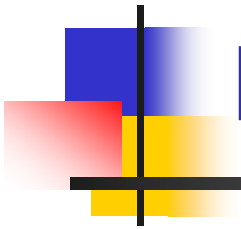


INF1000 (Uke 15)
Eksamen V 04

Grunnkurs i programmering

Institutt for Informatikk

Universitetet i Oslo

Anja Bråthen Kristoffersen og Are Magnus Bruaset

UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

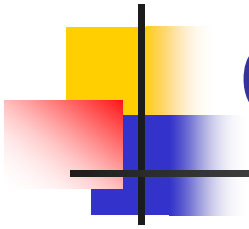
Eksamen i	INF1000 — Grunnkurs i objektorientert programmering
Eksamensdag:	11. juni 2004
Tid for eksamen:	9.00 – 12.00
Oppgavesettet er på 8 sider.	
Vedlegg:	Ingen
Tillatte hjelpemidler:	Alle

Kontroller at oppgavesettet er komplett før du begynner å besvare spørsmålene.

Innhold

Oppgave 1 Flervalgsoppgave (vekt 30%)

Denne oppgaven består av mange små deloppgaver. For alle unntatt deloppgavene 1d og 1e er det oppgitt diverse svaralternativer. Et vilkårlig antall av disse alternativene kan være riktige — ingen, ett eller flere. Ditt svar skal *bare* være tallet eller tallene for de alternativene du tror er riktig. Du vil få poeng for alle riktige svar, og trekk for alle gale forslag.



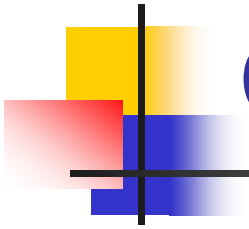
Oppgave 1a

Vi har et program med følgende setninger:

```
String tre = "3";
System.out.println(tre+4);
```

Hva skrives ut når programmet kjøres?

1. `34`
2. `7`
3. `tre4`
4. Ingen av alternativene ovenfor



Oppgave 1b

Vi har et program med følgende setninger:

```
int tre = 3;
System.out.println(4+tre);
```

Hva skrives ut når programmet kjøres?

1. `43`
2. `7`
3. `4tre`
4. Ingen av alternativene ovenfor

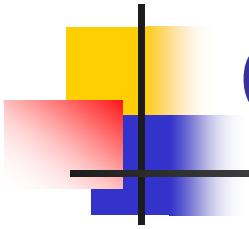
Oppgave 1c

Vi har et program med følgende setninger:

```
int k = 0;
boolean fortsett = true;
while (k < 5 && fortsett) {
      if (k == 3) fortsett = false;
      System.out.println("HEI");
}
```

Hvor mange ganger skrives teksten "HEI" ut?

1. 0
2. 3
3. 4
4. 5
5. Ingen av alternativene ovenfor



Oppgave 1d

Vi har et program med følgende setninger:

```
String s = "apekatt";
String t = "";
int n = s.length();
for (int pos = n-1; pos >= 0; --pos)
     t += s.substring(pos,pos+1);
System.out.println(t);
```

Hva skjer når programmet kjøres?

```
> ttakepa
```

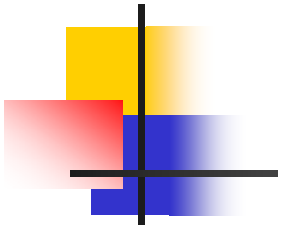
Oppgave 1e

Vi har et program med følgende setninger:

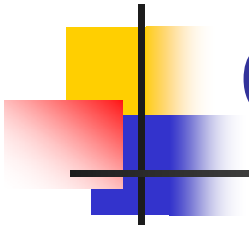
```
String s = "hus";
int j = 34;
int n = s.length();
if (n > 5 && j == 34) {
      if (j < 40)
            System.out.println("A");
} else
      if (s.charAt(2) == 's' || j > 40)
            System.out.println("B");
      else
            System.out.println("D");
```

Hva skrives ut når programmet kjøres?

> B



```
1  import easyIO.*;
2  import java.util.*;
3
4  class Dato {
5      private int dag, måned;
6      public int år;
7
8      Dato () {
9          dag = 1;  måned = 1;  år = 2004;
10     }
11
12     Dato (int d, int m, int å) {
13         dag = d;  måned = m;  år = å;
14     }
15
16     Dato les (In fil) {
17         return ...
18     }
19
20     public static void main (String arg[]) {
21         Dato d1 = new Dato(11, 6);
22
23         Dato d2 = new Dato();
24         ++d2.år;
25
26         Dato d3 = Dato.les(new In());
27
28         HashMap h = new HashMap();
29         h.put("nyttår", new Dato());
30         h.put("påske", new Dato(11,4,2004));
31
32     }
33 }
```

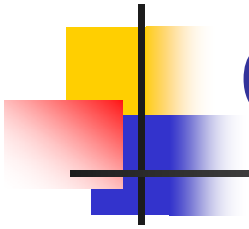


Oppgave 1f

De følgende spørsmålene besvares medutgangspunkt i Java-koden vist i figur 1 på neste side.

Hvilken verdi får d1.år etter definisjonen i linje 21?

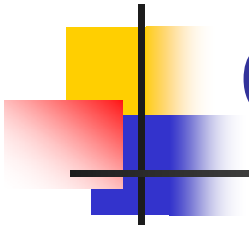
1. `1`
2. `11`
3. `2001`
4. `2004`
5. Noe annet
6. Definisjonen er ulovlig



Oppgave 1g

Hvilken verdi får d2.år etter at `++`-operasjonen i linje 24 er utført?

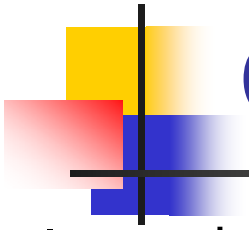
1. `0`
2. `1`
3. `2000`
4. `2004`
5. `2005`
6. Noe annet
7. Operasjonen er ulovlig



Oppgave 1h

Hva skjer hvis vi setter en static foran deklarasjonen av år i linje 6?

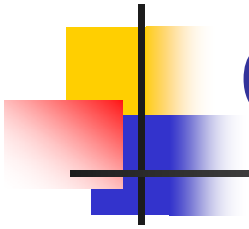
1. Ingen forskjell
2. Alle datoer vi lagrer vil ha samme år
3. Alle datoer vi lagrer vil være like
4. Endringen er ulovlig



Oppgave 1i

I metoden `les` ønsker vi å lese tre heltall som utgjør en dato fra filen `fil`. Hva er lovlige return-setninger i linje 17 for å få dette til?

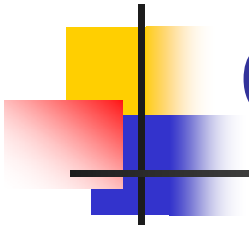
1. return new Dato();
2. return (Dato)new Dato();
3. return fil.inDato();
4. return new Dato(fil.inInt(), fil.inInt(), fil.inInt());
5. return new Dato(3*new inInt());
6. return Dato.in(fil):
7. return (Dato)fil.inText();



Oppgave 1j

Hva er korrekte uttrykk for å få tak i datoen for påskedagen etter at linje 30 er utført?

1. `h.get("påske")`
2. `h.get("PÅSKE")`
3. `(Dato)h.get("påske")`
4. `(Dato)h.get("PÅSKE")`
5. `(String)h.get("påske")`
6. `(String)h.get("PÅSKE")`
7. `(Bil)h.get("påske")`
8. `(Bil)h.get("PÅSKE")`



Oppgave 2 (vekt 35%)

Anta at vi har to filer med værdata på samme format som i den siste obligatoriske oppgave.

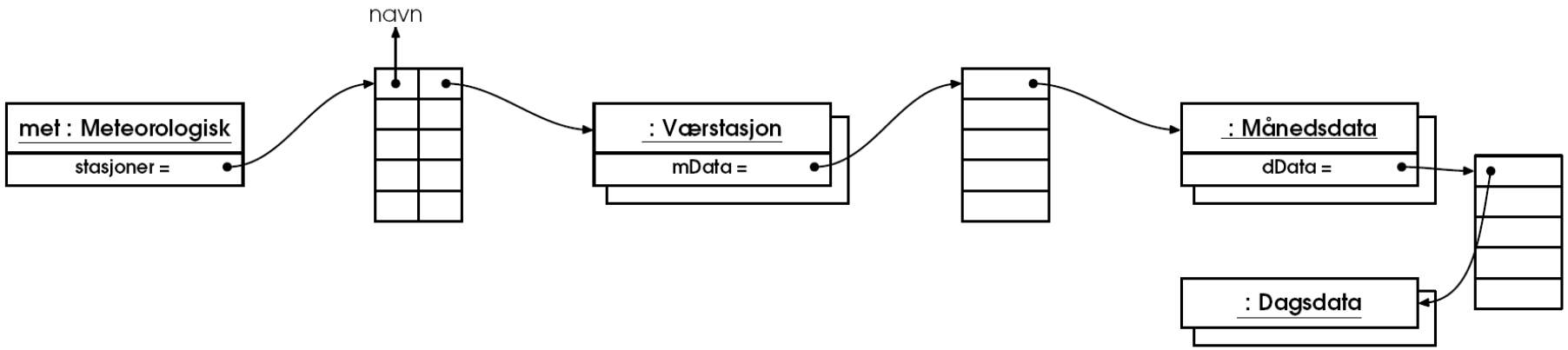
Det er mange måter å lagre dataene på; i denne oppgaven antar vi at det er gjort slik det er vist i figur 2 på neste side.

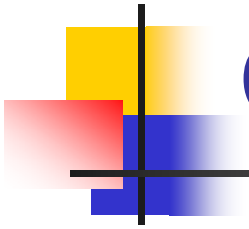
```
import java.util.*;

class Meteorologisk {
        HashMap stasjoner = new HashMap();
        public static void main (String arg[]) {
                Meteorologisk met = new Meteorologisk();
                // ...
        }
}

class Værstasjon {
        int nummer, hoh;
        String navn, kommune, fylke;
        boolean østlandet, vestlandet;
        Månedsdata mData[] = new Månedsdata[12];
        //...
}

class Månedsdata {
        int månedsNr;
        Dagsdata dData[] = new Dagsdata[31];
        //...
}

class Dagsdata {
        double vind, nedbør, temp_min, temp_max;
        //...
}
```

Oppgave 2a

Tegn UML klassediagram og objektdiagram for datastrukturen.

objektdiagram



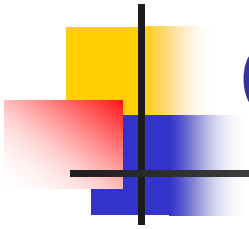


Oppgave 2b

Finn data

Skriv Java-kode (som skal ligge i `main`) for å finne nedbøren den 15.3 ved stasjon **GARDERMOEN**.

```
Værstasjon gmoen = (Værstasjon)met.stasjoner.get("GARDERMOEN");

double nb15mars = gmoen.mData[3-1].dData[15-1].nedbør;
```



Oppgave 2c

Finn kaldeste observasjon

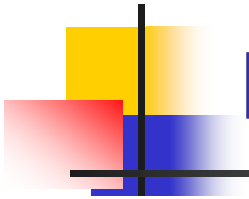
Skriv Java-kode (som skal ligge i `main`) som finner den laveste temperaturen som er observert og skriver den ut.

En enkel men oversiktlig måte å gjøre det på er å la klassene Meteorologisk, Værstasjon og Månedsdata ha hver sin metode mintemp som finner minimum blant sine data.

NB! Denne koden er litt for enkel til å være helt riktig: den antar at det finnes data for alle månedene i året og at alle måneder har 31 dager. Men en slik løsning vil bli godtatt til denne eksamenen.

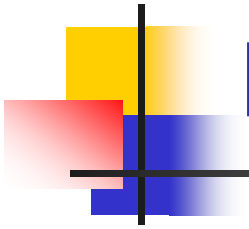
Koden i main

```
double minTemp = met.minTemp();
```



Metoden minTemp i Meteorologisk

```
double minTemp() {
      double min = 999;
      Iterator it = stasjoner.keySet().iterator();
      while (it.hasNext()) {
            Værstasjon vs = (Værstasjon)it.next();
            double mx = vs.minTemp();
            if (mx < min){
                  min = mx;
            }
      }
      return min;
}
```



Metoden minTemp i Værstasjon

```
double minTemp() {
      double min = 999;
      for (int m = 0; m < 12; ++m) {
            double mx = mData[m].mintemp();
            if (mx < min) {
                  min = mx;
            }
      }
      return min;
}
```

Metoden minTemp i Månedsdata

```
double mintemp () {
      double min = 999;
      for (int d = 0; d < 31; ++d) {
            double mx = dData[d].temp_min;
            if (mx < min){
                  min = mx;
            }
      }
      return min;
}
```

Oppgave 3
Personnummer (vekt 35%)

Fra og med 1964 har hver person med fast bosted i Norge fått tildelt et 11-sifret **fødselsnummer**. De seks første sifrene angir fødselsdato på formatet **ddmmåå**, der **dd**, **mm** og **åå** angir posisjoner for sifrene i henholdvis dag, måned og år. Bare de to siste sifrene av årstallet brukes. De fem siste sifrene i fødselsnummeret kalles **personnummer**.

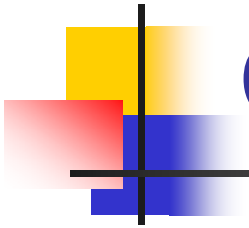
Et personnummer er delt inn i to grupper. De tre første sifrene angir et individnummer som skal skille personer født på samme dato. Det tredje av disse sifrene vil også angi kjønn, ved at kvinner får tildelt et partall, mens menn får tildelt et oddetall. De to siste sifrene I personnummeret er **kontrollsifre** som beregnes ut fra alle foregående siffer i fødselsnummeret.

Eksempel Betrakt fødselsnummeret

081273 24604

Dette tilhører en person som er født 8. desember 1973. Siden det niende sifferet (6) er et partall, er denne personen en kvinne. Kontrollsifrene i de to siste posisjonene, 0 og 4, er beregnet ut fra de foregående sifrene. I denne oppgaven skal du skrive innmaten i to metoder som inngår i en klasse Fødselsnummer som er vist i figur 3 på forrige side.

```
class Fødselsnummer {
          // Lagring av fødselsnummer, siffer for siffer
          int[] s = new int[11];
          // Faktorer for kontroll av fødselsnummer
          int[] a = {3,7,6,1,8,9,4,5,2};
          int[] b = {5,4,3,2,7,6,5,4,3,2};
          Fødselsnummer (String fnr) {
                    // Kontrollerer at fnr er en tekststreng bestående av 11 siffer.
                    // Deler opp strengen siffer for siffer. Lagrer hvert siffer
                    // som et heltall i arrayen s slik at det første sifferet ligger
                    // i s[0], og det siste i s[10].
                    // Du kan anta at denne metoden allerede finnes og oppfører
                    // seg som beskrevet ovenfor. Denne metoden skal derfor
                    // ikke skrives.
                    ...
          }
          boolean delelig11 (int tall) {
                    // Tester om fødselsnummeret er delelig med 11. Hvis det
                    // er tilfelle returneres verdien true, ellers returneres
                    // verdien false.
                    int tmp = tall/11;
                    return (11 * tmp == tall);
          }
          boolean gyldig () {
                    // Tester om fødselsnummeret som er lagret i arrayen s er gyldig. Hvis det er
                    //tilfelle returneres verdien true, ellers returneres verdien false.
                    ...
          }
          String tekst (boolean bare_dato) {
                    // Gjør om sifre som er lagret i arrayen s til en tekststreng og returnerer
                    //denne.
                    ...
          }
}
```



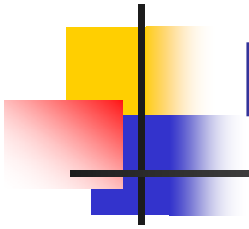
Oppgave 3a

For å unngå feil ved inntasting av fødselsnummere i et datasystem er det nødvendig å kontrollere om de inntastede sifrene er gyldige. Du skal gjøre slik kontroll tilgjengelig i klassen ovenfor ved å skrive innholdet i metoden gyldig. Algoritmen for sjekking av kontrollsifrene er som følger:

1. Beregn summen

$$sum1 = 3*s[0] + 7*s[1] + 6*s[2] + \ldots + 5*s[7] + 2*s[8]$$

der de ni første sifrene i fødselsnummeret hentes fra arrayen s deklarert i klassen Fødselsnummer. I et gyldig fødselsnummer skal summen av *sum1* og det første kontrollsifferet (*s*[9]) være delelig med 11. Benytt arrayen a og metoden delelig11 som begge er definert ovenfor.



Fortsettelse av oppgave 3a

2. Beregn summen

$$sum2 = 5\ s[0] + 4\ s[1] + 3\ s[2] + \ \ +3\ s[8] + 2\ s[9]$$

der de ti første sifrene i fødselsnummeret er hentet fra arrayen s deklarert ovenfor. I et gyldig fødselsnummer skal summen av *sum2* og det andre kontrollsifferet (*s*[10]) være delelig med 11. Benytt arrayen b og metoden delelig11 som begge er definert ovenfor.

Begge kontrollsifrene må være gyldige for at fødselsnummeret skal aksepteres.

Observer at arrayene a og b i klassen Fødselsnummer inneholder vektene som brukes til å beregne summene sum1 og sum2. I tillegg har du en metode som sjekker om et tall er delelig med 11 eller ikke. Metoden gyldig kan dermed skrives slik:

sum1 regnes ut:

```
int sum1 = 0;
for (int i = 0; i < 9; ++i){
      sum1 += a[i]*s[i];
}
```

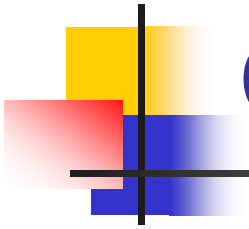
sum2 regnes ut:

```
int sum2 = 0;
for (int i = 0; i < 10; ++i){
      sum2 += b[i]*s[i];
}
```

Bruker funksjonen delelig11 som returnerer en boolsk varabel til å finne ut om (sum1 + s[9]) og (sum2 + s[10]) er delelig på 11

```
delelig11(sum1+s[9]);
delelig11(sum2+s[10]);
```

```
boolean gyldig () {
// Tester om fødselsnummeret som er lagret i arrayen s er
// gyldig. Hvis det er tilfelle returneres verdien true,
// ellers returneres verdien false.

      int sum1 = 0;
      int sum2 = 0;
      for (int i = 0; i < 9; ++i){
            sum1 += a[i]*s[i];
      }
      for (int i = 0; i < 10; ++i){
            sum2 += b[i]*s[i];
      }
      boolean ok = delelig11(sum1+s[9]);
      ok = ok && delelig11(sum2+s[10]);

      return ok;
}
```



Oppgave 3b

I forbindelse med utskrifter er det hensiktsmessig med en tekstlig representasjon av et fødselsnummer. Klassen Fødselsnummer har derfor en metode tekst(boolean bare_dato) som kan gjøre om sifrene i arrayen s til en tekststreng og returnere denne. Hvis parameteren bare_dato har verdien true, returneres kun fødselsdatoen (seks siffer), ellers returneres hele fødselsnummeret (11 siffer). I det siste tilfellet skal fødselsdatoen og personnummeret separeres ved å sette inn et blankt tegn i teksten som returneres.

Skriv innholdet i metoden tekst.

Vi må bygge opp en tekst bestående av sifrene lagret i arrayen s. Dette kan gjøres på flere måter, men den enkleste er å benytte pluss-operatoren til konkatenering av en tekst med et heltall (som konverteres til tekst). Metoden tekst kan skrives slik:

Tom tekst for å kunne konkatenere med +:

```
String t = "";
```

Løkke for å konvertere et og et tall til tekst (konverterer først de 6 første tallene = datoen som alltid skal skrives ut).

```
for (int i = 0; i < 6; ++i){
        t += s[i];
}
```

Sjekker om den boolske variabelen `bare_dato` er `false` før resten evt. konverteres til tekst:

```
if (!bare_dato) {
        t += " ";
        for (int i = 6; i < 11; ++i){
                t += s[i];
        }
}
```

```
String tekst (boolean bare_dato) {
// Gjør om sifre som er lagret i arrayen s til
// en tekststreng og returnerer denne. Hvis bare_dato
// har verdien true returneres bare fødselsdato
// (s[0],...,s[5]), ellers returneres alle sifrene
// s[0],...,s[10], med et blankt
// tegn satt inn mellom fødselsdato og personnummer.

      String t = ""; // Tom tekst for å kunne konkatenere med +
      for (int i = 0; i < 6; ++i)
            t += s[i];
      if (!bare_dato) {
            t += " ";
            for (int i = 6; i < 11; ++i){
                  t += s[i];
            }
      }
      return t;
}
```