージバッファの生成
    offImage = createImage(width, height);
  }
  // マウスモーションリスナーメソッド
  public void mouseDragged(MouseEvent e){}
  public void mouseMoved(MouseEvent e){
    // 配列の値を一つ後ろへ移動
    for(int i=0;i<ary.length-1;i++){
```

```
52        ary[ary.length-1-i][0] = ary[ary.length-2-i][0];
53        ary[ary.length-1-i][1] = ary[ary.length-2-i][1];
54      }
55      // 最初の配列要素に現座標を保存
56      ary[0][0] = e.getX();
57      ary[0][1] = e.getY();
58      // 再描画
59      repaint();
60    }
61    // 描画メソッド１
62    // 再描画の要求：update()を実行、 内部でpaint()を実行
63    // デフォルトのupdate()：ウィンドウ背景をクリアし、paint()を実行
64    // 背景クリアがチラつきの原因のためオーバーライドで削除
65    public void update(Graphics g){
66      paint(g);
67    }
68    // 描画メソッド２
69    public void paint(Graphics g){
70      // ウィンドウイメージバッファに描画
71      Graphics gv = offImage.getGraphics();
72      gv.clearRect(0, 0, width, height);
73      for(int i=0;i<ary.length;i++)
74        gv.drawImage(img[i%3],ary[ary.length-1-i][0],ary[ary.length-1-i][1],this);
75      // ウィンドウイメージを描画
76      g.drawImage(offImage, 0, 0, width, height, this);
77    }
78    // アプリ起動
79    public static void main(String[] args){
80      Assignment12_3d obj = new Assignment12_3d();
81    }
82
83    // ウィンドウイベント処理
84    private class SimpleWindowAdapter extends WindowAdapter{
85      public void windowClosing(WindowEvent e){
86        System.exit(0);
87      }
88    }
89    // コンポネントイベント処理（ウィンドのサイズ変更など）
90    private class SimpleComponentAdapter extends ComponentAdapter{
91      public void componentResized(ComponentEvent e){
92        Component c = e.getComponent();
93        // 変更後のウィンドウサイズの取得
94        width = c.getWidth();
95        height = c.getHeight();
96        // ウィンドウイメージバッファの再生成
97        offImage = createImage(width, height);
98      }
99    }
100 }
101
```

```
52    public void paint(Graphics g){
53      for(int i=0;i<ary.length;i++)
54        g.drawImage(img[i%3],ary[ary.length-1-i][0],ary[ary.length-1-i][1],this);
55    }
56    // アプリ起動
57    public static void main(String[] args){
58      Assignment12_3 obj = new Assignment12_3();
59    }
60
61    // ウィンドウイベント処理
62    private class SimpleWindowAdapter extends WindowAdapter{
63      public void windowClosing(WindowEvent e){
64        System.exit(0);
65      }
66    }
67  }
68
```

```java
////////////////////////////////////////////////////////////////
// 課題４　マウスをクリックした場所にクロスヘア（照準）を表示す…
////////////////////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

class Assignment12_4 extends Frame
{
  // 照準を描く座標
  int mx, my;

  // コンストラクタ
  public Assignment12_4(){
    // タイトル設定
    super("課題４");
    // リスナー登録
    addMouseListener(new SimpleMouseAdapter());
    addWindowListener(new SimpleWindowAdapter());
    // ウィンドウ設定
    setSize(500, 400);
    setVisible(true);
  }
  // 描画メソッド
  public void paint(Graphics g){
    // クロスヘアの描画
    g.drawOval(mx-15,my-15,30, 30);
    g.drawOval(mx-25,my-25,50, 50);
    g.drawLine(mx,my-30,mx,my+30);
    g.drawLine(mx-30,my,mx+30,my);
  }
  // アプリ起動
  public static void main(String[] args){
    Assignment12_4 obj = new Assignment12_4();
  }

  // マウスイベント処理
  private class SimpleMouseAdapter extends MouseAdapter{
    public void mouseClicked(MouseEvent e){
      // マウスがクリックされたら座標を保存
      mx=e.getX();
      my=e.getY();
      repaint();
    }
  }
  // ウィンドウイベント処理
  private class SimpleWindowAdapter extends WindowAdapter{
    public void windowClosing(WindowEvent e){
      System.exit(0);
    }
  }
}
```

```java
////////////////////////////////////////////////////////////
// 課題５　2つのボタンを配置し、ボタンが押されたらそのボタン…
////////////////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

class Assignment12_5 extends Frame implements ActionListener
{
  // ボタン
  Button bt1, bt2;

  // コンストラクタ
  public Assignment12_5(){
    // タイトル設定
    super("課題５");
    // ボタンの生成
    bt1 = new Button("●");
    bt2 = new Button("○");
    // ボタンの識別子を登録
    bt1.setActionCommand("Button1");
    bt2.setActionCommand("Button2");
    // ボタンにイベントハンドラを登録
    bt1.addActionListener(this);
    bt2.addActionListener(this);
    // 複数のGUI部品を配置するにはPanelクラスを利用
    // パネルを生成しボタンを配置
    Panel p = new Panel();
    p.add(bt1);
    p.add(bt2);
    // パネルをウィンドウに追加
    add(p);
    // ウィンドウにリスナー登録
    addWindowListener(new SimpleWindowAdapter());
    // ウィンドウ設定
    setSize(500, 400);
    setVisible(true);
  }
  // アクションリスナーメソッド
  public void actionPerformed(ActionEvent ae){
    // 押されたボタンの種類を判断してラベルを変更
    if(ae.getActionCommand().equals("Button1")){
      bt1.setLabel("●");
      bt2.setLabel("○");
    }else{
      bt1.setLabel("○");
      bt2.setLabel("●");
    }
  }
  // アプリ起動
  public static void main(String[] args){
```

```
52      Assignment12_5 obj = new Assignment12_5();
53    }
54
55    // ウィンドウイベント処理
56    private class SimpleWindowAdapter extends WindowAdapter{
57      public void windowClosing(WindowEvent e){
58        System.exit(0);
59      }
60    }
61  }
62
```

```java
//////////////////////////////////////////////////
// 課題６　血液型が選択できるチョイス（プルダウンメ…
//////////////////////////////////////////////////

import java.awt.*;
import java.awt.event.*;

class Assignment12_6 extends Frame implements ItemListener
{
  // 血液型プルダウンメニュー
  private Choice mychoice;
  // 選択項目の表示ラベル
  private Label message;

  // コンストラクタ
  public Assignment12_6(){
    // タイトル設定
    super("課題６");
    // プルダウンメニューとラベルの生成
    mychoice = new Choice();
    message = new Label("【未選択】");
    // ※Labelは初期文字を設定
    // プルダウンメニューの選択肢を登録
    mychoice.add("Ａ型");
    mychoice.add("Ｂ型");
    mychoice.add("ＡＢ型");
    mychoice.add("Ｏ型");
    // パネルの生成と各部品の貼り付け
    Panel p = new Panel();
    p.add(mychoice);
    p.add(message);
    // パネルをウィンドウへ貼り付
    add(p);
    // ブルダウンメニューにリスナーを登録
    mychoice.addItemListener(this);
    // ウィンドウにリスナー登録
    addWindowListener(new SimpleWindowAdapter());
    // ウィンドウ設定
    setSize(500, 400);
    setVisible(true);
  }
  // アイテムリスナーメソッド
  public void itemStateChanged(ItemEvent e){
    // 押されたらその血液型を設定
    String str = mychoice.getSelectedItem();
    message.setText(str);
  }
  // アプリ起動
  public static void main(String[] args){
    Assignment12_6 obj = new Assignment12_6();
```

```
51    }
52
53    // ウィンドウイベント処理
54    private class SimpleWindowAdapter extends WindowAdapter{
55      public void windowClosing(WindowEvent e){
56        System.exit(0);
57      }
58    }
59  }
60
```