

Lösningsförslag, omkontrollskrivning PTDC

2013-01-08

```
1. public class Ticket {
    private int id;
    private String title;
    private boolean open;
    private Date date;
    private ArrayList<Item> items;

    public Ticket(int id, String title) {
        this.title = title;
        this.id = id;
        open = true;
        date = new Date();
        items = new ArrayList<Item>();
    }

    public void close() {
        open = false;
    }

    public boolean isOpen() {
        return open;
    }

    public int getId() {
        return id;
    }

    public void add(Item item) {
        items.add(item);
    }

    public ArrayList<Item> getItems() {
        return items;
    }

    public int compareTo(Ticket t) {
        if (isOpen() && !t.isOpen()) {
            return 1;
        }
        if (!isOpen() && t.isOpen()) {
            return -1;
        }
        return date.compareTo(t.date);
    }
}
```

```

    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append(title);
        sb.append(" (ID ").append(id).append(")");
        if (open) {
            sb.append(" - ÖPPEN - ");
        } else {
            sb.append(" - STÄNGD - ");
        }
        sb.append(date).append("\n");
        for (int i = 0; i < items.size(); i++) {
            sb.append(" ").append(items.get(i)).append("\n");
        }
        return sb.toString();
    }
}

```

```

2. public class Bugs {
    private ArrayList<Ticket> tickets;
    private int nextId;

    public Bugs() {
        tickets = new ArrayList<Ticket>();
        nextId = 1;
    }

    public int newTicket(String title) {
        tickets.add(new NormalTicket(nextId, title));
        nextId++;
        sortTickets(tickets);
        return nextId - 1;
    }

    public boolean updateTicket(int id, Item item) {
        for (int i = 0; i < tickets.size(); i++) {
            if (tickets.get(i).getId() == id) {
                tickets.get(i).add(item);
                return true;
            }
        }
        return false;
    }

    public void closeTicket(int id) {
        for (int i = 0; i < tickets.size(); i++) {
            if (tickets.get(i).getId() == id) {
                tickets.get(i).close();
                sortTickets(tickets);
                return;
            }
        }
    }

    public void printTickets(boolean opened, boolean closed) {
        for (int i = 0; i < tickets.size(); i++) {
            Ticket t = tickets.get(i);
            if (t.isOpen() && opened || !t.isOpen() && closed) {
                System.out.println(t);
            }
        }
    }
}

```

```

public ArrayList<Ticket> getParticipantsTickets(String owner) {
    ArrayList<Ticket> temp = new ArrayList<Ticket>();
    for (int i = 0; i < tickets.size(); i++) {
        ArrayList<Item> items = tickets.get(i).getItems();
        for (int k = 0; k < items.size(); k++) {
            if (items.get(k).getOwner().equals(owner)) {
                temp.add(tickets.get(i));
                break;
            }
        }
    }
    sortTickets(temp);
    return temp;
}

private void sortTickets(ArrayList<Ticket> myTickets) {
    for (int i = 0; i < myTickets.size() - 1; i++) {
        int max = i;
        for (int j = i + 1; j < myTickets.size(); j++) {
            if (myTickets.get(max).compareTo(myTickets.get(j)) < 0) {
                max = j;
            }
        }
        Ticket tmp = myTickets.get(i);
        myTickets.set(i, myTickets.get(max));
        myTickets.set(max, tmp);
    }
}
}

```

```

3. public class Test {
    public static void main(String[] args) {
        Bugs bugs = new Bugs(); // 1

        int id1 = bugs.newTicket("Java-programmet kompilerar inte");
        bugs.updateTicket(id1, new Item("Nisse",
            "Jag kan inte kompilera int a = \"Hello\";"));

        int id2 = bugs.newTicket("Programmet kraschar");
        bugs.updateTicket(id2, new Item("Nisse",
            "Jag får IOException när jag vill öppna en fil."));
        bugs.updateTicket(id2, new Item("Olle", "Jag tittar på problemet"));

        bugs.closeTicket(id2);

        System.out.println("Stängda felrapporter:");
        bugs.printTickets(false, true);

        System.out.println("Felrapporter där Alfred bidragit:");
        ArrayList<Ticket> tickets = bugs.getParticipantsTickets("Alfred");
        for (Ticket t : tickets) {
            System.out.println(t);
        }
    }
}

```