Skriftlig eksamen i Datalogi

Modul 1

Vinter 1998/99

Opgavesættet består af 5 opgaver, der ved bedømmelsen tillægges følgende vægte:

Opgave 1	16%
Opgave 2	12%
Opgave 3	10%
Opgave 4	37%
Opgave 5	25%

Alle sædvanlige hjælpemidler er tilladt. I tilfælde af unøjagtigheder i opgaveteksterne forventes det, at deltagerne selv præciserer besvarelsernes forudsætninger.

Opgavesættet består af en forside og 5 paginerede sider. Kontroller at din kopi er fuldstændig.

Opgave 1: Træer (16%)

Lad der være givet nøglerne `M A C I N T O S H`.

Spørgsmål 1.1 Nøglerne indsættes i nævnte rækkefølge i et tomt binært søgetræ uden brug af balancering. Tegn det resulterende træ.

Spørgsmål 1.2 Nøglerne indsættes i nævnte rækkefølge i en tom binær hob. Tegn resultatet i form af et binært træ.

Spørgsmål 1.3 Nøglerne indsættes i nævnte rækkefølge i et tomt 2-3-4-træ. Tegn det resulterende træ.

Spørgsmål 1.4 Nøglerne indsættes i nævnte rækkefølge i et tomt rød-sort-træ. Tegn det resulterende træ.

Opgave 2: Grafer (12%)

Nedenfor er vist en matrix-repræsentation af en vægtet ikke-orienteret graf. Kanternes vægte er angivet med positive talværdier.

	A	**B**	**C**	**D**	**E**	**F**	**G**
A	0	2	7	1	0	0	0
B	2	0	0	3	10	0	0
C	7	0	0	8	0	9	0
D	1	3	8	0	5	10	11
E	0	10	0	5	0	0	6
F	0	0	9	10	0	0	12
G	0	0	0	11	6	12	0

Spørgsmål 2.1 Tegn grafen.

Spørgsmål 2.2 Tegn det mindste udspændende træ og angiv summen af dets vægte.

Spørgsmål 2.3 Angiv løsningen af alle-korteste-veje-problemet for grafen i form af en matrix.

Opgave 3: Rekursion (10%)

Nedenfor er vist en Java-klasse kaldet `Mystery`.

```
class Mystery {
    int a[], k, n;

    Mystery(int k, int n) {
        this.k = k; this.n = n;
        a = new int[k];
        up(1, k);
    }

    void print() {
        for (int i = 0; i < k; i++)
           System.out.print(a[i] + " ");
        System.out.println();
    }

    void up(int b, int c) {
        if (c == 0)
            print();
        else
            for (int i = b; i <= n; i++) {
                a[k - c] = i;
                down(i+1, c-1);
            }
    }

    void down(int b, int c) {
        if (c == 0)
            print();
        else
            for (int i = n; i >= b; i--) {
                a[k - c] = i;
                up(i+1, c-1);
            }
    }
}
```

Spørgsmål 3.1 Hvad udskrives ved udførelse af sætningen `new Mystery(3, 4)`?

Opgave 4: Objektorienteret programmering (37%)

Betragt følgende skitse fra et Javaprogram

```
abstract class Frugt {}

abstract class Trae {}

class Knop extends Trae {}

class Gren extends Trae {
   Frugt f;
   Trae t;
}

class Deling extends Trae {
   Trae v;
   Trae h;
}

class Citron extends Frugt {}

class Appelsin extends Frugt {}
```

Det kan i denne opgave antages, at felter i objekter ikke har eller kan få værdien `null`.

Spørgsmål 4.1 Tilføj konstruktorer til klasserne `Gren` og `Deling`, der sætter objekternes felter.

Spørgsmål 4.2 Antag at der findes følgende erklæring og initialisering af en variabel `t`:

```
Trae t = new Deling(new Gren(new Citron(),new Knop()),
                    new Gren(new Appelsin(),new Knop()));
```

Tilføj `toString`-metoder til klasserne, så metoden `toString` for træ `t` returnerer

```
Deling(Gren(Citron,Knop),Gren(Appelsin,Knop))
```

Spørgsmål 4.3 Hvis vi har en metode `p` defineret som

```
public static void p(String s){System.out.println(s);}
```

så kan træet `t` ikke udskrives ved et kald `p(t)`, da `p` forventer en streng, og `toString`-metoden kun kaldes, når et objektet bruges i et regneudtryk, som om det var en tekststreng. I stedet kan man bruge kaldene `p(""+t)` eller `p(t.toString())`. Dette er lidt klodset, så foreslå en anden definition af metode `p`, så den kan kaldes med et `Trae`-objekt og blive udskrevet.

Spørgsmål 4.4 Tilføj en ekstra variant af metode `p`, så boolske værdier kan udskrives.

Spørgsmål 4.5 Antag at der tilføjes en abstrakt metode

```
public abstract void saetFrugt(Frugt f);
```

til klassen `Trae`. Tilføj versioner af metoden for de nedarvede klasser, så frugten indsættes længst til højre i træet. Den knop, der sidder længst til højre i træet, skal erstattes med en gren, der har frugten og en knop. Hvis træet blot er en knop, skal metoden ikke gøre noget.

Spørgsmål 4.6 Skitser hvorledes træet `t` (fra spørgsmål 4.2) ser ud efter kaldet

```
t.saetFrugt(new Citron());
```

Spørgsmål 4.7 Et træ behøver ikke at have frugter. Vi ønsker at kunne undersøge, om et træ har frugter og definerer en metode

```
public boolean harFrugter(Trae t)
```

der undersøger, om argumentet er et træ, der har frugter. Skriv kroppen til denne metode. Metoden tænkes skrevet i en klasse uden for træ-hierarkiet.

Spørgsmål 4.8 Metoden `harFrugter` kan også skrives som metoder i de nedarvede klasser fra `Trae`. Vi erklærer da en abstrakt metode

```
public abstract boolean harFrugter();
```

i klassen `Trae`. Tilføj versioner af metoden for de nedarvede klasser af klassen `Trae`.

Opgave 5: OOA-modellering (25%)

I forbindelse med oprettelse af den nye IT-højskole skal flere universiteter med hver sin kursus-, projekt- og eksamensstruktur arbejde sammen om at udbyde undervisningstilbud, registrere tilmeldinger og eksamensresultater samt håndtere meritoverførsler. Forestil dig, at du indgår i en projektgruppe, som skal udarbejde et forslag til et nyt fælles edb-system, som skal understøtte dette samarbejde.

Alle universiteterne udbyder såvel kurser som projektvejledning. Et undervisningstilbud kan bestå af en vilkårlig kombination af kurser og projekter, inklusiv ingen kurser eller ingen projekter. Universiteterne har den fælles regel, at en studerende kan til- og framelde sig en eksamen et vilkårligt antal gange, men de kan kun gå op 3 gange (3-gangs reglen). Vi ser her bort fra dispensationsmuligheden. Universiteterne er desuden blevet enige om fælles tilmeldingsfrister, hhv. 1. april og 1. november for hhv. sommer- og vintereksamen, og tilsvarende er de blevet enige om, at 1. maj og 1. december er sidste frister for frameldinger, hvorefter tilmeldinger er bindende. Vi ser altså også bort fra eventuelle lægeerklæringer i sidste øjeblik.

Ledelsen af IT-højskolen ønsker at benytte den nyeste teknologi. Visionen er, at de respektive studienævn udbyder deres undervisningstilbud på en fælles hjemmeside. Herfra skal de studerende melde sig til de forskellige undervisningstilbud og til eksamen. Systemet skal også benyttes af de respektive eksamenskontorer til at registrere og holde styr på de studerendes eksamensresultater, og af studienævnene som grundlag for behandling af ansøgninger om meritoverførsler.

Din projektgruppe skal foretage en analyse ved hjælp af OOA. Sammen med repræsentanter fra de deltagende universiteter er I kommet frem til følgende foreløbige systemdefinition udtrykt i BATOFF:

Betingelser: Edb-systemet skal benyttes af studienævnssekretærer, studerende og ansatte på eksamenskontorerne, og systemet skal udvikles i tæt samarbejde med repræsentanter for hver af disse grupper. Systemet skal udvikles og tages i brug, selv om de måtte have modstridende krav.

Anvendelsesområde: Administration af tilmeldinger til undervisningstilbud og til eksaminer samt af eksamensresultater. Overvågning af at studerende overholder 3-gangs reglen.

Teknologi: De eksisterende PC'er på universiteterne skal benyttes, og de, der ikke måtte have adgang til www, skal have dette etableret.

Objektsystem: Student, lærer, censor, undervisningstilbud, eksamen.

Funktionalitet: Støtte til undervisnings- og eksamensadministration. På sigt overvejes det at føre kontrol med lærernes produktivitet i form af afholdt undervisning og gennemførelsesprocenter (antal tilmeldte/antal beståede).

Filosofi: Et værktøj til administration og overvågning samt et kommunikationsmedium.

Spørgsmål 5.1 Konstruer og begrund et klassediagram for objektsystemet.

Spørgsmål 5.2 Konstruer og begrund et tilstandsdiagram for student.

NB. Hvis du foretager afgrænsninger eller gør dig yderligere forudsætninger end de beskrevne, skal du beskrive disse i opgavebesvarelsen.