

Skriftlig eksamen i Programmering og Udvidet Programmering

KVL, 4. januar 1999

Alle hjælpemidler tilladt, dog ikke datamat.

Opgavesættet består af tre opgaver der alle ønskes løst.

For studerende i Programmering gives eksamenskarakteren på grundlag af besvarelsen af disse tre opgaver.

For studerende i Udvidet Programmering gives eksamenskarakteren på grundlag af besvarelsen af disse tre opgaver samt besvarelsen af kursets rapportopgave.

I besvarelsen må du gerne benytte klasser og metoder fra bogen og noterne uden at skrive dem af, men du skal give en præcis henvisning med afsnitsnummer eller sidetal.

Opgave 1

Nedenstående java-klasse Dato repræsenterer en dato ved årstallet og dagens nummer (1–365) i året. Der ses bort fra skudår, så februar har altid 28 dage og året har altid 365 dage.

```
class Dato { // Antag: ingen skudår
    private int aar, dagnr;

    private static int[] sidstedag =
        { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 };

    Dato(int aar, int maaned, int dag)
    { this.aar = aar; this.dagnr = dagnr(maaned, dag); }

    Dato(Dato d)
    { this.aar = d.aar; this.dagnr = d.dagnr; }

    public String toString()
    { return "år " + aar + " dag nummer " + dagnr; }

    private int dagnr(int maaned, int dag)
    { return sidstedag[maaned-1] + dag; }
}
```

Læg mærke til at den første konstruktor tager år, måned og dag som argument og konverterer til den interne repræsentation ved hjælp af tabellen `sidstedag`. Ideen er at `sidstedag[m]` er nummeret på den m 'te måneds sidste dag, hvor m er et af tallene fra 1 til 12.

Den anden konstruktor laver en frisk kopi af et Dato-objekt; dette skal bruges i opgave 2.

Opgave 1.1

Her er et lille program der benytter Dato-klassen:

```
public class Opgave990103 {  
  
    public static void main(String[] args) {  
        Dato d1 = new Dato(1999, 1, 4);  
        Dato d2 = new Dato(1999, 5, 1);  
        Dato d3 = new Dato(1999, 12, 31);  
        System.out.println("d1 = " + d1);  
        System.out.println("d2 = " + d2);  
        System.out.println("d3 = " + d3);  
    } }  
}
```

Hvad udskriver programmet når det køres?

Opgave 1.2

Udvid Dato-klassen med en metode:

```
private int maaned() { ... }
```

som returnerer månedsnummeret (1–12) for det givne dato-objekt. For eksemplet vist ovenfor skal `d1.maaned()` give 1, og `d2.maaned()` skal give 5.

Opgave 1.3

Udvid Dato-klassen med en metode:

```
public boolean foer_eller_lig(Dato dato2)
```

som undersøger om den givne dag kommer før eller på samme tidspunkt som `dato2`. For eksemplet vist ovenfor skal `d1.foer_eller_lig(d2)` give `true`, `d2.foer_eller_lig(d1)` skal give `false` og `d2.foer_eller_lig(d2)` skal give `true`.

Opgave 1.4

Udvid Dato-klassen med en metode:

```
public void flyt(int antaldage) { ... }
```

som ændrer det givne dato-objekt til den dato som ligger `antaldage` senere. For eksempel skal kaldet `d1.flyt(10)` ændre `d1` til den 14. januar 1999 (altså `aar = 1999` og `dagnr = 14`).

Man kan antage at `antaldage` ikke er negativt, så den resulterende dato vil ligge på eller efter den givne dato.

Der skal tages hensyn til eventuelt årsskifte. Eksempel: Kaldet `d3.flyt(10)` skal ændre `d3` til den 10. januar 2000 (altså `aar = 2000` og `dagnr = 10`).

Der skal ikke tages hensyn til skudår.

Opgave 2

Denne opgave handler om projekter. Et projekt har et navn, en startdato, og en slutdato. Dato-klassen er den der blev defineret i opgave 1, inklusive de metoder der skulle laves i delopgaverne. Du må gerne bruge alle de omtalte metoder fra Dato-klassen, også selvom du ikke har besvaret noget af opgave 1.

```
class Projekt {
    String navn;
    Dato start, slut;

    Projekt(String navn, Dato start, Dato slut) {
        this.navn = navn;
        this.start = new Dato(start);
        this.slut = new Dato(slut);
    }

    public String toString()
    { return navn + " fra " + start + " til " + slut; }
}
```

Opgave 2.1

Hvad udskriver dette program når det køres:

```
Dato d1 = new Dato(1999, 1, 4);
Projekt p1 = new Projekt("Renovering", new Dato(2000, 1, 1), new Dato(2002, 12, 31));
Projekt p2 = new Projekt("Rette opgaver", d1, new Dato(1999, 2, 1));
System.out.println("p1 = " + p1);
System.out.println("p2 = " + p2);
```

Opgave 2.2

Udvid Projekt-klassen med følgende to metoder:

```
public void forlaeng(int antaldage) { ... }

public void udskyd(int antaldage) { ... }
```

Kaldet `forlaeng(antaldage)` skal forlænge projektet med det givne antal dage (ved at flytte slutdatoen). Kaldet `udskyd(antaldage)` skal udskyde projektet med det givne antal dage (ved at flytte både start- og slutdatoen).

Opgave 2.3

Definér en klasse `ProjektMedMilepaele` som en subklasse (subclass) af `Projekt`. Den skal udvide `Projekt` med en tabel med milepæle, dvs. datoer på hvilke et eller andet delmål skal være nået. (Her er vi kun interesserede i datoerne, ikke i delmålene).

Vis subklasseerklæringen for `ProjektMedMilepaele`, inklusiv en passende konstruktor.

Opgave 2.4

Definér en metode `public boolean tjekmilepaele()` i subklassen `ProjektMedMilepaele`. Metoden skal returnere `true` hvis milepælene kommer i før-eller-lig-orden (som defineret i opgave 1.3) og alle milepæle ligger efter (eller på) projektets startdato og før (eller på) dets slutdato. Metoden skal returnere `false` ellers.

Opgave 3

Denne opgave handler om håndtering af DNA-sekvenser som strenge i Java. Den forudsætter intet kendskab til DNA-sekvensers biologiske egenskaber.

Opgave 3.1

Nedenstående program optæller hyppighed af nukleotiderne i en DNA-streng. Hjælpeметoden `kode` returnerer en talkode mellem 0 og 3 for hver nukleotid (A, C, G, T) for at lette indekseringen i tabellen `frekv1`. Husk at i Java er alle elementer af en heltalstabel (såsom `frekv1`) nul fra begyndelsen.

```
public class Opgave990101 {

    public static void main(String[] args) {
        final int antalnucl = 4;
        String s = "ACGTCAGGATACAGCA";

        int[] frekv1 = new int[antalnucl];
        for (int i=0; i<s.length(); i++) {
            char nukl = s.charAt(i);
            frekv1[kode(nukl)]++;
        }
        System.out.println("Hyppighed af nukleotider:");
        for (int i=0; i<antalnucl; i++)
            System.out.print(frekv1[i] + " ");
        System.out.println();
    }

    static int kode(char nukl) {
        switch (nukl) {
            case 'A': return 0;
            case 'C': return 1;
            case 'G': return 2;
            case 'T': return 3;
            default: { System.out.println("Fejl: ikke en nukleotid"); return -1; }
        }
    }
}
```

Vis hvad programmet udskriver når det køres.

Opgave 3.2

Udvid metoden `main` fra opgave 3.1 med en programstump som optæller og udskriver hyppigheden af alle nukleotidpar i en DNA-streng. For DNA-strengen lagret i variabelen `s` ovenfor skal resultatet være

```
Hyppighed af nukleotidpar:
0 2 2 1
3 0 1 0
1 1 1 1
1 1 0 0
```

Første linie svarer til par der begynder med A, anden linie svarer til par der begynder med C, osv. Første kolonne svarer til par der ender på A, anden kolonne til par der ender på C, osv.

Første linie viser altså at parret AA forekommer 0 gange, AC forekommer 2 gange, AG forekommer 2 gange, og AT forekommer 1 gang.

Anden linie viser at parret CA forekommer 3 gange, CC forekommer 0 gange, CG forekommer 1 gang, og CT forekommer 0 gange.

Tilsvarende for tredje og fjerde linie.

Vink: der skal bruges en to-dimensional tabel.