

# UNIVERSITETET I OSLO

## Det matematisk-naturvitenskapelige fakultet

Eksamen i : INF1000 — Grunnkurs i objektorientert programmering  
Eksamensdag : Fredag 1. desember 2006  
Tid for eksamen : 14.30 – 17.30  
Oppgavesettet er på : 13 sider  
Vedlegg : Ingen  
Tillatte hjelpemidler : Alle trykte og skrevne

- Les gjennom hele oppgaven før du begynner å løse oppgaven. Kontroller 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 såfall rede for forutsetningene og antagelsene du gjør.
- Dine svar skal skrives på disse oppgavearkene, og ikke på separate ark. 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.
- Husk å skrive såpass hardt at besvarelsen blir mulig å lese på alle gjennomslagsarkene, men ikke legg andre deler av eksamensoppgaven under når du skriver.

### Oppgave 1 (8 poeng)

Er disse programsetningene lovlige i Java?

JA	NEI	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<code>int[][] a = new int[100];</code>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<code>int[3][5] tabell = new int[3][5];</code>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<code>char tegn = "@";</code>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<code>String tekst = 1 + 2 + 3 + " kr";</code>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<code>String[] s = {"Siri"};</code>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<code>String[] s = new String;</code>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<code>double svar = 2 * svar;</code>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<code>for (int i=0; int i&lt;10; int i+=2) {System.out.print(i);}</code>

## Oppgave 2 (10 poeng)

Anta at vi har følgende metode:

```
boolean erLike(String s, String t) {
    if (s.length() == t.length()) {
        int antall = 0;
        for (int i=0; i<s.length(); i++) {
            boolean b1 = s.charAt(i) == '*' || t.charAt(i) == '*';
            boolean b2 = s.charAt(i) == t.charAt(i);
            if (b1 || b2) {
                antall++;
            }
        }
        return (antall == s.length());
    } else {
        return false;
    }
}
```

Kryss av hvilken verdi variabelen **like** får i hvert av disse tilfellene:

TRUE	FALSE	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	boolean like = erLike("ABC", "ABC*");
<input checked="" type="checkbox"/>	<input type="checkbox"/>	boolean like = erLike("HEI", "H*I");
<input checked="" type="checkbox"/>	<input type="checkbox"/>	boolean like = erLike("E*sam*n", "Eks*m*n");
<input checked="" type="checkbox"/>	<input type="checkbox"/>	boolean like = erLike("****", "AAA");
<input checked="" type="checkbox"/>	<input type="checkbox"/>	boolean like = erLike("", "");

## Oppgave 3 (6 poeng)

a) Hvor mange int-verdier er det plass til i hver av disse arrayene?

```
int[] sauer = new int[9];
int[][] hester = new int[10][11];
int[][] geiter = new int[9][9];
```

Svar: .. **9, 110, 81** .....

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

```
for (int j=10; j<100; j=j+10 ) {
    for (int i=j-1; i<j+3; i++) {
        System.out.println("INF1000");
    }
}
```

Svar: ... **36** .....

### Oppgave 4 (9 poeng)

Hvilken verdi får variabelen **resultat** i hvert av disse tilfellene?

a) `int resultat = 1;`  
`for (int i=0; i<4; i++) {`  
    `resultat = 2 * (i + resultat);`     2, 6, 16, 38  
`}`

Svar: .....38.....

b) `int resultat = 0;`  
`for (int i=0; i<5; i++) {`  
    `for (int j=i; j<5; j++) {`     0, 1, 2,3,4  
        `resultat += 1;`     0-4, 1-4, 2-4, 3-4, 4-4  
    `}`  
`}`

Svar: .....15.....

c) `int resultat = 0;`  
`for (int i=1; i<4; i++) {`     1, 2, 3  
    `int prod = 1;`  
    `for (int j=1; j<=i; j++) {`     1, 1-2, 1-3  
        `prod = prod * j;`  
    `}`  
    `resultat += prod;`  
`}`

Svar: .....32.....

### Oppgave 5 (10 poeng)

Anta at programmet "Regnemaskin.java" ser slik ut:

```
class Regnemaskin {
    public static void main(String[] args) {
        // Se oppgaveteksten
    }
}
```

```
class Analyse {
    int a;
    int b;

    Analyse(int a, int b) {
        this.a = a;
        this.b = b;
    }

    double sum(double[] x) {
        double svar = 0;
        for (int i=a; i<b; i++) {
            svar += x[i];
        }
        return svar;
    }
}
```

```
}
```

La innholdet av **main**-metoden være et av alternativene under. Hvilke alternativer gir feil? (med 'feil' menes enten kompilator-feil eller kjørefeil). Hva blir verdien til **resultat** for de alternativene som ikke fører til feil?

```
1. double[] y = {2.0, 3.0, 4.0};
   double resultat = Analyse.sum(y);
```

Svar: .....Feil, sum er ikke statisk .....

```
2. Analyse ana = new Analyse();
   double[] y = {2.0, 3.0, 4.0};
   double resultat = ana.sum(y);
```

Svar: .... Gal konstruktør.....

```
3. Analyse ana = new Analyse(0, 3);
   double[] x = {1, 2, 3, 4};
   double resultat = ana.sum(x);
```

Svar: .....6.....

```
4. Analyse ana = new Analyse(1, 5);
   double[] verdier = new double[3];
   verdier[0] = 3;
   verdier[1] = 5;
   verdier[2] = 8;
   double resultat = ana.sum(verdier);
```

Svar: .....Gale param til konstruktør.....

```
5. Analyse ana1 = new Analyse(0, 3);
   Analyse ana2 = new Analyse(4, 6);
   double[] tall = {1, 2, 3, 4, 5, 6, 7, 8};
   double resultat = ana1.sum(tall) + ana2.sum(tall);
```

Svar: .....17.....

### Oppgave 6 (10 poeng)

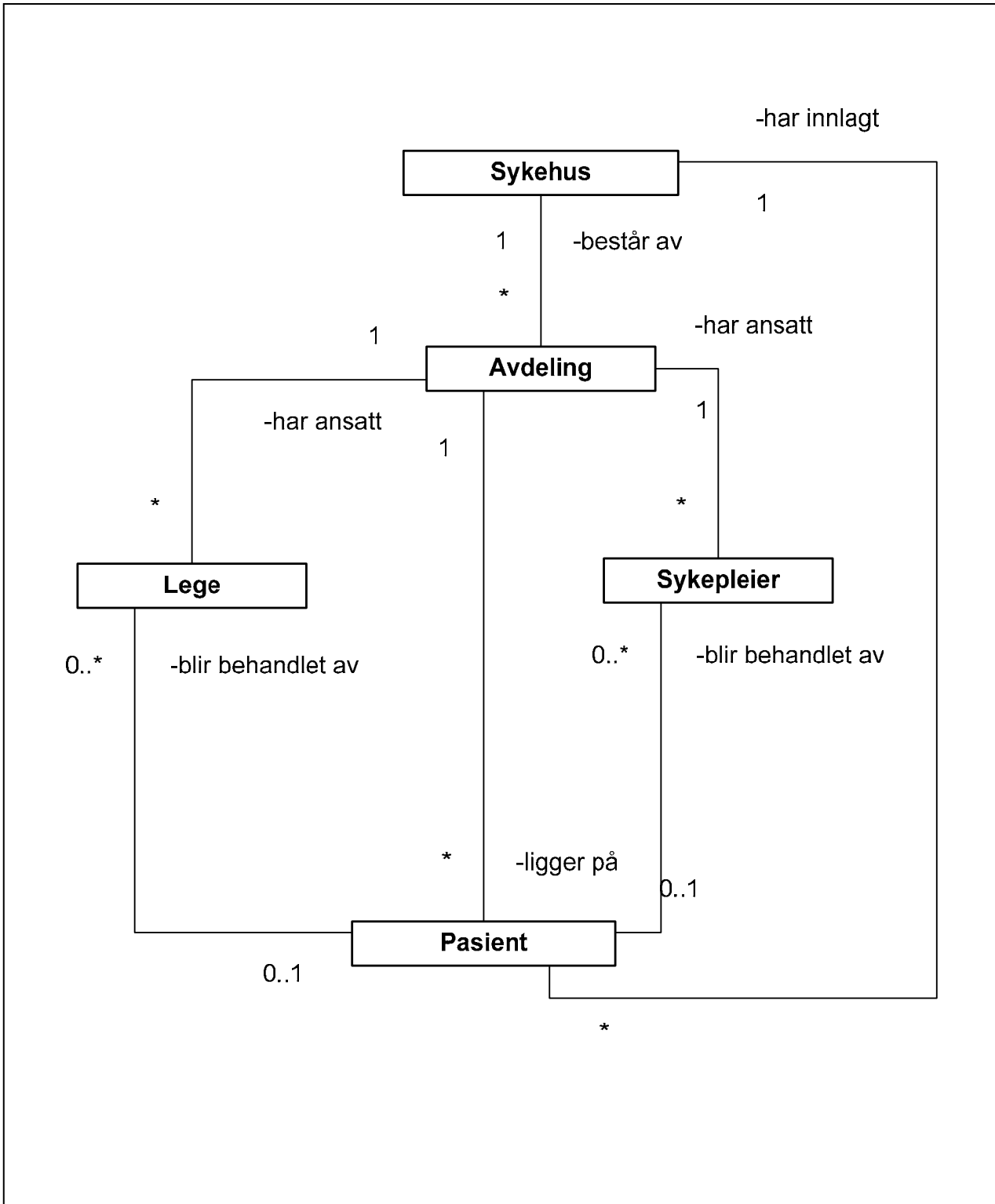
Skriv ferdig metoden under, som med utgangspunkt i en array med vindobservasjoner for et visst antall dager teller og returnerer antall dager med vindstyrke over 10.7 m/s (= kuling eller mer). Elementene i arrayen inneholder vindstyrke for hver sin dag, målt i meter per sekund (m/s). Svar:

```
int antallDagerMedMyeVind(double[] vind) {
    int antDager = 0;
    for (int i = 0; i < vind.length; i++)
        if (vind[i] > 10.7)
            antDager++;
    return antDager;
}
```

### Oppgave 7 (20 poeng)

Et sykehus består av et antall avdelinger. På hver avdeling er det ansatt et antall leger og sykepleiere. En lege/sykepleier kan ikke være ansatt på mer enn én avdeling. En pasient som legges inn på sykehuset innskrives på én avdeling hvor hun/han på et gitt tidspunkt blir behandlet av null eller flere leger og av null eller flere sykepleiere. En lege eller en sykepleier behandler høyst én pasient av gangen. Lag et UML-klassediagram mellom klassene Sykehus, Avdeling, Lege, Sykepleier og Pasient med assosiasjoner som nevnt ovenfor. Skriv også navn på assosiasjonene og antall på hver side av dem.

Svar:



## Oppgave 8 (20 poeng)

Nedenfor er skissen til et program som først skal lese en fil med informasjon om endel kontakter (en kontakt består av et navn, en adresse og et telefonnummer) og deretter gå i løkke og be brukeren om et navn og skrive ut all kontaktinformasjon knyttet til dette navnet.

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

class TelefonregisterMain {
    public static void main(String[] args) {
        Telefonregister tr = new Telefonregister();
        tr.lesFraFil("kontakter.txt");
        tr.snakkMedBruker();
    }
}

class Telefonregister {
    HashMap h = new HashMap();
    // Her skal det ligge to objektmetoder: lesFraFil og snakkMedBruker
}

class Kontakt {
    String navn, adresse, tlf;
    // Her skal det ligge en konstruktør og en utskriftsmetode
}
```

a) Lag først konstruktøren i klassen Kontakt. Konstruktøren skal ha tre parametre (en for hver av objektvariablene i klassen) og skal benytte parameterverdiene til å initiere objektvariablene. Svar:

```
Kontakt( String navn, String adresse, String adr) {
    this.navn = navn;
    this.adresse = adresse;
    this.tlf = tlf;
}
```

b) Lag utskriftsmetoden i klassen Kontakt. Metoden skal hete skrivUt og skal ikke ha noen parametre eller returverdi. Den skal skrive ut på skjermen verdiene til de tre objektvariablene i klassen. Svar:

```
void skrivUt() {
    System.out.println(" Kontakt: "+navn+", adr: " + adresse + ", tlf: " + tlf);
}
```

c) Lag metoden `lesFraFil` som skal ligge i klassen `Telefonregister` og som blir kalt på fra main-metoden i klassen `TelefonregisterMain`. Metoden `lesFraFil` skal ha et filnavn som parameter og skal lese fra denne filen som antas å inneholde informasjon om et antall kontakter. Filen er en tekstfil som inneholder først tre linjer for den første kontakten, deretter tre linjer for den neste kontakten, osv. De tre linjene for en kontakt inneholder henholdsvis navn, adresse og telefonnummer (i den rekkefølgen). Hver kontakt skal representeres med et objekt av klassen `Kontakt`, og du *skal* benytte konstruktøren fra punkt a) når du oppretter nye `Kontakt`-objekter. Programmet skal legge kontaktene inn i HashMapen `h` (med navn som søkeord).

Svar:

```
void lesFraFil (String f) {  
    In fil = new In(f);  
    while (fil.hasNext()) {  
        String n = fil.inLine();  
        String a = fil.inLine();  
        String t = fil.inLine();  
        Kontakt k = new Kontakt(n,a,t);  
        h.put(n,k);  
    }  
    fil.close();  
}
```

d) Lag metoden `snakkMedBruker` som skal ligge i klassen `Telefonregister` og som blir kalt på fra `main`-metoden i klassen `TelefonregisterMain`. Metoden `snakkMedBruker` skal inneholde en evig løkke hvor hver gjennomløp består i at programmet ber om og leser inn et navn fra terminal og deretter skriver ut på skjerm (ved hjelp av metoden `skrivUt`) all informasjon knyttet til dette navnet (eller en feilmelding hvis navnet er ukjent).

Svar:

```
void snakkMedBruker() {  
    In tast = new In();  
  
    while (true) {  
        System.out.print("\n Gi kontaktnavn:");  
        String navn = tast.inWord();  
        if ( h.containsKey(navn) ) {  
            Kontakt k = (Kontakt) h.get(navn);  
            k.skrivUt();  
        } else {  
            System.out.println("Fant ikke :'" + navn+ "' i registeret");  
        }  
    }  
}
```



## Oppgave 9 (30 poeng)

LabFix er et medisinsk laboratorium som foretar analyser av pasientprøver (blodprøver, urinprøver, osv). Hver forespørsel til laboratoriet kommer fra en bestemt lege og gjelder en bestemt pasient, og forespørselen omfatter et vilkårlig antall prøver foretatt på pasienten. Når laboratoriet har foretatt sine analyser, lager de en fil "Pasient.txt" knyttet til forespørselen. Denne har følgende struktur:

```
<lege-ID> <legens adresse>
<pasient-ID> <pasientens navn>
<prøve-ID> <prøvens resultat>
<prøve-ID> <prøvens resultat>
.....
<prøve-ID> <prøvens resultat>
```

De to øverste linjene gir generelle opplysninger om henholdsvis legen og pasienten. Deretter kommer det en linje for hver av prøvene som er foretatt på pasienten. Hver av disse linjene består av en prøve-ID (en kode som entydig identifiserer hva slags type prøve det er snakk om) etterfulgt av et prøveresultat (et desimaltall som angir resultatet av analysen). Du kan anta at de to feltene på hver linje i filen er atskilt av blanke tegn, og at det ikke forekommer blanke tegn i noen av ID'ene.

Din oppgave er å lage et program som leser filen ovenfor og som lager en ny fil "KommentertPasient.txt". Denne skal inneholde all informasjon i filen "Pasient.txt", men den skal *tillegg* inneholde en kommentar på slutten av hver prøveresultat-linje som angir (1) det fulle navnet på prøven og (2) om prøvens verdi er normal, for lav eller for høy. Til å gjøre dette har du også en annen fil til rådighet som heter "Laboratorieverdier.txt". Den inneholder en linje for hver type prøve LabFix kan analysere:

```
<prøve-ID> <prøvens fulle navn> <min> <max>
<prøve-ID> <prøvens fulle navn> <min> <max>
.....
<prøve-ID> <prøvens fulle navn> <min> <max>
```

hvor feltene på hver linje er atskilt av blanke tegn (det er ingen andre blanke tegn enn dette). Her angir *min* og *max* henholdsvis laveste og høyeste normale verdi for denne prøven. Du skal ta utgangspunkt i følgende programskisse:

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

class LabFix {
    public static void main(String[] args) {
        Analyse ana = new Analyse();
        ana.lesLabData("Laboratorieverdier.txt");
        ana.lagKommentertPasientFil("Pasient.txt", "KommentertPasient.txt");
    }
}

class Analyse {

    // Her skal du legge inn programkode (se oppgaven)

}
```

Oppgaven din består altså i å skrive ferdig programmet på forrige side. I klassen Analyse må du deklare nødvendige objektvariabler, og du må skrive objektmetodene lesLabData (som skal lese filen "Laboratorieverdier.txt") og lagKommentertPasientFil (som skal lese filen "Pasienter.txt" og lage filen "KommentertPasient.txt" som beskrevet ovenfor). Husk å deklare disse metodene med korrekte parametre slik at kallene som foretas i main-metoden blir riktige.

Svar:

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

class LabFix {
public static void main(String[] args) {
    Analyse ana = new Analyse();
    ana.lesLabData("Laboratorieverdier.txt");
    ana.lagKommentertPasientFil("Pasient.txt",
                               "KommentertPasient.txt");
}
} // end class LabFix

class Analyse {
    // Her skal du legge inn programkode (se oppgaven)
    HashMap <String,Prove> tester =
        new HashMap <String,Prove> ();

    void lesLabData(String f) {
        In fil = new In(f);

        while (fil.hasNext()) {
            String id = fil.inWord();
            String navn = fil.inWord();
            double min = fil.inDouble();
            double max = fil.inDouble();
            Prove p = new Prove(id, navn, min,max);
            tester.put(id,p);
        }
        fil.close();
    } // end lesLabData
}
```

(svarfeltet fortsetter på neste side)

```
void lagKommentertPasientFil(String fraLege, String tilLege) {
    In fra = new In (fraLege);
    Out til = new Out(tilLege);

    til.outln(fra.inLine()); // legeID + adresse
    til.outln(fra.inLine()); // pasientID + navn

    while (fra.hasNext()) {
        String id = fra.inWord();
        double verdi = fra.inDouble();
        if (tester.containsKey(id) ) {
            Prove p = tester.get(id);
            til.out(p.id + " " + p.navn+" "+ verdi+" ");
            if (verdi > p.max)
                til.out("FOR HØY");
            else if (verdi < p.min)
                til.out("FOR LAV");
            til.outln();
        } else {
            til.outln("FEIL: fant ikke prøve med id: "+ id);
        }
    } // end while

    til.close();
    fra.close();

} // end lagKommentertPasientFil
} // end class Analyse

class Prove {
    String id,navn;
    double min,max;

    Prove (String id, String navn, double min, double max) {
        this.id = id;
        this.navn=navn;
        this.min = min;
        this.max = max;
    }
} // end class Prove
```

### Oppgave 10 (15 poeng)

Betrakt oppgaveteksten til oppgave 9 og vurder den opp mot Lov om behandling av personopplysninger.

a) Er dette systemet tillatt, og kan isåfall LabFix sette i gang å lage dette systemet etter å ha meldt fra om dette til Datatilsynet?

LabFix synes at dette systemet er fint, men vil gjerne bruke de innsamlede data til litt mer. Konkret planlegger de å lage et markedsføringssystem som blant annet skal ha to funksjoner:

b) Det skal lages en 'månedens ti-på-topp' liste over de meste brukte tester. Listen skal sendes ut til alle leger som benytter LabFix til å foreta analyser.

c) De legene som i løpet av en måned ikke bruker mer enn to av testene på 'månedens-ti-på-topp' liste tilskrives særskilt hvor de tilbys 20% rabatt neste måned på de ti testene på listen.

Vurder også b) og c) opp mot Lov om behandling av personopplysninger, og nevnt i alle dine svar hvilke paragrafer du mener kommer inn i hvert tilfelle med begrunnelse.

a) Dette systemet er tillatt, men de må søke konsesjon – ikke bare sende melding jfr. §33 (§ 2.nr 8 fordi dette klart er sensitive opplysninger : Helse-forhold). Pasientene har opplagt gitt sitt samtykke implisitt da de tok selve prøven.

b) Dette er klart ikke tillatt §8, fordi data blir brukt til et annet forhold (markedsføring) enn det pasientene (og legene) ga samtykke til. Det har blitt hevdet i noen besvarelser at siden dette ikke omhandler persondata, omfattes dette ikke av loven (Det er en god, men sannsynligvis gal tolkning da det datagrunnlaget dette tar sitt utgangspunkt i er basert på persondata)

c) Samme svar som b ) , men her er det opplagt at dette baserer seg på data innsendt av den enkelte lege, og de må klart gi sitt samtykke til slik markedsføring.

At det foreslåtte markedsføringssystemet har en rekke uheldige virkninger som at det prøver å påvirke hvilke tester legene skal ta, er leit, men dekkes ikke av (denne) loven.