Ohjelmistotuotanto, kurssikoe 29.4.2014

Kirjoita jokaiseen palauttamaasi konseptiin kurssin nimi, kokeen päivämäärä, nimesi ja opiskelijanumerosi.

Vastaa tehtäviin ytimekkäästi. Minkään tehtävä vastauksen pituuden ei tule ylittää kolmea sivua, kaksikin sivua tiukkaa asiaa riittää hyvin.

Vastaa jokainen tehtävä omalle konseptilleen.

1. (a) (2p) Kerro mikä on User story? Mitä ovat hyvän User storyn tunnusmerkit?
   (b) (2p) Kerro mikä on Product backlog. Kurssimateriaalissa todettiin että hyvä Product backlog on DEEP. Mitä tällä tarkoitetaan? Perustele myös mihin olettamuksiin perustuu DEEPin backlogin hyvyys.
   (c) (2p) Mitä tarkoitetaan Sprint backlogilla? Miten se liittyy Product backlogiin?

2. (a) (1p) Kurssimateriaalissa puhuttiin ohjelmiston ulkoisesta ja sisäisestä laadusta. Mitä ohjelmiston *ulkoisella laadulla* tarkoitetaan? Sisäiseen laatuun palataan kysymyksessä 3a.
   (b) (1p) Mitä tarkoitetaan ohjelmiston validoinnilla ja verifioimisella?
   (c) (3p) Miten ja mitä menetelmiä käyttäen ohjelmiston ulkoisen laadun varmistus ketterässä ohjelmistotuotannossa tyypillisesti tapahtuu?

3. (a) (1p) Minkälaista on sisäiseltä laadultaan hyvä ohjelmakoodi?
   (b) (1p) Mitä sisäisen laadun kannalta ongelmallisia asioita tehtäväpaperin lopun koodissa on?
   (c) (3p) Selitä miten voisit parantaa tehtäväpaperin lopun koodia soveltaen suunnittelumalleja. Vastaukseen kannattanee liittää mukaan luonnosmaista koodia. Älä kuitenkaan kirjoita "javaboilerplatea" (luokka+näkyvyysmäärittelyt ym) kokonaisuudessaan

Tehtävään 3 liittyvä ohjelmakoodi

```
public class Bibrary {
    private List<BibItem> items;
    private BibFileReader reader;

    public Bibrary() {
        reader = new BibFileReader("bibitems.db");
        items = reader.readFromFile();
    }

    public void addItem(BibItem item){
        items.add(item);
    }

    public List<BibItem> publishedYear(int year){
        List<BibItem> found = new ArrayList<>();

        for (BibItem bibItem : items) {
            if ( bibItem.year==year) {
                found.add(bibItem);
            }
        }

        return found;
    }

    public List<BibItem> publishedBy(String publisher){
        List<BibItem> found = new ArrayList<>();

        for (BibItem bibItem : items) {
            if ( bibItem.publisher.contains(publisher)) {
                found.add(bibItem);
            }
        }

        return found;
    }

    public List<BibItem> writtenBy(String author){
        List<BibItem> found = new ArrayList<>();

        for (BibItem bibItem : items) {
            if ( bibItem.author1.equals(bibItem)
                    ||bibItem.author2.equals(bibItem)
                    ||bibItem.author3.equals(bibItem)
                    ||bibItem.author4.equals(bibItem)) {
                found.add(bibItem);
            }
        }

        return found;
    }

    // searching methods based on other fields
```

```
    public void generateBibtex(String file){
        String bibtex = "";
        for (BibItem bibItem : items) {
            bibtex += bibItem.toBibTex()+ "\n";
        }

        // save the generated string to file...
    }

    public void close(){
        reader.saveToFile();
    }
}



public class BibItem {
    String type;
    String author1;
    String author2;
    String author3;
    String author4;
    String name;
    String publisher;
    int year;
    String pages;

    // other fields...

    public BibItem(String t, String a1, String a2, String a3, String a4, String n, int y, String p) {
        this.type = t;
        this.author1 = a1;
        this.author2 = a2;
        this.author3 = a3;
        this.author4 = a4;
        this.name = n;
        this.publisher = p;
        this.year = y;
    }

    public BibItem(String type, String a1, String name, int year) {
        this(type, a1, "", "", "", name,  year, "");
    }

    public BibItem(String type, String a1, String a2, String name, int year) {
        this(type, a1, a2, "", "", name, year, "");
    }

    public BibItem(String type, String a1, String a2, String a3, String name, int year) {
        this(type, a1, a2, a3, "", name, year, "");
    }

    public BibItem(String type, String a1, String name, int year, String p) {
        this(type, a1, "", "", "", name,  year, p);
    }

    public BibItem(String type, String a1, String a2, String name, int year, String p) {
        this(type, a1, a2, "", "", name, year, p);
```

```
    }

    public BibItem(String type, String a1, String a2, String a3, String name, int year, String p) {
        this(type, a1, a2, a3, "", name, year, p);
    }

    // constructors for other fields...

    public String toBibTex(){
        String bibitem = "@misc{\n";
        if ( type.equals("inproceedings")) {
            bibitem = "@inproceedings{\n";
        } else if ( type.equals("book")) {
            bibitem = "@book{\n";
        } else if ( type.equals("article")) {
            bibitem = "@article{\n";
        }

        bibitem += " name = {"+name+"}";
        bibitem += " year = {"+year+"}";
        bibitem += " publisher = {"+pages+"}";

        String authors = author1;
        if ( !author2.isEmpty() ){
            authors = "," + author2;
        }
        if ( !author3.isEmpty() ){
            authors = "," + author3;
        }
        if ( !author4.isEmpty() ){
            authors = "," + author4;
        }
        bibitem += " authors = {"+authors+"}";

        return bibitem;
    }
}



public class BibFileReader {
    private String fileName;

    public BibFileReader(String fileName) {
        this.fileName = fileName;
    }

    public List<BibItem> readFromFile(){
        List<BibItem> items = new ArrayList<>();
        // read items from file
        return items;
    }

    public void saveToFile(){
        // save items to file
    }
}
```