

**Written exam in  
Introductory Programming  
Solutions**  
IT-C, June 16, 2000  
English version

*The solutions have not been checked on machine and may therefore contain syntax and other errors.*

## Question 1

### Question 1.1

```
6
11
15
18
20
Gennemsnittet er: 4
```

### Question 1.2

The following Java program will output the six lines:

```
class Opgave1_2 {
    public static void main(String[] args) {
        for (int i=1; i<=6; i=i+1) {
            for (int j=1; j<i; j=j+1)
                System.out.print("-");
            System.out.println("\\");
        }
    }
}
```

The outer loop is executed once for each line. The inner loop outputs a hyphen  $i-1$  times for a given line  $i$ . The character “\” is output at the end of each line – note the double backslash.

### Question 1.3

The solution in question 1.2 is general and works for any number of lines by substituting 6 (in the outer loop) with the variable number of lines  $n$ .

```
static void figur(int n) {
    for (int i=1; i<=n; i=i+1) {
        for (int j=1; j<i; j=j+1)
            System.out.print("-");
        System.out.println("\\");
    }
}
```

## Question 1.4

We present three different implementations of the method `static void parallel(int n)` all based on the method `figur` from question 1.3.

```
static void parallel(int n) {
    for (int i=1; i<=n; i=i+1) {
        for (int j=1; j<i; j=j+1)
            if ((i - j) % 2 == 0)
                System.out.print("\\");
            else
                System.out.print("-");
            System.out.println("\\");
        }
    }

static void parallel_sol2(int n) {
    for (int i=1; i<=n; i=i+1) {
        for (int j=i; j>1; j=j-1)
            if (j % 2 == 0)
                System.out.print("-");
            else
                System.out.print("\\");
            System.out.println("\\");
        }
    }

static void parallel_sol3(int n) {
    for (int i=1; i<=n; i=i+1) {
        for (int j=i; j>0; j=j-1)
            if (j % 2 == 0)
                System.out.print("-");
            else
                System.out.print("\\");
            System.out.println();
        }
    }
}
```

## Question 2

### Question 2.1

```
static int antal_lige(int[] t) {
    int antal = 0;
    for (int i=0; i<t.length; i=i+1)
        if (t[i] % 2 == 0)
            antal = antal + 1;
    return antal;
}
```

The variable `antal` is initialized to 0 which is the value returned if the array `t` is empty.

```
static int antal_ulige(int[] t) {
    return t.length - antal_lige(t);
}
```

Because of `t.length = antal_ulige(t) + antal_lige(t)`, we can implement the method `antal_ulige` using the method `antal_lige`.

The method `antal_ulige` can also be implemented using a loop as in `antal_lige`:

```
static int antal_ulige(int[] t) {
    int antal = 0;
    for (int i=0; i<t.length; i=i+1)
        if (t[i] % 2 != 0)
            antal = antal + 1;
    return antal;
}
```

### Question 2.2

```
static int st_mlm(int[] t) {
    int st_mlm = 0;
    for (int i=1; i<t.length; i=i+1)
        if (t[i] - t[i-1] > st_mlm)
            st_mlm = t[i] - t[i-1];
    return st_mlm;
}
```

If `t` contains 0 or 1 element, then the body of the loop is not executed and the value 0 is returned. Because the array is sorted in ascending order, the number `t[i] - t[i-1]` is always positive or 0.

### Question 2.3

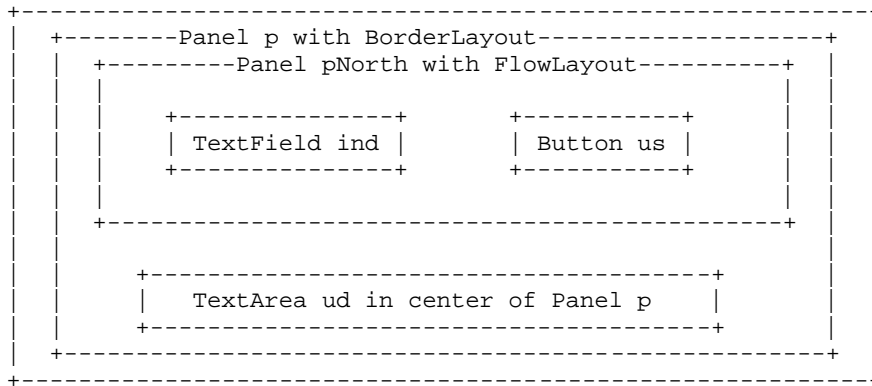
```
static int st_mlm_lige(int[] t) {
    int[] lige = new int[antal_lige(t)];
    int ligenr = 0;
    for (int i=0; i<t.length; i=i+1)
        if (t[i] % 2 == 0) {
            lige[ligenr] = t[i];
            ligenr = ligenr + 1;
        }
    return st_mlm(lige);
}
```

We filter out the odd numbers by creating a new array `lige` of size `antal_lige(t)` and then copy the even numbers from `t` to `lige`. Then we call the method `st_mlm` to compute the largest difference between two even neighbours.

## Question 3

### Question 3.1

The applet contains a panel `p` with `BorderLayout`. Another panel `pNorth` with `FlowLayout` containing the text field `ind` and the button `us` is added to `p` in north. A text area `ud` is added in center of panel `p`.



### Question 3.2

The text area `ud` contains the following two lines:

```
9 svarer til B.
12 svarer til ?.
```

### Question 3.3

The button `Nulstil` can be added as follows. At `/* A */` we initialize a field `nulstil` and declare a listener class as an inner class:

```
/* A */
Button nulstil = new Button("Nulstil");

class NulstilLytter implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        ind.setText("");
        ud.setText("");
    }
}
```

At `/* D */` we add a listener to button `Nulstil` and adds the button `Nulstil` at the south of the panel `p`.

```
/* D */
nulstil.addActionListener(new NulstilLytter());
p.add(nulstil, "South");
```

### Question 3.4

We declare the button and a listener class at `/* A */`.

```
/* A */
Button dk = new Button("DK");

class DKlytter implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        String karakterUS = ind.getText();
        String karakterDK = "?";
        if (karakterUS.equals("A"))
            karakterDK = "11";
        else if (karakterUS.equals("B"))
            karakterDK = "9";
        else if (karakterUS.equals("C"))
            karakterDK = "7";
        else if (karakterUS.equals("D"))
            karakterDK = "5";
        else if (karakterUS.equals("F"))
            karakterDK = "03";
        ud.append(karakterUS + " svarer til " + karakterDK + ".\n");
    }
}
```

At `/* B */`, we add the button `dk` to the panel `pNorth` after the button `us` has been added. We also add a listener to the button.

```
/* B */
dk.addActionListener(new DKlytter());
pNorth.add(dk);
```

## Question 4

### Question 4.1

The class Rum is declared below.

```
class Rum {
    String navn;
    int areal;

    Rum(String navn, int areal) {
        this.navn = navn;
        this.areal = areal;
    }

    public int getAreal() {
        return areal;
    }
}
```

### Question 4.2

The class Køkken is declared below as a subclass of Rum.

```
class Køkken extends Rum {
    Køkken(int areal) {
        super("Køkken", areal);
    }
}
```

### Question 4.3

The class NVærelses is declared below:

```
class NVærelses {
    Rum[] rum;
    Køkken køkken;
    int andelsværdi;

    NVærelses(Rum[] rum, Køkken køkken, int andelsværdi) {
        this.rum = rum;
        this.køkken = køkken;
        this.andelsværdi = andelsværdi;
    }

    public int getAreal() {
        int areal = køkken.getAreal();
        for (int i=0; i<rum.length; i=i+1)
            areal = areal + rum[i].getAreal();
        return areal;
    }

    public int getAndelsværdi() {
        return getAreal() * andelsværdi;
    }
}
```

## Question 4.4

The class RingeEtVærelses is declared below as a subclass of class EtVærelses.

```
class RingeEtVærelses extends EtVærelses {
    private int nedskriv;

    RingeEtVærelses(Rum rum, Køkken køkken, int andelsværdi, int nedskriv) {
        super(rum, køkken, andelsværdi);
        this.nedskriv = nedskriv;
    }

    public int getAndelsværdi() {
        return super.getAndelsværdi() - nedskriv;
    }
}
```