Kandidatnummer:_____

Bokmål

UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Prøveeksamen 2015

Prøveeksamen i :	INF1000 — Grunnkurs i objektorientert programmering
Prøveeksamensdag :	Tirsdag 10. november 2015
Tid for eksamen :	(f eks) 10:15 til 14:15 – 4 timer
Oppgavesettet er på :	7 sider (elektronisk versjon)
Vedlegg :	Ingen
Tillatte hjelpemidler :	Alle trykte og skrevne

1. Kontroller at oppgavesettet er komplett, og les nøye gjennom oppgavene før du løser dem.
2. Du kan legge dine egne forutsetninger til grunn og gjøre rimelige antagelser, så lenge de ikke bryter med oppgavens "ånd". Gjør i så fall rede for disse forutsetningene og antagelsene.
3. Poengangivelsen øverst i hver oppgave angir maksimalt antall poeng.Sammenlagt gir alle oppgavene maksimalt 100 poeng. Unngå å bruke en stor del av tiden din på oppgaver som gir deg få poeng.
4. Svarene skal skrives på gjennomslagspapir. Skriv hardt nok til at besvarelsen blir mulig å lese på alle gjennomslagsarkene, og ikke legg andre deler av eksamensoppgaven under når du skriver.
5. Du beholder selv underste ark etter levering av de to øverste til eksamensinspektøren. Nummerer sidene, og husk å skrive kandidatnummeret ditt på besvarelsen.

Oppgave 1 (5 poeng)

a) Hva er verdien til tall etter at følgende kode er utført?

```
int tall = (3*4)-2;
tall = tall-1;
```

b) Anta at følgende programsetninger utføres. Hva skrives ut på skjermen?

```
int a = 10;
int b = 1;
for (int i = b; i<a; i=i+2) {
  b = b+i;
}
System.out.println(b);
```

c) Hva skrives ut her?

```
String serie = "0";
for (int i=5; i<10; i++) {
   serie = serie + i;
}
System.out.println ("serie = " + serie);
```

Oppgave 2 (6 poeng)

a) Gitt følgende kode. Hva returneres fra metodekallet **metode (32, 6)**?

```
int metode (int n, int m) {
  int x = 0;
  for (int i=n; i>=0; i=i-m) {
    x=i;
  }
  return x;
}
```

b) Anta at følgende program utføres:

```
class Studentregister {
   public static void main (String[] args) {
      Student s = new Student ("Ole");
      Student p = new Student ("Marit");
      System.out.println (s.faaNavn() + " og " + p.faaNavn());
   }
}

class Student {
   private String navn = ”Grete”;
   public Student (String navn) {
      this.navn = navn;
   }

   String faaNavn() {
      return navn;
   }
}
```

Hva blir utskriften på skjermen? Svar med nummer for riktig alternativ.

1. Grete og Grete
2. Ole og Johan
3. Marit og Marit
4. navn og navn
5. Ole og Marit
6. s.faaNavn() og p.faaNavn()
7. Marit og Ole
8. Ingen av alternativene over

Oppgave 3 (4 poeng)

a) Skriv binærtallet 1000 1011 som et desimaltall.
b) Skriv desimaltallet 39 som et heksadesimalt tall.
c) Skriv summen av de to binærtallene 110 og 100 som et binærtall.
d) Skriv det heksadesimale tallet 2E som et desimaltall.

Oppgave 4 (5 poeng)

Skriv ferdig metoden under. Metoden skal returnere det tallet som verken er størst eller minst av de tre tallene i parameterne a, b og c. Du kan anta at alle tallene har forskjellige verdier.

double median (double a, double b, double c) { ...}

Oppgave 5 (7 poeng)

Du skal skrive en metode med en **int**-array som parameter og som returnerer en **int**-array. Metoden skal opprette en ny **int-**array som er dobbelt så lang som den i parameteren, kopiere over verdiene i parameter-arrayen til annenhver plass (fra og med indeks 0) i den nye arrayen. De øvrige verdiene i den nye arrayen skal være 0. Til slutt skal metoden returnere den nye arrayen.

Oppgave 6 (10 poeng)

Følgende kode leser inn fra tekstfil hvor mye henholdsvis Peter og Pål har hatt i ferieutgifter. Koden kjører og gir riktig svar, men det er en del unødvendige gjentakelser. Du skal skrive en ny statisk metode i klassen **FerieUtgifter** som kan kalles for å erstatte det som er av felles funksjonalitet. Denne metoden skal kunne kalles fra metoden **main** slik at det modifiserte programmet skriver ut det samme som det opprinnelige, men med mindre gjentakelser i koden.

Merk forøvrig at formålet med oppgaven utelukkende er å vise at man behersker fornuftig introdusering av metoder, så det er ikke nødvendig vurdere eventuelle andre aspekter ved oppgaven eller koden.

```
import java.util.Scanner;
import java.io.File;

public class FerieUtgifter {
  public static void main(String[] args) throws Exception {
    String fnPeter = "Peter.txt";
    Scanner scannerPeter = new Scanner(new File(fnPeter));
    int totPeter=0;
    int utgiftPeter;

    while (scannerPeter.hasNextLine() ){
      utgiftPeter = Integer.parseInt(scannerPeter.nextLine());
      totPeter += utgiftPeter;
    }
    System.out.println("Peter har brukt: " + totPeter);

    String fnPaul = "Paul.txt";
    Scanner scannerPaul = new Scanner(new File(fnPaul));
    int totPaul=0;
    int utgiftPaul;

    while (scannerPaul.hasNextLine() ){
      utgiftPaul = Integer.parseInt(scannerPaul.nextLine());
      totPaul += utgiftPaul;
    }
    System.out.println("Paul har brukt: " + totPaul);
  }
}
```

Oppgave 7 (47 poeng)

Herr Glum lager en julekalender til barna sine hvert år. Kalenderen inneholder en gave for hver dag i desember, frem til og med julaften 24.12. Det går på omgang mellom barna hvem som får lov å åpne dagens gave – når alle har åpnet en gave hver, er det førstemann sin tur igjen. Nå ønsker herr Glum seg et program som kan hjelpe ham med holde rede på hvilke barn som får hvilke gaver, og hvor mye gavene hvert barn får, har kostet.

Programmet skal kunne lese inn data om gavene fra en fil som herr Glum oppdaterer etter hvert som han handler inn gaver i tiden før desember. Filen inneholder 48 linjer, 2 for hver gave: Én linje med navn på gaven (en tekststreng), deretter én linje med prisen (et heltall). Du skal hjelpe ham å skrive dette programmet i Java.

a) Skriv en klasse **Gave** med to variabler som forteller hva som er i gaven, og hvor mye den har kostet. Klassen skal ha en konstruktør med parametere som angir verdier for objektvariablene. Foruten konstruktøren skal klassens grensesnitt omfatte tre metoder: En som returnerer gavepris; en som returnerer gavenavn; og en metode **toString** som returnerer gavens navn og verdi som en **String**.

b) Klassen **Barn** skal ha en datarepresentasjon for barnets navn, alle gavene barnet har åpnet, og totalverdien av gaver barnet har åpnet. Grensesnittet til klassen skal være en konstruktør med barnets navn som parameter, en metode for avlesing av totalverdien av alle gaver barnet har mottatt, en metode **apneGave** som legger til en ny gave og oppdaterer totalverdien av gaver barnet har fått, og en metode **skrivBarn** som skriver ut på terminal barnets navn, en linje for hver av barnets gaver og til slutt totalverdien av barnets gaver. Skriv klassen **Barn** med alt innhold.

Vi skal i de senere oppgavene utvide programmet, som skal inneholde en klasse **Julekalender** med blant annet følgende innhold:

```
private Gave [] kalender;              // pekere til en gave for hver dag
private Barn [] apnere;                // pekere til hvert av barna i familien
private int nesteApner;                // holder rede på hvem sin tur det er til å åpne
private int dag;                       // holder rede på hvilken dag som skal åpnes neste gang
Julekalender (String[] barneNavn, String filnavn)      // Konstruktør
private void lesGavefil (String filnavn)               // leser inn gaver med pris fra fil
void nyDag ()                          // åpner en ny gave og oppdaterer datastrukturen
void gaveOversikt ();                  // Skriver en oversikt over barna, deres åpnede gaver og
                                       // totalpris per barn på skjermen
```

c) Skriv metoden **lesGavefil** i klassen **Julekalender**, som leser inn alle gavene fra filen oppgitt i parameteren og legger disse inn i kalenderen. Filformatet er beskrevet ovenfor.

d) Skriv konstruktøren til klassen **Julekalender**. Konstruktøren skal opprette objekter for alle barna i familien og opprette selve julekalenderen med gaver.

e) Skriv metoden **nyDag** i klassen **Julekalender**. Husk å oppdatere nesteApner.

f) Skriv metoden **gaveOversikt** i klassen **Julekalender**.

g) Tegn et UML klasse-diagram som viser programmets klasser med attributter og metoder, og deres relasjoner.

h) Vi skal nå utvide klassen Julekalender med historikk fra tidligere år. I første omgang er hensikten å redusere sjansen for at et barn får samme gave flere år på rad. Du kan anta at gavenavn er entydige. Klassen Juklekalender er utvidet med objektvariabel og metoder som vist nedenfor. Du skal skrive metoden **avvergetLike** som beskrevet nedenfor, og utvide metoden **nyDag** med kall på **avvergetLike.** Dersom **avvergetLike** returnerer false, skal du skrive ut en beskjed på terminalen . Beskriv eventuelle andre behov for endringer i metoder du allerede har skrevet.

```
/* HashMap historikk inneholder Gave-objekter for alle gaver som har
vært delt ut tidligere år. Nøkkel er en String bestående av navn på
barnet som fikk den, konkatenert (sammensatt) med navnet på gaven.*/

private HashMap<String,Gave> historikk = new HashMap<>();

/* Metoden lesHistorikk leser inn all historikk om tidligere utdelte
gaver fra filen Historikk.txt, og bygger opp HashMap'en historikk
som beskrevet over. Denne metoden skal du ikke skrive selv. */

void lesHistorikk(String filnavn) throws Exception {}

/* Metoden avvergetLike prøver å avverge at en barn får en gave det
har fått i tidligere år. Den kalles hver dag før metoden apneGave.
Dersom gaven og barnet som står for tur sammenfaller med noe som er
gitt tidligere år, byttes gaven som står for tur med gaven for neste
dag i kalenderen. I de tilfeller denne strategien ikke kan hindre at
et barn får en gave det har fått før, skal metoden returnere false.
Dersom barnet ikke har fått gaven som står for tur tidligere, eller
du klarer å avverge det ved å bytte, skal metoden returnere true: */
private boolean avvergetLike () {}

/* Metoden skrivHistorikk lagrer årets gaver på fil sammen med
tidligere historikk. Metoden skal ikke skrives av deg: */
void skrivHistorikk (String filnavn) {}
```

Oppgave 8 (10 poeng)

I spillet Yatzy får man poeng for ulike kombinasjoner av verdier på fem terninger. En av kombinasjonene som gir poeng kalles "hus" og krever at tre at terningene viser én verdi (er like) og at de to resterende terningene viser en annen (lik) verdi. Altså at man blant de fem terningene har tre like og to like. Det beste huset man kan ha er tre seksere og to femmere.

a) Skriv en metode
boolean besteHus(int[] t)
som tar inn en array av type int som parameter, og returnerer true dersom arrayen t består av tre verdier 6 og to verdier 5 (i vilkårlig rekkefølge). Ellers skal den returnere false. Du kan anta at du alltid får inn en array av lengde 5, der hver verdi er større eller lik 1 og mindre eller lik 6.

Altså skal følgende kode føre til at variabelen b får verdien true:

```
int[] terninger = {5,6,6,5,6};
boolean b = besteHus(terninger);
```

b) Skriv en metode

```
boolean hus(int[] t)
```

med samme parameter og returverdi som i a), men der metoden returnerer true for alle terningkombinasjoner som er hus (ikke bare hus av tre seksere og to femmere).

Oppgave 9 (6 poeng)

a) Er dette en personopplysning? Begrunn med henvisning til Personopplysningsloven.

Mann, født 1990. Mangler fast bopæl. Straffedømt for narkotikabruk.

b) Nevn tre sentrale hensyn til informasjonssikkerhet som ifølge Personopplysningsloven skal ivaretas ved behandling av personopplysninger.

c) Som ledd i arbeidet mot forsikringssvindel har norske forsikringsselskaper fått konsesjon for lagring av følgende opplysninger om sine skadeoppgjør i et felles, sentralt skaderegister: Forsikringstakers fødselsnummer, saksnummer, bransjekode, selskap, skadetype, dato, saksbehandlers initialer og skadeland.

Konsesjonen omfatter ikke tillatelse til registrering av sensitive personopplysninger.

Diskuter om noen av opplysningene konsesjonen omfatter kan komme til å omfatte sensitive personopplysninger slik at spesiell varsomhet bør utvises ved registrering av disse i det felles registeret.