オブジェクト指向プログラミング H24 年度 後期中間試験 (2012.12.03 重村 哲至)

IE5 ____ 番 氏名 模範解答

1. 次のようなクラスを定義しなさい.

Zahyo クラスは，点 (x,y) を表現するデータ構造を定義します．変数 (x,y) が座標を，変数 (r) が原点 (0,0) からの距離を表現します．コンストラクタは座標 (x,y) を引数とし，変数 (x,y,r) を初期化します．メソッド (setX(),setY()) は変数 (x,y) を変更たあと変数 (r) を再計算します．メソッド (getR()) は変数 (r) の値を読み取ります．private なメソッド (calR()) は変数 (x,y) の値から変数 (r) を計算します.

以上の説明と以下のクラスを表現する図から下の Java プログラムを完成しなさい.

(平方根の計算には `Math.sqrt(d)` を使用すること)

Zahyo	
- r	: double
- x	: double
- y	: double
- calR()	: void
+ Zahyo(double, double)	: コンストラクタ
+ setX(double)	: void
+ setY(double)	: void
+ getR()	: double

```
public class Zahyo {
  // (1) 以下に変数(x,y,r)を宣言しなさい．(10点)

  private double x;
  private double y;
  private double r;

  // (2) メソッド calR() を宣言しなさい．(10点)

  private void calR() {
    r = Math.sqrt(x*x + y*y);
  }
```

// (3) コンストラクタを宣言しなさい. (10 点)

```
public Zahyo(double a, double b) {
  x = a;
  y = b;
  calR();
}
```

// (4) メソッド setX(), setY() を宣言しなさい. (10 点)

```
public void setX(double a) {
  x = a;
  calR();
}

public void setY(double b) {
  y = b;
  calR();
}
```

// (5) メソッド getR() を宣言しなさい. (10 点)

```
public double getR() {
  return r;
}
```

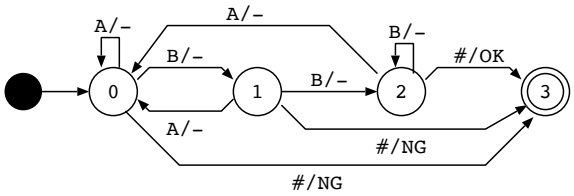
}

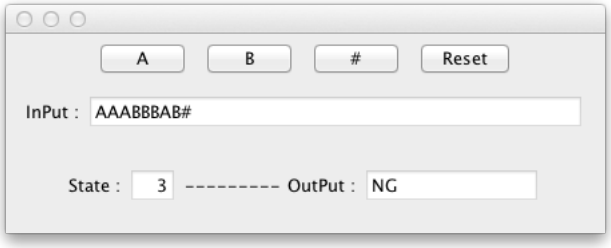
2. 次の説明を読み，Fsm クラスを定義しなさい．

Fsm クラスは'A' と'B' を繰返し'#' で終わる入力列を受け取り，「'#' の直前に'B' が２つ以上連続していたか」を判定し"OK","NG" のどちらかの文字列を JTextField に表示します．Fsm クラスには下のような有限状態機械が組込まれています．

Fsm クラスは次の GUI を持ったアプリケーションプログラム (FsmGui) から利用します．このアプリケーションは，ボタンからの入力に従いステートマシンが動作する様子を観察するものです．

'A','B','#' ボタンを押す毎に Input フィールドにこれまでの入力文字列を表示します．State フィールドには Fsm の現在の状態を表示します．Output フィールドには Fsm の出力を表示します．Reset ボタンを押すと Fsm の状態と表示が最初の状態に戻ります．





FsmGui クラスのソースコードは別紙のとおりです．FsmGui クラスから利用できるような Fsm クラスを次の図と各メンバの意味を参考に完成しなさい．なお，予定外の状態や，予定外の入力は無いものとして解答すれば十分です．

Fsm	
- state	: int
- field	: JTextField
+ Fsm(JTextField)	: コンストラクタ
+ input(char)	: boolean
- stat0(char)	: void
- stat1(char)	: void
- stat2(char)	: void
+ reset()	: void
+ getState()	: int

各メンバの意味は次のとおりです．

state: Fsm の現在の状態 (状態 0...3) を記憶する変数です．

field: Fsm の出力を表示する JTextField インスタンスを記憶する変数です．

Fsm(JTextField): コンストラクタです．出力を表示するための JTextField インスタンスを引数にします．

input(char): 'A', 'B', '#' ボタンが押された時に呼び出されます．このメソッドの引数は押されたボタンの種類を表します．Fsm の状態が既に終了状態 (状態 3) になっていて，それ以上の入力を受け付けない場合は false を返します．

stat0..2(char): input から現在の状態に応じて呼び出されます．入力を調べて次の状態を決めます．'#' が入力された場合は，field に “OK” か “NG” を表示します．

reset(): Reset ボタンが押された時に呼び出されます．

getState(): FsmGui が State の表示のために Fsm の状態を読み取るために呼出します．

```
import javax.swing.JTextField;
public class Fsm {
  private int state;
  private JTextField field;
  // (1) コンストラクタを完成しなさい. (10 点)
  public Fsm(JTextField f) {

    state = 0;     // ホントは必要ないけど...
    field = f;

  }

  // (2) input を完成しなさい. (10 点)
  public boolean input(char c) {
    boolean r = true;
    switch (state) {

    case 0:
      stat0(c);
      break;
    case 1:
      stat1(c);
      break;
    case 2:
      stat2(c);
      break;

    default:
      r = false;
      break;
    }
    return r;
  }
```

```
// (3) stat0 を完成しなさい. (10 点)
private void stat0(char c) {


  if (c=='A') {
  } else if (c=='B') {
    state = 1;
  } else {
    state = 3;
    field.setText("NG");
  }


}

// (4) stat1 を完成しなさい. (10 点)
private void stat1(char c) {


  if (c=='A') {
    state = 0;
  } else if (c=='B') {
    state = 2;
  } else {
    state = 3;
    field.setText("NG");
  }


}
```

```
  // (5) stat2 を完成しなさい. (10 点)
  private void stat2(char c) {


    if (c=='A') {
      state = 0;
    } else if (c=='B') {
    } else {
      state = 3;
      field.setText("OK");
    }


  }

  public void reset() {
    state = 0;
    field.setText("");
  }

  public int getState() {
    return state;
  }
}
```

[別紙]

```
public class FsmGui {
  private JFrame frame;
  private JTextField inputField;
  private JTextField outputField;
  private Fsm fsm;
  private JTextField stateField;
  ... 途中省略 ...

  // コンストラクタ
  public FsmGui() {
    initialize();
    fsm = new Fsm(outputField);
  }

  /**
   * Initialize the contents of the frame.
   */
  private void initialize() {
    frame = new JFrame();
    frame.setBounds(100, 100, 449, 171);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

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

    JButton btnA = new JButton("A");
    btnA.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent arg0) {
        if (fsm.input('A'))
          inputField.setText(inputField.getText()+"A");
        stateField.setText(Integer.toString(fsm.getState()));
      }
    });
    panel.add(btnA);

    JButton btnB = new JButton("B");
    btnB.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent arg0) {
        if (fsm.input('B'))
          inputField.setText(inputField.getText()+"B");
        stateField.setText(Integer.toString(fsm.getState()));
      }
    });
    panel.add(btnB);

    JButton btnEof = new JButton("#");
    btnEof.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent arg0) {
        if (fsm.input('#'))
          inputField.setText(inputField.getText()+"#");
        stateField.setText(Integer.toString(fsm.getState()));
      }
    });
    panel.add(btnEof);

    JButton btnReset = new JButton("Reset");
    btnReset.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent arg0) {
        fsm.reset();
        inputField.setText("");
        stateField.setText(Integer.toString(fsm.getState()));
      }
    });
    panel.add(btnReset);
```