

# Skriftlig eksamen i Programmering

KVL, 10. januar 1998

*Alle hjælpemidler tilladt, dog ikke datamat.*

*Opgavesættet består af tre opgaver, der alle ønskes løst og vægtes lige ved bedømmelsen.*

*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

Betragt nedenstående Java-program:

```
public class Opgave980111 {

    public static void main(String[] args)
    {
        int[] tabel = { 3, 7, 6, 9, 8, 8, 4, 5, 5, 5, 2 };
        System.out.println(maxsekvens(tabel, tabel.length));
    }

    static int maxsekvens(int[] a, int antal)
    {
        int sek, maxsek = 0;
        if (antal > 0)
        {
            int tmp = a[0];
            sek = 1;
            for (int i=1; i<antal; i++)
                if (a[i] != tmp)
                {
                    tmp = a[i];
                    sek = 1;
                }
            else
            {
                sek++;
                if (sek > maxsek)
                    maxsek = sek;
            }
        }
        return maxsek;
    }
}
```

### Opgave 1.1

Hvilket tal udskrives på skærmen når programmet køres?

### Opgave 1.2

Modificér metoden `maxsekvens` så den i stedet returnerer det tal der har den længste udbrudte gentagelse i tabellen `a[0..(antal-1)]`. Eksempel: I programmet ovenfor er det tallet 5.

### Opgave 1.3

Modificér ovenstående program så det indlæser tal fra tekstfilen `"talfil.txt"` og kalder `maxsekvens` på de indlæste tal.

## Opgave 2

Nedenstående Java-metoder implementerer *boblesortering*, jævnfør opgave 1 i afsnit 7.4 af noterne om Søgning og Sortering:

```
private static void ombyt(int[] arr, int s, int t)
{
    int tmp = arr[s]; arr[s] = arr[t]; arr[t] = tmp;
}

public static void boblesort(int[] arr, int n)
    // sorter arr[0..n-1]
{
    for (int i = n-1; 0 < i; i--)
        for (int j = 0; j < i; j++)
            {
                if (arr[j] > arr[j+1])
                    ombyt(arr, j, j+1);
            }
}
```

### Opgave 2.1

Vis alle ombytninger der foretages når boblesortering køres på tabellen indeholdende følgende 6 heltal: 23, 25, 18, 60, 50, 44.

### Opgave 2.2

Hvor mange gange udføres den indre løkke krop { if (arr[j] > arr[j+1]) ... } i boblesort, når man sorterer en tabel med 6 tal?

Hvor mange gange udføres den indre løkke krop når man sorterer en tabel med  $n$  tal? Det er nok at angive størrelsesorden.

Hvad er bedste (og værste) køretid for sortering af  $n$  tal med ovenstående metode boblesort? Det er nok at angive størrelsesorden.

### Opgave 2.3

I kurset er gennemgået tre andre sorteringsmetoder. Hvilken af dem ligner boblesortering mest med hensyn til tidsforbrug?

## Opgave 3

Her erklæres en abstrakt klasse af beholdere samt to konkrete subclasser:

```
abstract class Beholder {
    abstract public double rumfang();
    double indhold;
}

class Kasse extends Beholder {
    double laengde, bredde, hoejde;

    Kasse(double side)
    { laengde = bredde = hoejde = side; }

    Kasse(double laengde, double bredde, double hoejde)
    { this.laengde = laengde; this.bredde = bredde; this.hoejde = hoejde; }

    public double rumfang()
    { return laengde * bredde * hoejde; }
}

class Toende extends Beholder {
    double radius, hoejde;

    Toende(double radius, double hoejde)
    { this.radius = radius; this.hoejde = hoejde; }

    public double rumfang()
    { return hoejde * 3.14 * radius * radius; }
}
```

### Opgave 3.1

Beskriv hvilke tre tal nedenstående program udskriver på skærmen når det køres:

```
public static void main(String[] args)
{
    Beholder k1 = new Kasse(2);
    Beholder k2 = new Kasse(1, 1.5, 2);
    Beholder t = new Toende(1, 2);
    System.out.println(k1.rumfang());
    System.out.println(k2.rumfang());
    System.out.println(t.rumfang());
}
```

### Opgave 3.2

Erklær en ny Java-klasse `Trug` som en subclasse af `Beholder`. Et trug er en liggende cylinder, der er halveret på langs. Et trug har altså en bredde  $b$  (lig med to gange cylinderens radius) og en længde  $\ell$  (lig med cylinderens højde). Rumfanget af et trug er halvdelen af cylinderens rumfang, altså  $\ell \cdot \pi \cdot \frac{(b/2)^2}{2}$  hvor  $\pi$  kan sættes til 3,14.

### Opgave 3.3

Definér en Java-metode `static double totalrumfang(Beholder[] b)` som returnerer det samlede rumfang af beholderne i tabellen `b`.

### Opgave 3.4

Definér en ny metode `public void fyldpaa(double maengde)` for beholdere, som forøger feltet `indhold` med `maengde`, dog sådan at `indhold` højst bliver lig med beholderens faktiske rumfang. (Resten af påfyldningen løber så ved siden af).

I hvilke(n) klasse(r) skal metoden `fyldpaa` defineres?