



Prøveeksamen INF1000, h2005 28. november

Arne Maus og Arild Waaler

Prøveeksamen i : INF1000 — Grunnkurs i objektorientert programmering

- Les *nøye* gjennom hver oppgave før du løser den. For hver oppgave er angitt det maksimale antall poeng du kan få hvis du svarer helt riktig. Summen av poengene er 304, slik at f.eks 5 poeng tilsvarer omlag 3 minutter og 8 poeng omlag 5 min. (hvis du regner med å komme igjennom alt). Pass på at du bruker tiden din riktig.
- Kontroller også at oppgavesettet er komplett før du begynner å besvare det. Dersom du savner opplysninger i oppgaven, kan du selv legge dine egne forutsetninger til grunn og gjøre rimelige antagelser, så lenge de ikke bryter med oppgavens "ånd". Gjør i så fall rede for forutsetningene og antagelsene du gjør.
- Dine svar *skal* skrives på disse oppgavearkene, og *ikke* på separate ark. Dette gjelder både spørsmål med avkryssningssvar og spørsmål hvor du bes om å skrive programkode. I de oppgavene hvor det skal skrives programkode, anbefales det at du først skriver en kladd på eget ark før du fører svaret inn i disse oppgavearkene på avsatt plass.
- Noen av spørsmålene er flervalgsoppgaver. På disse oppgavene får du poeng etter hvor mange korrekte svar du gir. Du får ikke poeng hvis du lar være å besvare et spørsmål, eller dersom du krysser av begge svaralternativer.
- Hvis du har satt et kryss i en avkrysningsboks og etterpå finner ut at du ikke ønsket å krysse av der, kan du skrive "FEIL" like til venstre for den aktuelle avkrysningsboksen.

Teksten over er identisk til forsiden av oppgaveteksten dere får til den ordinære eksamen. Dette oppgavesettet er nesten helt likt som for den ordinære eksamen hva angår formatet, dvs. antall oppgaver og antall poeng pr. oppgave. Men spørsmålene er selvsagt ikke de samme



Sensurer deg selv

- Flervalgstester med M alternativer
 - 0 eller max/M poeng per svar
- Andre oppgaver
 - Helt riktig = Max
 - Nesten helt riktig = Max- 2
 - Endel riktig = Max/2
 - Litt riktig = 2-5

Oppgave 1 (8 poeng)

a) Hvor mange double-verdier er det plass til i arrayen dim3?

```
double[][][] dim3 = new double[2][4][6];
```

Svar:48.(=2*4*6).....

b) Hvor mange ganger blir "INF1000" skrevet ut av følgende løkke:

```
for (int j=20; j >= 10; j-- )  
    System.out.println("INF1000");
```

Svar:11.....

c) Hvor mange ganger blir "INF1000" skrevet ut av følgende løkke:

```
for (int j= 1; ++j < 5; j++)  
    System.out.println("INF1000");
```

Svar:2.....

d) Hvor mange ganger blir "INF1000" skrevet ut av følgende dobbelt-løkke:

```
for (int j=0; j <4 ; j++){  
for (int i= 4-j; i > 0 ; i--)  
    System.out.println("INF1000");  
}
```

Svar:10.....

Oppgave 2 (16 poeng)

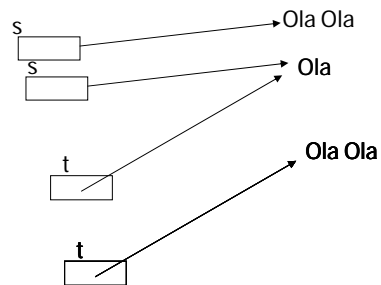
Er disse programsetningene lovlige i Java?

- | JA | NEI | |
|-------------------------------------|-------------------------------------|--|
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>int i = int j=1;</code> |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | <code>int i=3, j=i-3,k;</code> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>int[String[]] intString = new int[new String[2]];</code> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>boolean[] b = new boolean[true];</code> |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | <code>boolean b = (true != true);</code> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>boolean b = (true => true);</code> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>int b = new int;</code> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>char c = 'true';</code> |

Oppgave 3 (5 poeng)

Anta at følgende kodelinjer utføres:

```
String s = "Ola";  
String t = s;  
if( s==t ) {  
    s = s+t;  
    t = t+t;  
}  
if( s==t ) System.out.println("Identiske");  
else System.out.println("Ikke identiske");  
if( s.equals(t) ) System.out.println("Like");  
else System.out.println("Ulike");
```



OPPGAVE 3
Ikke identiske
Like

Oppgave 4 (15 poeng)

Skriv ferdig metoden under, som med utgangspunkt i avstand og tid regner ut hastigheten ved å dividere avstand med tid. Hvis tiden er mindre eller lik null, eller hvis avstanden er negativ, skal det returneres -1. Svar:

```
double hastighet( double avstand, double tid ){
```

```
    if (tid<=0 || avstand < 0 ) return -1;  
    return (avstand/tid);
```

```
}
```

Oppgave 5 (30 poeng)

Anta at HashMap **bilene** er deklartert i en metode i et program, og at klassen **Bil** er deklartert et annet sted på samme fil:

```
class Bil {  
    String eier;  
    Bil(String e) {eier =e;}}
```

.....

```
HashMap<String,Bil> bilene = new HashMap<String,Bil>();
```

Vil de ulike setningene nedenfor resultere i at HashMapen **bilene** inneholder 10 biler med eiere: Preben1, Preben2,...,Preben10, og hvor søkenøkkelen er lik eierens navn;

JA NEI


```
int k = 0; String e = "Preben";  
    while (k <10) {  
        bilene.put(new Bil(e+k));  
    }
```

JA NEI

- ```
int k = 0; String e = "Preben";
while (k++ <10) {
 bilene.put(new String(e),new Bil(e+k));}
```
- ```
int k = 0;  
while (k++ <11) {  
    Bil b = new Bil("Preben"+(k-1));  
    bilene.put(b.eier,b)  
}
```
- ```
int k = 0;
while (++k <10) {
 String e = "Preben"+k;
 bilene.put(e,new Bil(e));
}
```
- ```
int k = 10;  
while (k > 0) {  
    Bil b = new Bil("Preben"+ k--);  
    bilene.put(new Bil(b.eier).eier, b);  
}
```
- ```
int k = 1;
while (k < 11) {
 Bil b = new Bil("Preben"+ k++);
 bilene.put(new String(b.eier), new Bil(b.eier));
}
```

### Oppgave 6 (15 poeng)

```
int tungvindt(int n, int m){
 int k = 0;
 while(n-->0)
 for(int i=0; i<m; i++)
 k++;
 return k;
}
```

Hva returneres fra metodekallet `tungvindt(371,100)`?

Svar: .....37100.....

### Oppgave 7 (15 poeng)

I oppgaven under skal du skrive en metode som ut fra fire deklarte arrayer i skal returnerer en tilfeldig setning, for eksempel: "Hans spiser i veien.". Til å gjøre det skal du bruke metoden `int nextInt(int k)` i klassen Java-klassen `Random`. Metoden vil for hvert kall returnere en int-verdi i intervallet `0, ..., k-1`. De globalt definerte variable som du kan bruke er: `Random tall = new Random(); String[] subjekt;` `String[] verb;` `String[] preposisjon;` `String[] objekt.` Du skal anta at disse fire arrayene hver har blitt initiert med like mange enkelt-ord av den typen som navnene antyder.

```
Random tall = new Random();

String[] subjekt = {"Han", "Hun", "Per", "Kari"};
String[] verb = {"sitter", "spiser", "går", "ser"};
String[] preposisjon = {"i", "på", "over", "under"};
String[] objekt = {"veien", "hagen", "taket", "huset"};
String lagNySetning(){

 return subjekt[tall.nextInt(subjekt.length)] + " "+
 verb[tall.nextInt(verb.length)] + " "+
 preposisjon[tall.nextInt(preposisjon.length)] + " "+
 objekt[tall.nextInt(objekt.length)];

}
```

### Oppgave 8 (25 poeng)

I denne oppgaven skal du lage en metode som returnerer en "vasket" tekststreng, dvs. en streng fremkommet ved å fjerne alle forekomster i strengen tekst av tegn som også forekommer i strengen skilletegn. For eksempel vil kallet: vask("spisetiden er 0700-1000", "s0i") returnere: "petden er 7-1". De to input-parametere til metoden skal ikke endres. Du må gjerne skrive ekstra hjelpemetoder om du ønsker å gjøre det.

```
String vask(String tekst, String skilletegn){
 String setning = "";
 for(int i=0; i<tekst.length(); i++)
 if(okTegn(tekst.charAt(i),skilletegn))
 setning += tekst.charAt(i);
 return setning;
}

boolean okTegn(char c, String s){
 for(int i=0; i<s.length(); i++)
 if(c==s.charAt(i)) return false;
 return true;
}
```

```
import easyIO.*;

class MultipleChoice {
 public static void main(String [] args) {
 Oppgavesett os = new Oppgavesett("oppgavesett.txt");
 os.skrivProsentResultater();
 }
}

class Oppgavesett {
 char[] fasit = new char[ANT_OPPG];
 int[] poengsum = new int [ANT_OPPG+1];
 int[] riktigsvar = new int [ANT_OPPG];
 In fil;
 int antallKandidater = 0;

 Oppgavesett(String filnavn) {
```

### Oppgave 9 (30 poeng)

Du skal i denne oppgaven lage et system for å rette "multiple choice"-oppgaver. Oppgavesettene er organisert som en rekke spørsmål der hvert spørsmål har ett og bare ett riktig svar. Første linje i filen "oppgavesett.txt" inneholder en sekvens av 18 bokstaver på en linje, for eksempel:

```
a c e a f a b b a d e f f d b a a b
```

Dette betyr at oppgavesettet har totalt 18 oppgaver, der riktig svar på den første oppgaven er a og riktig svar på den siste oppgaven er b. Deretter følger samtlige svar linje for linje, der hver linje begynner med et studentnummer og fortsetter med en sekvens av studentens svar:

```
22 a c e c c c c c c c c c c c c c c c c c c
```

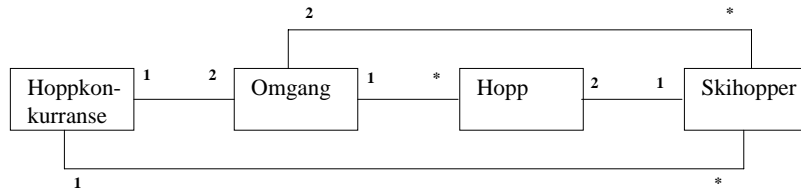
Denne linjen innebærer at student 22 har gitt svar riktig på de tre første spørsmålene og feil svar på resten. Du kan anta at hver student har svart på alle spørsmålene (og ingen flere). Hvert riktig svar gir 1 poeng mens feil svar gir 0, slik at student 22 får 3 poeng. Du skal lage et fullstendig program som leser inn filen og skriver ut til skjerm studentnummer og poengsum for alle som har besvart. Videre skal du skrive ut statistikk. Først prosentantall som fikk 0 poeng, så prosentantall som fikk 1 poeng, osv. Deretter skal du skrive prosentantall som klarte oppgave 1, prosentantall som klarte oppgave 2, osv. Bruk variabelen **poengsum** til å lagre, for hver mulig totalsum, hvor mange som fikk denne poengsummen. Bruk variabelen **riktigsvar** til å lagre, for hver oppgave, hvor mange som fikk til denne oppgaven. I konstruktøren skal alle de oppgitte variablene initialiseres med data fra input-filen.

```
Oppgavesett(String filnavn) {
 genererTestFil(filnavn);
 fil = new In(filnavn);
 System.out.print("Fasit: ");
 for(int i=0; i<ANT_OPPG; i++){
 fasit[i] = fil.inChar(" ");
 poengsum[i]=riktigsvar[i]=0;
 }
 fil.readLine();
 while(!fil.endOfFile()){
 antallKandidater++;
 int sum = 0;
 System.out.print("Nr " + fil.inInt(" ") +": ");
 for(int i=0; i<ANT_OPPG; i++)
 if(fasit[i]==fil.inChar(" ")){
 sum++;
 riktigsvar[i]++;
 }
 fil.readLine();
 System.out.println(sum + " poeng");
 poengsum[sum]++;
 }
 System.out.println("Antall kandidater: " + antallKandidater);
}
```



### Oppgave 12 (20 poeng)

I en hopp-konkurranse deltar et antall skihoppere. En konkurranse består av to omganger. I begge omgangene starter alle hopperne, og hver hopper hopper ett hopp i hver omgang. Tegn et UML-klassediagram med de 4 (Java-)klassene som kan brukes til å representere systemet. Gi navn på disse klassene og plasser antall på forholdet mellom disse klassene.



### Oppgave 13 (25 poeng) – meget vanskelig

Gitt følgende program:

```

final int N = 6;

void skriv(int n){
 if(n==N) return;
 for(int i=0; i<n; i++)
 System.out.print("**");
 System.out.println();
 skriv(n+1);
 for(int i=0; i<n; i++)
 System.out.print("**");
 System.out.println();
}

```

Hvilken skjermutskrift genereres ved metodekallet `skriv(1);`?

Svar:

```

*
**

**
*

```

### Oppgave 14 (50 poeng)

Du skal i denne oppgaven skrive fullstendig kode for to klasser: Kunde og Arkiv. Kundeklassen skal lagre informasjon om kundenummer og navn, som begge skal være String-variabel, samt saldo av type double. Arkiv-klassen skal tilby 6 metoder med følgende parametere og returverdier:

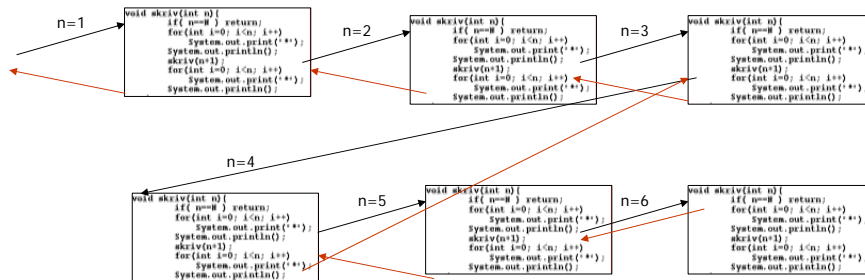
```

Arkiv(String filnavn)
boolean registrer(Kunde k)
boolean slettKunde(String kundenr)
void skrivAlleMedNegativSaldo()
Kunde hentKunde(String kundenr)
void lagreTilFil()

```

Konstruktøren skal ta inn fra tastatur et navn på fila som den bruker til å initialisere datastrukturen. Du må selv bestemme formatet til datafilen. Metoden `registrer` skal oppdatere arkivet ved å lagre et nytt kundeobjekt. Den returnerer true hvis det finnes et objekt med samme kundenummer i arkivet fra før, som i så fall slettes. Metoden `slett` skal slette kundeobjektet med angitt kundenr og returnere true hvis det finnes et objekt med dette kundenummer i arkivet fra før (som i så fall slettes). Den neste metoden skal skrive ut all informasjon om kunder med negativ saldo til skjerm. De to siste metodene er selvforklarende ut fra metodenavnen. Datastrukturen i klassen `Arkiv` skal deklarerer private. Dette er ikke nødvendig for klassen `Kunde`.

Svar:



N.B. Hver av disse metodene har sin egen n-variabel. Hvert metodekall er, som på tegningen, egne 'metode-objekter'. i-variablene er jo også flere av (faktisk 2 stk i hvert metode-objekt), men de er som vanlig lokale

```

class Kunde {
 String kundenr, navn;
 double saldo;

 Kunde(String kundenr, String navn, double saldo){
 this.kundenr = kundenr;
 this.navn = navn;
 this.saldo = saldo;
 }

 public String toString(){
 return "Knr. " + kundenr + " [" + navn + "]: " + saldo;
 }

 String tilFil(){
 return kundenr + ":" + navn + ":" + saldo;
 }
}

```

```

import java.util.*;
import easyIO.*;

class Arkiv {

 private HashMap<String,Kunde> kundearkiv = new
 HashMap<String,Kunde>();
 private In fil;
 private Out utfil;

 Arkiv(String filnavn){
 fil = new In(filnavn);
 utfil = new Out(filnavn);
 while(!fil.endOfFile()){
 String knr = fil.inWord(":");
 String navn = fil.inWord(":");
 Double saldo = fil.inDouble(":");
 kundearkiv.put(knr,
 new Kunde(knr,navn,saldo));
 fil.readLine();
 }
 }
}

```

```

boolean registrer(Kunde k){
 return (null != kundearkiv.put(k.kundenr,k));
}

Kunde hentKunde(String kundenr){
 return kundearkiv.get(kundenr);
}

boolean slettKunde(String kundenr){
 return (null != kundearkiv.remove(kundenr));
}

void lagreTilFil(){
 System.out.println("Lagrer til fil:");
 for(Kunde k: kundearkiv.values()){
 System.out.println(" " + k.tilFil());
 utfil.outln(k.tilFil());
 }
 utfil.close();
}

void skrivAlleMedNegativSaldo(){
 System.out.println("Negativ saldo:");
 for(Kunde k: kundearkiv.values())
 if(k.saldo<0) System.out.println(" " + k);
}
}

```

## Sensur – hva fikk du?

- N.B Grensene justeres etter ferdig poengsensur. Gjetter nå flg:
  - A = 304 - 290 10%
  - B = 289 - 260 25 %
  - C = 259 - 220 30%
  - D = 219 - 190 25%
  - E = 189 - 160 10%
  - F = resten

Strykprosenten avgjøres separat.  
Leverer f.eks alle gode nok besvarelser, stryker ingen