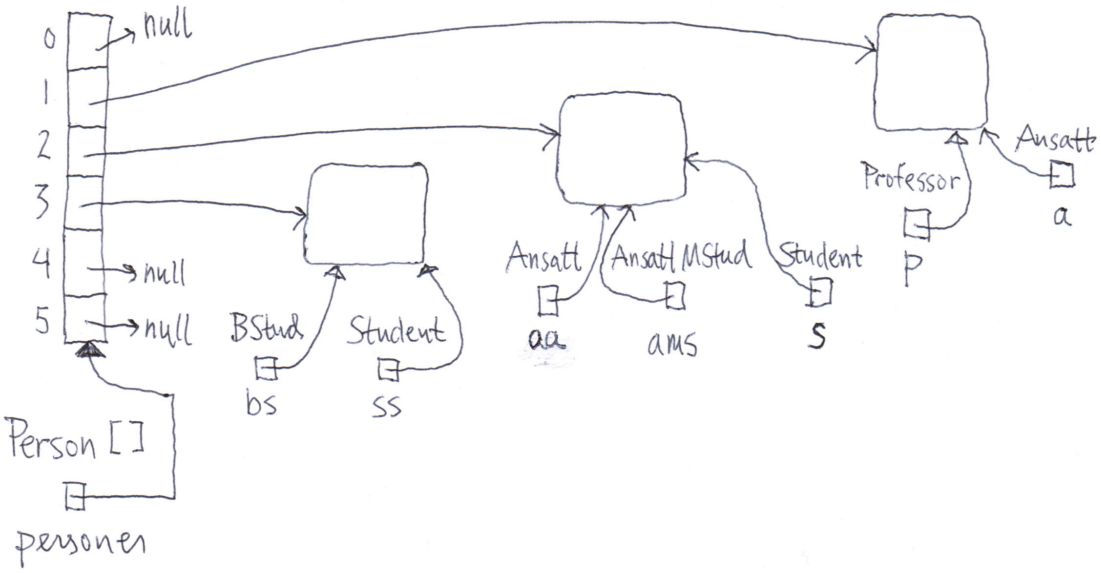
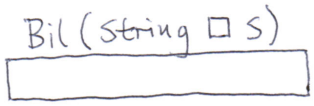
vedlegg 1


0
1
2
3
4
5
null
null
null
BStud
Student
bs
ss
Person [ ]
personer
Ansatt
AnsattMStud
Student
aa
amS
S
Professor
Ansatt
p
a

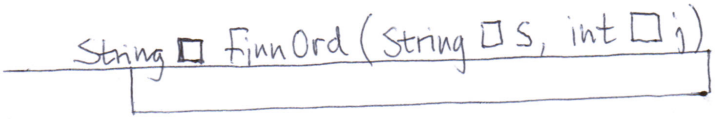
vedlegg 2

```
class Oppgave {
    public static void main(String[] args) {
        new BilListe(args);
    }
}

class BilListe {

    private Bil fBil = null;

    BilListe ( String [] inputstrenger ) {
        Bil b = null;
        fBil = new Bil("Førstebil?");
        for (String s: inputstrenger) {
            b = new Bil(s);
            b.neste = fBil;
            fBil = b;
        }

        /*
          OPPGAVE 2
        */

        while (fBil != null) {
            System.out.println( fBil + "␣" +
                                fBil.regNr + "␣" + fBil.bilNr + "␣" + Bil.nr);
            fBil = fBil.neste;
        }

    }

}

class Bil {
    static int nr = 0;

    Bil neste;
    String regNr;
    int bilNr;

    Bil(String s) {
        bilNr = nr++;
        regNr = s;
    }
}
```

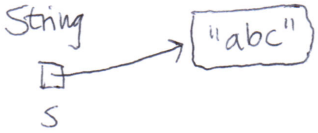
Konstruktøren i Bil:



En metode med to parametre og returverdi:



Et stringobjekt med verdien "abc":



Et objekt med en heltallsvariabel:

vedlegg 4

```
import javax.swing.*; import java.awt.*; import java.awt.event.*;

class Rute extends JPanel implements MouseListener {
    JLabel hull = null; // objektet vi bruker for å visualisere hull eller brikke

    // Opppgave 3a
    // deklarer andre variable for et ruteobjekt, f.eks. hvilken kolonne
    // dette obejektet hører til

    ImageIcon brikkeB = new ImageIcon("blank.png"); // hull
    ImageIcon brikkeR = new ImageIcon("raud.png");  // rød brikke
    ImageIcon brikkeG = new ImageIcon("gronn.png"); // grønn brikke

    Rute ( .... ) {

        // Oppgave 3a (skriv hele konstruktøren sammen med variablene ovenfor)
        // Konstruktøren for rute. For å visualisere ei tom rute
        // kan man legge dette objektet til ruta:
        //
        // hull = new JLabel(brikkeB);

    }
    // Oppgave 3b
    // andre metoder, bl.a. den som skal fange opp museklikk

    // Oppgave 3c
    // metode som får ruta til å skifte utseende, avhengig av om det er
    // den ene eller andre fargens tur. Det eneste du trenger å kode her er
    // testen i if-testen, og oppdatere datastrukturen om at neste gang er det den
    // andre fargens tur.

    void skiftFarge() {
        if ( /* <rød sin tur> */ )  hull.setIcon(brikkeR);
        else hull.setIcon(brikkeG);
    }
}

// Oppgave 3d
// annen datastruktur, f.eks. eget kolonneobjekt, men det er mulig å løse oppgaven
// uten annen struktur enn en dobbeltarray med ruteobjekter.

class FirePR {
    public static void main(String [] a) {
        JFrame ramme = new JFrame("Fire på rad");
        JPanel panel = new JPanel(); // GUI-brettet
        panel.setLayout(new GridLayout(6,7));

        // Oppgave 3e
        // Her skal du skrive kode som oppretter datastrukturen for spillet. Hullene
        // som spillet består av skal være objekter av klassen Rute. Siden Rute er
        // subklasse av JPanel, er det plass til 6 * 7 ruter i GUI-brettet (panel).
        // Et ruteobjekt r legges til med kallet panel.add(r).
        // Når strukturen er opprettet og alle rutene
        // lagt til panelet, legges dette inn i ramma med kallet:

        ramme.add(panel);
        ramme.pack();
        ramme.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ramme.setVisible(true);
    }
}
```