Lösningsförslag, kontrollskrivning 3 PTDC

2012–12–17

1.
```
public class ReservationList {
    private ArrayList<Reservation> reservations;
    private int currentNbr;

    public ReservationList() {
        reservations = new ArrayList<Reservation>();
        currentNbr = 0;
    }

    public Reservation find(int reservationNbr) {
        for (int i = 0; i < reservations.size(); i++) {
            if (reservations.get(i).getReservationNbr() == reservationNbr) {
                return reservations.get(i);
            }
        }
        return null;
    }

    public int insertReservation(int tickets, Performance perf) {
        currentNbr++;
        Reservation r = new Reservation(currentNbr, tickets, perf);
        reservations.add(r);
        return currentNbr;
    }

    public void delete(Reservation r) {
        reservations.remove(r);
    }
}
```

2.
```
public class ReservationManager {
    private PerformanceList pl;
    private ReservationList rl;

    public ReservationManager(PerformanceList pl, ReservationList rl) {
        this.pl = pl;
        this.rl = rl;
    }

    public void makeReservation(String time, int tickets) {
        Performance p = pl.findPerformance(time);
        if (p == null) {
            System.out.println("Ingen sådan föreställning");
        } else if (p.getTicketsLeft() < tickets) {
            System.out.println("För få biljetter kvar");
        } else {
            p.reserveTickets(tickets);
            int nbr = rl.insertReservation(tickets, p);
            System.out.println("Bokningsnummer: " + nbr);
        }
    }
```

```
    public void printTickets(int reservationNbr) {
        Reservation r = rl.find(reservationNbr);
        if (r == null) {
            System.out.println("Ingen sådan bokning");
        } else {
            int nbrTickets = r.getNbrTickets();
            for (int i = 0; i < nbrTickets; i++) {
                r.printOneTicket();
            }
            rl.delete(r);
        }
    }
}
```

3.
```
class CountSelection {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int nbrPersons = scan.nextInt();
        int n = scan.nextInt();
        Ring ring = new Ring();
        scan.nextLine();
        for (int i = 0; i < nbrPersons; i++) {
            ring.insert(new Person(scan.nextLine()));
        }
        ring.reset();
        while (ring.size() > 1) {
            ring.countTo(n);
            ring.removeCurrent();
        }
        System.out.println(ring.removeCurrent().getName() + " valdes ut");
    }
}
```

4.
```
public class Ring {
    private ArrayList<Person> ring;
    private int current;

    public Ring() {
        ring = new ArrayList<Person>();
    }

    public void insert(Person p) {
        ring.add(p);
    }

    public void reset() {
        current = 0;
    }

    public void countTo(int n) {
        for (int i = 0; i < n - 1; i++) {
            current = (current + 1) % ring.size();
        }
    }
```

```
    public Person removeCurrent() {
        Person p = ring.remove(current);
        if (current > ring.size() - 1) {
            current = 0;
        }
        return p;
    }

    public int size() {
        return ring.size();
    }
}
```