

Tentamen i Objektorienterad programmering

Lördagen 12 mars 2011, 8.30 – 12.30.

Förslag till lösningar.

1. Utskriften blir

2
2
2

Detta följer av att `incr` har en parameter av den primitiva typen `int`. För en sådan parameter kopieras värdet av argumentet i anropet över till parametern; uppdateringar i subrutinen sker sedan i denna kopia, som försvinner då vi återgår till `main`. Därför kommer `k` och `v[1]` att vara oförändrade efter anropen av `incr`.

Subrutinen `incrFst`, däremot, har en parameter av referenstyp. Det medför att vid anrop kopieras pekaren i argumentet till parametern; själva fältet delas av subrutin och `main`. Därför kommer den ökning av `a[0]` som görs i subrutinen att synas också i `main` (där som `v[0]`) efter anropet.

OBS: Det är inte alls nödvändigt med så här lång diskussion för full poäng. En figur och ett par meningar räcker.

2. `public class Uppgift2 {`

```
    public static double length(double[] v) {
        double res = 0;
        for (int i=0; i<v.length; i++)
            res += v[i]*v[i];
        return Math.sqrt(res);
    }

    public static double[] normalize(double[] v) {
        double[] res = new double[v.length];
        double len = length(v);
        for (int i=0; i<res.length; i++)
            res[i] = v[i]/len;
        return res;
    }

    public static void main(String[] args) {
        double[] v = {1,2,3,4};
        double[] w = normalize(v);
        for (int i=0; i<w.length; i++)
            System.out.print(w[i] + " ");
        System.out.println();
    }
}
```

3. import java.util.*;

```
public class Horse {

    private String name;
    private List<Horse> children;

    public Horse(String name) {
        this.name = name;
        children = new LinkedList<Horse>();
    }

    public String getName() {return name;}
    public int getNrOfChildren() {return children.size();}
    public Horse getChild(int i){return children.get(i);}
    public void addChild(Horse h){children.add(h);}
}
```

Anm: Jag har här valt att samla barnen i en lista. Det går lika bra att använda ett fält och en heltalsvariabel (som anger antalet barn). I så fall bör konstrueraren ha en ytterligare parameter, dimensioneringen av fältet (dvs det maximala antalet barn hästen i fråga kommer att kunna ha).

4. import java.util.*;

```
public class Uppgift4 {

    public static void printWords(Scanner in) {
        while (in.hasNextLine()) {
            String w = in.findInLine("[a-zA-Z]+");
            while (w != null) {
                System.out.println(w);
                w = in.findInLine("[a-zA-Z]+");
            }
            in.nextLine();
        }
    }

    public static void printWordsSorted(Scanner in) {
        TreeSet<String> words = new TreeSet<String>();
        while (in.hasNextLine()) {
            String w = in.findInLine("[a-zA-Z]+");
            while (w != null) {
                words.add(w.toLowerCase());
                w = in.findInLine("[a-zA-Z]+");
            }
            in.nextLine();
        }
        for (String w : words)
            System.out.println(w);
    }
}
```

```

    public static void main(String[] args) {
        printWordsSorted(new Scanner(System.in));
    }
}

5. (a) public void move(int r0, int c0, int r, int c) {
    int mr = (r+r0)/2;
    int mc = (c+c0)/2;
    if (board[r0][c0]==State.OCCUPIED &&
        board[mr][mc]==State.OCCUPIED &&
        board[r][c]==State.EMPTY &&
        ((r==r0 && Math.abs(c-c0)==2) ||
         (c==c0 && Math.abs(r-r0)==2))) {
        board[r0][c0] = State.EMPTY;
        board[mr][mc] = State.EMPTY;
        board[r][c] = State.OCCUPIED;
    }
}

(b) public void mouseClicked(MouseEvent e) {
    if(hasSelection) {
        model.move(selRow, selCol, e.getY()/GRID_SIZE, e.getX()/GRID_SIZE);
        hasSelection = false;
    }
    else {
        selRow = e.getY()/GRID_SIZE;
        selCol = e.getX()/GRID_SIZE;
        hasSelection = model.getState(selRow, selCol)==State.OCCUPIED;
    }
    repaint();
}

6. public static void printDescendants(Horse h) {
    printIndent(h, 0);
}

private static void printIndent(Horse h, int level) {
    for (int i=0; i<level; i++) System.out.print("  ");
    System.out.println(h.getName());
    for(int i=0; i<h.getNrOfChildren(); i++)
        printIndent(h.getChild(i), level+1);
}

```