Vejledende løsninger til F00 eksamenssættet.

Opgave 1

Spørgsmål 1.1

```
+--+
|xo|
|o*|
+--+
```

Spørgsmål 1.2

```
private static
void myShape(int n)
   {
      for(int i=1; i<=n; i=i+1)
      {
         // spaces before
         for(int j=0; j<n-i; j=j+1)
            System.out.print(" ");

         switch(i)
         {
             case 1:
                System.out.print("*"); break;
             case 2:
                System.out.print("***"); break;
             default:
                System.out.print("**");
                for(int j=0; j<1+2*(i-3); j=j+1)
                   System.out.print(" ");
                System.out.print("**");
                break;
         } // end switch(i)
         System.out.println();
      } // end for i
   }
```

Opgave 2

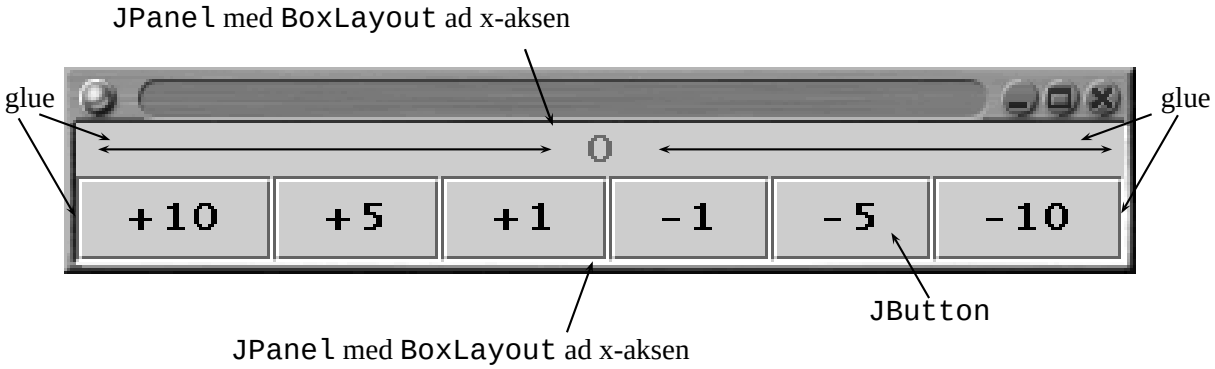
Spørgsmål 2.1

```
private static
String running(int[] a)
   {  // assumption: a.length>=3
      String res = "- -";

      for(int i=2; i<a.length; i=i+1)
         res = res + " " + Integer.toString((a[i-2]+a[i-1]+a[i])/3);

      return res;
   }
```

Opgave 3

Spørgsmål 3.1

Layoutet kan opnås med følgende komponenter og layout managere — læg mærke til, at der er sat lim ind på begge sider af knapperne for at de altid er centrerede, uanset hvor bredt vinduet bliver.



Spørgsmål 3.2

Konstruktoren kunne se således ud:

```
public
Counter()
   {
      label = new JLabel("0");
      counterValue=0;

      Container pane = this.getContentPane();
      JPanel labelPanel = new JPanel();
      labelPanel.setLayout(new BoxLayout(labelPanel,BoxLayout.X_AXIS));

      labelPanel.add(Box.createHorizontalGlue());
      labelPanel.add(label);
      labelPanel.add(Box.createHorizontalGlue());

      pane.add(labelPanel, BorderLayout.NORTH);

      JPanel buttonPanel = new JPanel();
      buttonPanel.setLayout(new BoxLayout(buttonPanel,BoxLayout.X_AXIS));
      buttonPanel.add(Box.createHorizontalGlue());

      int[] values = {10,5,1,-1,-5,-10};
      buttons= new JButton[values.length];

      for(int i=0; i<values.length; i=i+1)
      {
         buttons[i] = new JButton(intToString(values[i]));
         buttons[i].addActionListener(new MyListener(values[i]));
         buttonPanel.add(buttons[i]);
      }

      buttonPanel.add(Box.createHorizontalGlue());

      pane.add(buttonPanel,BorderLayout.SOUTH);
   }
```

eller således, hvor man bruger et `BoxLayout` i y-aksen i stedet for `BorderLayout`:

```
public
Counter2()
   {
      label = new JLabel("0");
      counterValue=0;

      Container pane = this.getContentPane();

      pane.setLayout(new BoxLayout(pane,BoxLayout.Y_AXIS));
      pane.add(label);

      JPanel buttonPanel = new JPanel();
      buttonPanel.setLayout(new BoxLayout(buttonPanel,BoxLayout.X_AXIS));

      int[] values = {10,5,1,-1,-5,-10};
      buttons= new JButton[values.length];

      for(int i=0; i<values.length; i=i+1)
      {
         buttons[i] = new JButton(intToString(values[i]));
         buttons[i].addActionListener(new MyListener(values[i]));
         buttonPanel.add(buttons[i]);
      }

      pane.add(buttonPanel);
   }
```

Spørgsmål 3.3

Et forslag til Listener-klassen:

```
private
class MyListener implements ActionListener
{
   private
   int value;

   public
   MyListener (int value)
      {
         this.value=value;
      }

   public
   void actionPerformed (ActionEvent e)
      {
         counterValue = counterValue + value;
         label.setText(Integer.toString(counterValue));
      }
} // MyListener
```

Opgave 4

Spørgsmål 4.1

Et DC-9 fly kunne laves således:

```
public class DC_9 implements Plane
{
   private
   String id;

   public
   DC_9(String id)
      {
         this.id = id;
      }

   public
   String id () { return id; }

   public
   int capacity() { return 100; }
} // DC_9
```

Spørgsmål 4.2

`available` og `bookSeats` metoderne kunne se ud som følger:

```
public
int available() { return plane.capacity() - passengers; }

public
void bookSeats(int seats) throws OverBooked
   {
      if (passengers + seats > plane.capacity())
         throw new OverBooked("Only room for " + plane.capacity()
                              + " passengers.");
      else
         passengers = passengers + seats;
   }
```

Spørgsmål 4.3

For at finde alle flyvninger med mellemlanding mellem A og B bliver man nødt til, at lave et gennemløb af alle flyvningerne for at finde mulige flyvninger fra A til en by C. Dernæst skal man undersøge om man kan flyve fra C til B — dette klares også — ved hjælp af et gennemløb af alle flyvningerne.

```
private static
int twoLegs(MyFlight[] fs, String from, String to)
   {
      int res=0;

      for(int i=0; i<fs.length; i=i+1)
      {
         if (from.equals(fs[i].from())
             && !to.equals(fs[i].to())
             && fs[i].available()>0)
         {
            for(int j=0; j<fs.length; j=j+1)
            {
               if (to.equals(fs[j].to())
                   && fs[i].to().equals(fs[j].from())
                   && fs[j].available()>0)
                  res = res +1;
            }
         }
      }
      return res;
   }
```