オブジェクト指向プログラミング 令和５年度 後期中間試験

(2023.12.18 重村 哲至) IE5 ＿＿＿番 氏名＿＿＿＿＿＿ 模範解答

以下では，付録に掲載した Svanna アプリについて答えなさい．

1 用語

語群から最適な言葉を記号で選びなさい．(3 点 ×10 問=30 点)

SavannaApp クラスと SavannaView クラスの関連は **(1)** である．SavannaView クラスは JPanel クラスを **(2)** し，JPanel クラスの **(3)** である `paintComponet()` を **(4)** している．

Animal クラスの **(5)** である `image` の **(6)** は private である．`image` の値を参照・変更するための **(7)** である `getImage()` と `setImage()` が用意されている．

このアプリは MVC アーキテクチャ (MVC モデル) を採用しており，Animal クラスが **(8)**，SavannaApp クラスが **(9)**，SavannaView クラスが **(10)** の役割を担っている．

語群:

(あ) アクセスメソッド，(い) オーバーライド，(う) コントローラ，(え) モデル，(お) ビュー，(か) 可視性，(き) 継承，(く) 合成集約，(け) 操作 (メソッド)，(こ) 属性 (プロパティ)

(1)	(く)	(2)	(き)	(3)	(け)	(4)	(い)	(5)	(こ)
(6)	(か)	(7)	(あ)	(8)	(え)	(9)	(う)	(10)	(お)

2 型に関する問題

1. SavannaApp クラスの`###(4)###`に許される型を全て答えなさい．(7 点)

   Animal と Elephant

2. SavannaApp クラスの`###(5)###`に許される型を全て答えなさい．(7 点)

   Animal と Lion

3. SavannaApp クラスの`###(1)###`に書くことのできる型は，`Animal`，`Elephant`，`Lion` の型の中でどれが許されるか，複数の型が許されるか許されないかなどの理由を付けて答えなさい．(7 点)

   Animal 型だけが許される．
   animal には Elephant と Lion 両方のインスタンスが渡されるので，animal の型は Elephant と Lion に共通のスーパークラスである Animal 型が適切である．

オブジェクト指向プログラミング 令和５年度 後期中間試験

(2023.12.18 重村 哲至) IE5 _____番 氏名_____ 模範解答

3 プログラムの穴埋め

1. SavannaApp クラスの###(2)###と###(3)###に適切な 2 行の記述を答えなさい．(7 点)

```
animal.setX(x);
animal.setY(y);
```

2. クラス図を参照し Animal クラスの###(6)###に適切な記述を答えなさい．(7 点)

```
private BufferedImage
```

3. クラス図を参照し Animal クラスの###(7)###と###(8)###に適切な共通の記述を答えなさい．(7 点)

```
private int
```

4. クラス図を参照し Animal クラスの###(9)###部分の getImage() メソッドを完成し，以下にメソッド全体を書きなさい．(7 点)

```
public BufferedImage getImage() {
  return image;
}
```

5. クラス図を参照し Elephant クラスと Lion クラスの###(10)###に適切な共通の記述を答えなさい．(7 点)

```
Animal
```

6. クラス図を参照し Elephant クラスと Lion クラスの###(11)###に適切な共通の記述を答えなさい．(7 点)

```
super
```

7. SavannaView クラスのコメントに###(12)###と記載のある行はもっと簡潔に書くことができます．以下に記述を答えなさい．（ヒント：オブジェクトは自身の振る舞い方を知っている．）(7 点)

```
a.draw(g);
```

付録 Savanna アプリ

画面の様子

Savanna アプリの動作画面です．ボタンで動物を追加できます．
`savanna.png` 上に `elphant.png` や `lion.png` を重ねて描いています．

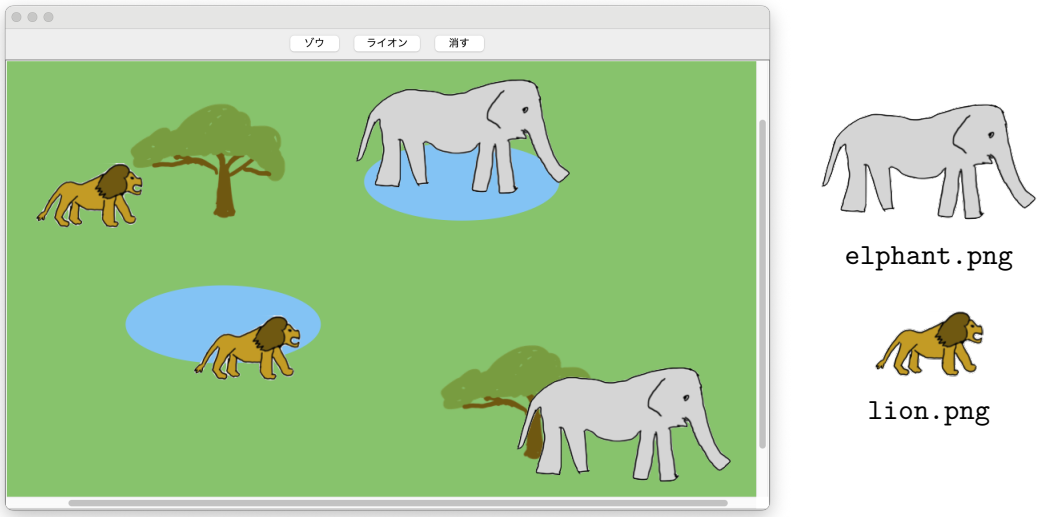

図 1: SavannaApp の表示例

クラス図

`Animal[*]` は `Animal` 型の配列を表しています．

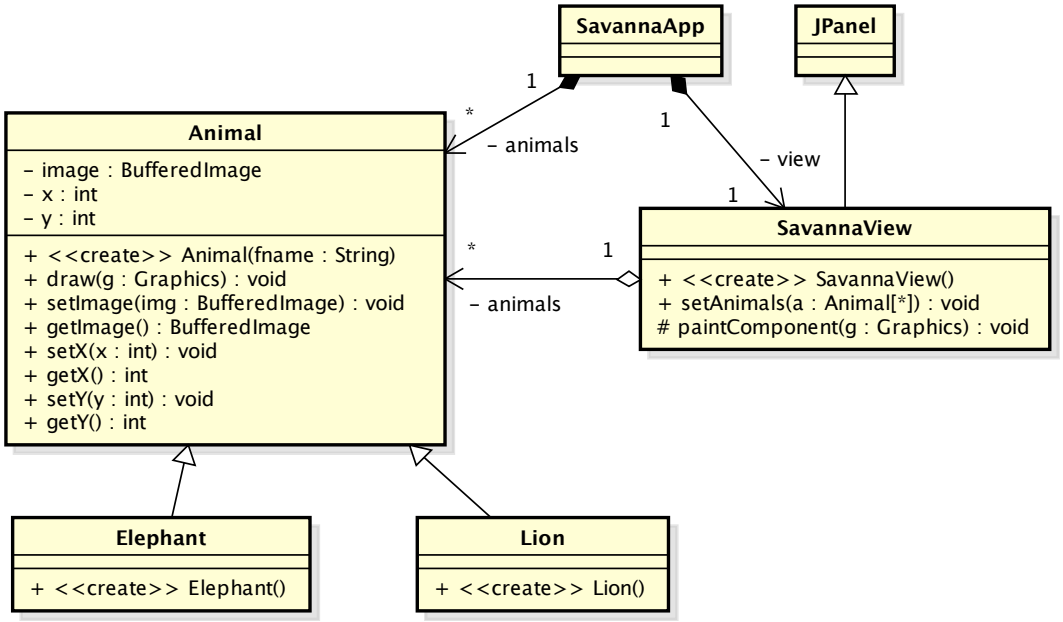

図 2: SavannaApp のクラス図

ソースプログラム

リスト 1: SavannaApp.java

```
... import 省略 ...

public class SavannaApp {
  private JFrame frame;
  private Animal[] animals;  // モデル(動物の配列)
  private int aniCnt = 0;
  final private Random rnd = new Random();
  private SavannaView view; // ビュー

  // 動物を追加する
  private void addAnimal(Animal ani) {
    animals[aniCnt] = ani;
    aniCnt = aniCnt + 1;
  }

  // 動物の配列を初期化する
  private void clearAnimals() {
    animals = new Animal[100]; // サバンナにいる動物（最大100匹）
    aniCnt = 0;                // 動物が何匹いるか
    view.setAnimals(animals);  // 最初は配列要素が全てnullになっている
  }

  // 動物の表示位置を乱数で決める
  private void locateAnimal(###(1)### animal) {
    BufferedImage img = animal.getImage();
    // 背景からはみ出さない範囲で，動物の位置を乱数で決める
    int x = rnd.nextInt(view.getWidth()-img.getWidth());
    int y = rnd.nextInt(view.getHeight()-img.getHeight());
    ###(2)###                  // 動物オブジェクトにX座標をセットする
    ###(3)###                  // 動物オブジェクトにY座標をセットする
  }

  public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
      public void run() {
        try {
          SavannaApp window = new SavannaApp();
          window.frame.setVisible(true);
        } catch (Exception e) {
          e.printStackTrace();
        }
      }
    });
  }

  public SavannaApp() {
    initialize();
    clearAnimals();  // 動物の配列を初期化する
  }
```

```
  private void initialize() {
    frame = new JFrame();
    frame.setBounds(100, 100, 450, 300);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel panel = new JPanel();
    frame.getContentPane().add(panel, BorderLayout.NORTH);

    JButton btnElephant = new JButton("ゾウ");
    btnElephant.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        ###(4)### elp = new Elephant();
        locateAnimal(elp);
        addAnimal(elp);
        view.repaint();
      }
    });
    panel.add(btnElephant);

    JButton btnLion = new JButton("ライオン");
    btnLion.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        ###(5)### lion = new Lion();
        locateAnimal(lion);
        addAnimal(lion);
        view.repaint();
      }
    });
    panel.add(btnLion);

    JButton btnClear = new JButton("消す");
    btnClear.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        clearAnimals();
        view.repaint();
      }
    });
    panel.add(btnClear);

    JScrollPane scrollPane = new JScrollPane();
    frame.getContentPane().add(scrollPane, BorderLayout.CENTER);

    view = new SavannaView();
    scrollPane.setViewportView(view);
  }
}
```

リスト 2: Animal.java

```
... import 省略 ...

// 動物を表現するクラス
public class Animal {
  ###(6)### image;                // 動物のイラスト
  ###(7)### x;                    // 動物の表示位置(x座標)
  ###(8)### y;                    // 動物の表示位置(y座標)
  // コンストラクタ（引数はイラストファイルのファイル名）
  public Animal(String fname) {
    BufferedImage img = null;
    try {
      img = ImageIO.read(new File(fname));
    } catch (IOException e) {
      e.printStackTrace();
    } finally {
      setImage(img);
    }
  }
  // 動物を表示する
  public void draw(Graphics g) {
    g.drawImage(image, x, y, null);
  }
  // 以下はアクセスメソッド
  public void setImage(BufferedImage image) {
    this.image = image;
  }
  ###(9)### getImage() {
  #########
  }
  public void setX(int x) {
    this.x = x;
  }
  public int getX() {
    return x;
  }
  public void setY(int y) {
    this.y = y;
  }
  public int getY() {
    return y;
  }
 }
```

リスト 3: Elephant.java

```
// ゾウを表すクラス
public class Elephant extends ###(10)### {
  // コンストラクタ
  public Elephant() {
    ###(11)###("elephant.png");  // elephant.pngはゾウのイラスト
  }
}
```

リスト 4: Lion.java

```
// ライオンを表すクラス
public class Lion extends ###(10)### {
  // コンストラクタ
  public Lion() {
    ###(11)###("lion.png");  // lion.pngはライオンのイラスト
  }
}
```

リスト 5: SavannaView.java

```
... import 省略 ...

// サバンナの様子を表示するJPanelクラス
@SuppressWarnings("serial")
public class SavannaView extends JPanel {
  private BufferedImage image;   // サバンナの背景イラスト
  private Animal[] animals;      // サバンナの動物たち

  // コンストラクタ
  public SavannaView() {
    try {
      image = ImageIO.read(new File("sabanna.png"));
    } catch (IOException e) {
      e.printStackTrace();
    } finally {
      if (image!=null) {
        setPreferredSize(new Dimension(image.getWidth(),image.getHeight()));
      } else {
        setPreferredSize(null);
      }
    }
  }

  // 動物の配列を登録する
  public void setAnimals(Animal[] animals) {
    this.animals = animals;
  }

  // サバンナの様子を描く
  @Override
  protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    g.drawImage(image, 0, 0, null);           // 背景イラストを表示
    for (int i=0; animals[i]!=null; i++) {    // 背景に重ねて
      Animal a = animals[i];                  //   全ての動物を描く
      g.drawImage(a.getImage(), a.getX(), a.getY(), null); // ###(12)###
    }
  }
}
```