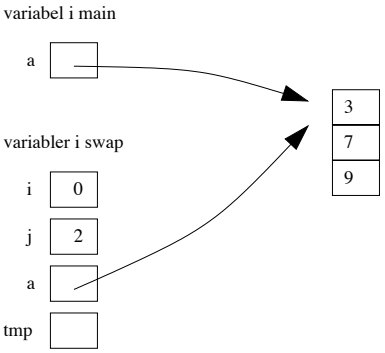
Tentamen i Objektorienterad programmering E

Förslag till lösningar

Måndagen 8 mars 2010, 8.30 – 12.30.

1. Variabeln a i `main` initieras till att peka på ett fält med värdena 3, 7 och 9 som i nedanstående figur.

   Då `swap` anropas kopieras värdet av argumenten till parametrarna och när koden i `swap` ska börja köra är situationen som nedan:



   Koden i `swap` byter värdena i a[0] och a[2]; när vi återkommer till `main` syns denna förändring och utskriften blir

```
9 7 3
```

2.
```
public int aliveCells() {
  int res = 0;
  for (int i=0; i < isAlive.length; i++)
    for (int j=0; j < isAlive[0].length; j++)
      if (isAlive[i][j]) res++;
  return res;
}
```

3.
```
public class Uppgift3 {

    public static double[] average3(double[] a) {
        double[] res = new double[a.length];
        for (int i=1; i<a.length-1; i++)
            res[i] = (a[i-1] + a[i] + a[i+1])/3;
        res[0] = a[0];
        res[res.length-1] = a[res.length-1];
        return res;
    }
```

```
    public static void main(String[] args) {
        double[] a = new double[args.length];
        for (int i=0; i<args.length; i++)
            a[i] = Double.parseDouble(args[i]);
        double[] avg = average3(a);
        for (int i=0; i<avg.length; i++)
            System.out.print(avg[i] + " ");
        System.out.println();
    }
}
```

4. Vi måste kunna lagra flera produkter och antal valda i ett objekt av klassen `ShoppingCart`. För detta kan vi antingen använda listor eller fält. För enkelhets skull väljer vi här fält och begränsar inköpslistans storlek i konstrueraren. Vidare måste ett element i listan bestå av både produkt och antal. Vi kan antingen definiera en ny klass vars objekt består av en produkt och ett antal eller använda två fält i klassen `ShoppingCart`. Vi väljer här det första alternativet.

   I nedanstående lösning definieras hjälpklassen `Entry` inne i klassen `ShoppingCart`. Denna klass kan också definieras separat, i en egen fil.

```
import java.io.*;

public class ShoppingCart {

    private static class Entry {
        private Product product;
        private int quantity;

        public Entry (Product product, int quantity) {
            this.product = product;
            this.quantity = quantity;
        }

        public Product getProduct() {return product;}
        public int getQuantity() {return quantity;}
    }

    private Entry[] entries;
    private int count;

    public ShoppingCart (int size) {
        entries = new Entry[size];
    }

    public void add(Product product, int quantity) {
        entries[count] = new Entry(product, quantity);
        count++;
    }

    public int totalAmount() {
```

```
        int res = 0;
        for (int i=0; i<count; i++)
            res += entries[i].getQuantity() * entries[i].getProduct().getPrice();
        return res;
    }

    public void print(PrintStream out) {
        for (int i=0; i<count; i++) {
            out.println( entries[i].getQuantity() + "  " +
                         entries[i].getProduct().getName() + "   " +
                         entries[i].getProduct().getPrice());
        }
    }
}
```

5. (a)
```
public class FIARModel {

    private State[][] board;

    public FIARModel(int width, int height) {
        board = new State[width][height];
        for (int x=0; x<board.length; x++)
            for (int y=0; y<board[0].length; y++)
                board[x][y] = State.FREE;
    }

    public void drop(int x) {
        int y = board[0].length-1;
        boolean canMove = tryMove(x,State.USER);
        if (canMove) {
            int x1 = (int)(Math.random()*board.length);
            boolean canMove1 = tryMove(x1,State.PROGRAM);
            // We could loop here until canMove1 becomes true; omitted
        }
    }

    private boolean tryMove(int x, State player) {
        int y=board[0].length-1;
        if (board[x][y] == State.FREE) {
            while (y>0 && board[x][y-1]==State.FREE) y--;
            board[x][y] = player;
            return true;
        } else
            return false;
    }

    public State getState(int x, int y) {
        return board[x][y];
    }
```

```
        public int getWidth() {return board.length;}

        public int getHeight() {return board[0].length;}

    }
```

(b)
```
public void mouseClicked (MouseEvent e) {
    model.drop(e.getX()/SIZE);
    repaint();
}
```

6.
```
public class Uppgift6 {

    public static interface Function {
        public double apply(double x);
    }

    public static double findZero(Function f, double a, double b, double eps) {
        double mid = (a+b)/2;
        if (b-a < 2*eps)
            return mid;
        else {
            double fmid = f.apply(mid);
            if (fmid < 0)
                return findZero(f,mid,b,eps);
            else
                return findZero(f,a,mid,eps);
        }
    }

    public static void main(String[] args) {
        Function f = new Function() {
                public double apply(double x) {
                    return x - Math.cos(x);
                }
            };
        double zero = findZero(f,0,1.5,1e-6);
        System.out.println("Nollstället är " + zero);
    }
}
```