

Forslag til løsninger til

Skriftlig eksamen i Programmering og Udvidet Programmering

KVL, 4. januar 1999

NB: Disse forslag til løsninger er ikke blevet tjekket på maskine. Løsningerne kan altså indeholde såvel syntaksfejl som andre fejl. Peter Sestoft 1999-12-13.

Opgave 1

Opgave 1.1

Programmet udskriver

```
d1 = år 1999 dag nummer 4
d2 = år 1999 dag nummer 121
d3 = år 1999 dag nummer 365
```

Opgave 1.2

Månednummeret kan findes ud fra feltet `dagnr` ved brug af tabellen `sidstedag`:

```
public int maaned() {
    int md = 0;
    while (sidstedag[md] < dagnr)
        md++;
    return md;
}
```

Når while-løkken standser gælder `sidstedag[md] >= dagnr`, og `md` er det mindste indeks for hvilket dette gælder, så `md` er nummeret (1–12) på måneden der indeholder `dagnr`.

Det er nok en fejl at der i opgaveteksten står at metoden skal være `private`. Det er mere nyttigt at lave den `public`.

Opgave 1.3

For at sammenligne datoerne skal man lave en leksikografisk sammenligning på felterne `aar` og `dagnr`:

```
public boolean foer_eller_lig(Dato dato2) {
    return aar < dato2.aar || (aar == dato2.aar && dagnr <= dato2.dagnr);
}
```

Opgave 1.4

```
public void flyt(int antaldage) {
    int res = dagnr + antaldage;
    aar = aar + res / 365;
    dagnr = res % 365;
}
```

Opgave 2

Opgave 2.1

Programmet udskriver:

```
p1 = Renovering fra år 2000 dag nummer 1 til år 2002 dag nummer 365
p2 = Rette opgaver fra år 1999 dag nummer 4 til år 1999 dag nummer 32
```

Opgave 2.2

```
public void forlaeng(int antaldage) {
    slut.flyt(antaldage);
}

public void udskyd(int antaldage) {
    start.flyt(antaldage);
    slut.flyt(antaldage);
}
```

Eftersom Dato-objekterne `start` og `slut` er oprettet som kopier af de givne Dato-objekter (i Projekt-objektets konstruktor), så har ændringen af Dato-objekterne `start` og `slut` ingen utilsigtede sideeffekter.

Opgave 2.3

```
class ProjektMedMilePaele extends Projekt {
    Dato[] milepaele;

    ProjektMedMilePaele(String navn, Dato start, Dato slut, Dato[] milepaele) {
        super(navn, start, slut);
        this.milepaele = milepaele;
    }
}
```

Opgave 2.4

Metoden `tjekmilepaele` skal returnere falsk hvis de er to milepæle der kommer i den gale rækkefølge (det ordner for-løkken) eller hvis der er mindste én milepæl og den første milepæl kommer før `start` eller den sidste kommer efter `slut` (det ordner if-sætningen):

```
public boolean tjekmilepaele() {
    for (int i=1; i<milepaele.length; i++)
        if (!milepaele[i-1].foer_eller_lig(milepaele[i]))
            return false;
    if (milepaele.length != 0
        && (!start.foer_eller_lig(milepaele[0])
            || !milepaele[milepaele.length-1].foer_eller_lig(slut)))
        return false;
    return true;
}
```

En alternativ, muligvis mere forståelig løsning:

```
public boolean tjekmilepaele() {
    boolean tjek = true;
    int i = 1;
    while (tjek && i<milepaele.length
        && milepaele[i-1].foer_eller_lig(milepaele[i]))
        i++;
    if (tjek && milepaele.length >= 1)
        tjek = start.foer_eller_lig(milepaele[0])
            && milepaele[milepaele.length-1].foer_eller_lig(slut);
    return tjek;
}
```

Opgave 3

Opgave 3.1

Programmet udskriver:

```
Hyppighed af nukleotider:  
6 4 4 2
```

Opgave 3.2

I slutningen af metode main tilføjes:

```
int[][] frekv2 = new int[antalnukl][antalnukl];  
for (int i=1; i < s.length; i++) {  
    char nukl1 = s.charAt(i-1);  
    char nukl2 = s.charAt(i);  
    frekv2[kode(nukl1)][kode(nukl2)]++;  
}  
System.out.println("Hyppighed af nukleotidpar:");  
for (int n1=0; n1 < antalnukl; n1++) {  
    for (int n2=0; n2 < antalnukl; n2++)  
        System.out.println(frekv2[n1][n2] + " ");  
    System.out.println();  
}
```