```java
/*************************************************************
 * @author AnneLine
 * class Start
 *
 * Dette er en oppstartsklasse som starter klassen Main og
 * laster tidligere objekter fra klassen Filbehandling.
 *************************************************************/
public class Start
{
    public static void main(String[] args)
    {
        /* starter Main klassen og sender inn tidligere
         * student-objekter for ha disse med*/
        new Main(Filbehandling.lastGruppe());
    }
}// slutt på klassen Start
```

```java
/*************************************************************
* @author AnneLine
* Denne klassen skal først og fremst starte opp en meny
* slik at brukeren får noen valg å gjøre. Valgene brukeren
* får er å legge til student, generere grupper, endre student
* eller legge til oppgaver, eller å avslutte programmet.
* Alt etter hva brukeren velger å gjøre i menyen så vil det
* bli kalt opp en metode som så starter, enten innad i klassen
* eller i andre klasser.
*
* Det første som skjer i denne klassen er at vi initialiserer
* en variabel som kalles gruppe, og setter den til null. Dette
* gjør vi fordi vi vil at denne variabelen skal være
* tilgjengeig i hele klassen. Etter dette kaller vi opp klassen
* Gruppe og setter verdier fra den inn i variabelen gruppe.
* Når dette er gjort så vises menyen til brukeren på skjermen.
*
* Når visMeny starter får brukeren fire valg og skal så trykke
* det tallet som tilsvarer det brukeren ønsker å gjøre. Her får
* brukeren en feilmelding om den skriver et annet tall som ikke
* finnes eller om den skriver bokstaver.
* Etter at brukeren har valgt sitt tall blir det gjort en sjekk
* som sender brukeren dit den ønsker, altså kaller opp ulike
* metoder innad i klassen eller i andre klasser.
*
* I metoden leggTilStudent blir opprettet en konstruktør
* student som viser til klassen Student. I tillegg blir det
* opprettet to variabler; en boolean og en int. Før det settes
* i gang en do/while-løkke som kjører så lenge brukeren velger
* å trykke ja på spørsmål om den vil lese inn flere studenter.
* Inne i denne do/while-løkka får brokered beskjed om å legge
* til informasjon om studenter. Informasjon som skal legges
* inn er fornavn, etternavn, kjønn, studiestart og fagområde.
* Fornavn, etternavn, kjønn og studiestart er input der
* brukeren kan bruke fritekst. Dette blir sjekket for noen
* feil, men til forbedring av programmet så kunne man ha
* sjekket om det ble lagt inn stor bokstav i navnene. På
* fagområde inputen skal brukeren velge mellom tre ulike fag og
* skrive inn tallet som stemmer overens med det faget studenten
* har. Dette gjorde vi fordi vi så at det ble for omfattende
* sjekker om brukeren skulle få lov til å skrive inn fritekst.
* Til senere forbedring kan det bli sett på hvordan man skal
* gjøre disse sjekkene slik at man kan bruke informasjonen fra
* brukeren her til å dele inn i grupper senere. Etter at vi har
* fått all informasjonen fra brukeren skal vi plassere dette
```

```
46  * inn i klassen Student. Vi sender dermed en og en
47  * informasjonsbolk inn i klassen Student og til de ulike
48  * set.metodene. Når dette er gjort kaller vi på klassen Gruppe
49  * og sender inn informasjonsbolkene om studenten inn der til
50  * leggTilStudent metoden. Helt til slutt kaller vi på klassen
51  * Filbehandling og metoden lagreGruppe og sender inn gruppe-
52  * objektene til denne klassen slik at de blir lagret.
53  *
54  * I metoden lagDiverseGrupper blir det laget en do/while-løkke
55  * som kjører så lenge brukeren velger å lage nye grupper, hvis
56  * ikke så vises start menyen på nytt. Valgene brukeren får i
57  * lagDiverseGrupper er 1. Sortere etter kjønn, 2. Sortere etter
58  * fag, 3. Sortere etter studiestart, 4. Sorter alfabetisk, 5.
59  * Lage tilfeldig sammensetning, 6. Lage liste over de som kan
60  * gå opp til eksamen og 0. avslutte programmet. Alt etter hva
61  * brukeren velger å gjøre, så blir den sendt videre til til
62  * klassen GenererGruppe klassen og forskjellige metoder innad i
63  * den klassen.
64  ************************************************************/
65 // importerer ulike klasser
66 import java.util.FormatterClosedException;
69
70 public class Main
71 {
72     /*Dette er en konstruktør som initialiserer
73      * variabelen gruppe og setter denne til null
74      * i første omgang, for at denne skal være
75      * tilgjengelig i hele klassen.*/
76     private Gruppe gruppe = null;
77
78     // sender inn verdier fra klassen Gruppe
79     public Main(Gruppe grp)
80     {
81         /*Setter verdiene fra klassen Gruppe til
82          * den lokale gruppevariablen slik at verdien
83          * i gruppe blir fylt opp med objekter.*/
84         gruppe = grp;
85
86         //Viser menyen (starter programmet)
87         visMeny();
88     }
89
90     public void visMeny()
91     {
92         // gir brukeren ulike valg som kan taes
```

```
 93        String valg =JOptionPane.showInputDialog(
 94                        null,
 95                        "Skriv inn tallet p\u00E5 hva du vil gj
   \u00F8re:\n"
 96                        + "1: Legg til student.\n"
 97                        + "2: Generer gruppe.\n"
 98                        + "3: Endre student eller legg til
   oppgaver.\n"
 99                        + "0: Avslutt.");
100        /* sjekker om brukeren trykker på cancel og gir
101        mulighet til å starte på nytt*/
102        if (valg == null)
103        {
104                System.exit(0);
105        }
106        if (valg.equals("1"))// hvis brukeren skriver 1 slår denne
   til
107        {
108            // starter opp metoden aapneFil og leggTilStudent
109            leggTilStudent();
110        }
111        else if (valg.equals("2"))// hvis brukeren skriver 2 slår
   dette til
112        {
113            // starter metoden lagDiverseGrupper-GenererGruppe fra
   Ida
114            lagDiverseGrupper();
115        }
116        else if (valg.equals("3"))
117        {
118            // kaller opp vindus klassen GUIMain til Øystein
119            new GUIMain(gruppe, this);
120        }
121        else if (valg.equals("0"))// hvis brukeren skriver 0 så
   slår dette til
122        {
123            System.exit(0); // Avslutt
124        }
125        else/* sjekker om brukeren skriver inn noe feil eller
   trykker
126            ok/kryss og gir brukeren mulighet til å starte på
   nytt*/
127        {
128                int ja = JOptionPane.showOptionDialog(null, "Du
   skrev ikke riktig,"
```

```
129                         + " pr\u00F8ve igjen?",
130                         "Feil",
131                         JOptionPane.YES_NO_OPTION,
132                         JOptionPane.QUESTION_MESSAGE,
133                         null,
134                         null,
135                         null);
136                 if (ja == JOptionPane.YES_OPTION)
137                 {
138                     // start på nytt om bruker trykker ja
139                     visMeny();
140                 }
141                 else
142                 {
143                     // avslutt om bruker trykker nei eller cancel/
    kryss
144                     System.exit(0);
145                 }
146         }// slutt på else/if tester
147     }// slutt på visMeny-metode
148 
149     public void leggTilStudent()
150     {
151         // oppretter en konstruktør til klassen Student
152         Student student = null;
153         // gjør mann tilgjengelig gjennom hele metoden
154         boolean mann = false;
155         // gjør svar tilgjengelig gjennom hele metoden
156         int svar;
157 
158         // starter en løkke der brukeren kan skrive inn input
159         do
160         {
161             // kaller opp Studentklassen
162             student = new Student();
163             // fornavn fra brukeren
164             String elevFornavn = JOptionPane.showInputDialog(null,
165                     "Skriv inn fornavnet p\u00E5 eleven:",
166                     "Fornavn",
167                     JOptionPane.QUESTION_MESSAGE);
168             /* sjekker om brukeren trykker cancel eller kryss
169             og avslutter programmet*/
170             if (elevFornavn == null || elevFornavn == "")
171             {
172                 System.exit(0);
```

```
173             }
174             // hvis brukeren lar input stå tomt får sjans på nytt
175             if (elevFornavn.isEmpty())
176             {
177                 do
178                 {
179                     elevFornavn = JOptionPane.showInputDialog(null,
180                             "M\u00E5 ha et fornavn, skriv inn p
    \u00E5 nytt:");
181                     // avslutt om det blir trykket cancel/kryss
182                     if (elevFornavn == null || elevFornavn == "")
183                     {
184                         System.exit(0);
185                     }
186                 }while (elevFornavn.isEmpty());// fortsett så lenge
    input er tom
187             }
188             // input fra brukeren
189             String elevEtternavn =
    JOptionPane.showInputDialog(null,
190                     "Skriv inn etternavn p\u00E5 eleven:",
191                     "Etternavn",
192                     JOptionPane.QUESTION_MESSAGE);
193             // sjekker om brukeren trykker cancel eller kryss
    avslutt program
194             if (elevEtternavn == null || elevEtternavn == "")
195             {
196                 System.exit(0);
197             }
198             // hvis brukeren lar input stå tomt får sjans på nytt
199             if (elevEtternavn.isEmpty())
200             {
201                 do
202                 {
203                     elevEtternavn =
    JOptionPane.showInputDialog(null,
204                             "M\u00E5 ha et etternavn, skriv inn p
    \u00E5 nytt:");
205                     // sjekker om bruker trykker cancel/kryss og
    avslutter
206                     if (elevEtternavn == null || elevEtternavn ==
    "")
207                     {
208                         System.exit(0);
209                     }
```

```
210                 }while (elevFornavn.isEmpty());// fortsett så lenge
    input er tom
211             }
212             // input fra brukeren
213             String elevKjonn = JOptionPane.showInputDialog(null,
214                     "Skriv inn hvilket kj\u00F8nn, mann eller
    dame?",
215                     "Kj\u00F8nn",
216                     JOptionPane.QUESTION_MESSAGE);
217             // sjekker om bruker trykker cancel/kryss og avslutter
218             if (elevKjonn == null || elevKjonn == "")
219             {
220                 System.exit(0);
221             }
222             // hvis brukeren lar input stå tom, gi en ny sjans
223             if (elevKjonn.isEmpty())
224             {
225                 do
226                 {
227                     elevKjonn = JOptionPane.showInputDialog(null,
228                             "M\u00E5 ha et kj\u00F8nn, skriv inn p
    \u00E5 nytt:");
229                     // hvis bruker trykker cancel/kryss avslutt
230                     if (elevKjonn == null || elevKjonn == "")
231                     {
232                         System.exit(0);
233                     }
234                 }while (elevKjonn.isEmpty());// fortsett så lenge
    input er tom
235             }
236             // sjekker om brukeren har skrevet inn mann
237             if (elevKjonn.toLowerCase().equals("mann"))
238                 mann = true;// setter da mann til true
239             // sjekker om brukeren har skrevet inn noe annet enn
    mann
240             else
241                 mann = false;// setter da mann til false
242             // input fra brukeren
243             String studieStart = JOptionPane.showInputDialog(null,
244                     "Hvilket \u00E5r startet studenten?",
245                     "Studiestart",
246                     JOptionPane.QUESTION_MESSAGE);
247             // hvis brukeren trykker cancel skriv ut beskjed og ny
    input
248             if (studieStart == null || studieStart == "")
```

```
249             {
250                 System.exit(0);
251             }
252             // hvis brukeren lar input stå tom start på nytt
253             if (studieStart.isEmpty())
254             {
255                 do
256                 {
257                     studieStart = JOptionPane.showInputDialog(null,
258                             "M\u00E5 ha et studie\u00E5r, skriv inn
    p\u00E5 nytt:");
259                     if (studieStart == null || studieStart == "")
260                     {
261                         System.exit(0);
262                     }
263                 }while (studieStart.isEmpty());// fortsett så lenge
    input er tom
264             }
265             int studStart = Integer.parseInt(studieStart);// parse
    til integer
266             // hvis input er mindre enn 2010 eller større enn 2014
267             if (studStart < 2010 || studStart > 2014)
268             {
269                 do
270                 {
271                     studieStart = JOptionPane.showInputDialog(null,
272                             "Studenter som g\u00E5r p\u00E5 dette "
273                             + "studiet har startet mellom 2010 og
    2014, "
274                             + "skriv inn \u00E5rstall p\u00E5
    nytt.");
275                     // hvis bruker trykker cancel/kryss avslutt
276                     if (studieStart == null || studieStart == "")
277                     {
278                         System.exit(0);
279                     }
280                     studStart = Integer.parseInt(studieStart);
281                 }while (studStart < 2010 || studStart > 2014);/*
    fortsett
282                 så lenge input er mindre enn 2010 eller større enn
    2014*/
283             }
284             // input fra brukeren
285             String fagomrade = JOptionPane.showInputDialog(
286                     "Velg nr. for faget til studenten:\n"
```

```
287                     + "1. Norsk\n"
288                     + "2. Engelsk\n"
289                     + "3. Matematikk");
290             // hvis bruker trykker cancel/kryss avslutt
291             if (fagomrade == null || fagomrade == "")
292             {
293                 System.exit(0);
294             }
295             // hvis input står tom, gi en ny sjans
296             if (fagomrade.isEmpty())
297             {
298                 do
299                 {
300                     fagomrade = JOptionPane.showInputDialog(
301                             "M\u00E5 ha et fag, pr\u00F8v p\u00E5
    nytt.\n"
302                             + "1. Norsk\n"
303                             + "2. Engelsk\n"
304                             + "3. Matematikk");
305                     if (fagomrade == null || fagomrade =="")// hvis
    kryss/cancel avslutt
306                     {
307                         System.exit(0);
308                     }
309                 }while (fagomrade.isEmpty());// fortsett så lenge
    input er tom
310             }
311             int fag = Integer.parseInt(fagomrade);// parse til
    integer
312             // hvis bruker har tatt et tall utenfor intervallet,
    start på nytt
313             if (fag < 1 || fag > 3)
314             {
315                 do
316                 {
317                 JOptionPane.showMessageDialog(null,
318                         "Feil nr, pr\u00F8v igjen.");
319                 fagomrade = JOptionPane.showInputDialog(
320                         "Velg nr. for faget til studenten:\n"
321                         + "1. Norsk\n"
322                         + "2. Engelsk\n"
323                         + "3. Matematikk");
324                 if (fagomrade == null || fagomrade == "")// hvis
    kryss/cancel, avslutt
325                 {
```

```
326                     System.exit(0);
327                 }
328                 fag = Integer.parseInt(fagomrade);
329                 }while (fag < 1 || fag > 3);/* fortsett så
330                 lenge tallet er utenfor intervallet*/
331             }
332             // slutt på input fra brukeren
333             // start å sende input fra brukeren til klassen Student
    for senere arbeid
334             try
335             {
336                 student.setFornavn(elevFornavn);//
337                 student.setEtternavn(elevEtternavn);
338                 student.setKjonn(mann);
339                 student.setStudiestart(studieStart);
340                 if (fagomrade.equals("1"))
341                 {
342                     // her bestemmes det hvilket fag studenten har
343                     student.setFag("Norsk");
344                 }
345                 else if (fagomrade.equals("2"))
346                 {
347                     // her bestemmes det hvilket fag studenten har
348                     student.setFag("Engelsk");
349                 }
350                 else if (fagomrade.equals("3"))
351                 {
352                     // her bestemmes det hvilket fag studenten har
353                     student.setFag("Matematikk");
354                 }
355                 // sender studentene til klassen Gruppe for å gjøre
    til objekter
356                 gruppe.leggTilStudent(student);
357                 /* etter hver gang leggTilStudent har kjørt så
358                 sendes de inn i filbehandlings-klassen og lagres i
    objekter der*/
359                 Filbehandling.lagreGruppe(gruppe);
360             }
361             catch (FormatterClosedException fce)// hvis try ikke
    slår til - feilmelding
362             {
363                 JOptionPane.showMessageDialog(
364                         null,
365                         "",
366                         "Feil ved skriving til fil",
```

```
367                         JOptionPane.PLAIN_MESSAGE);
368                 return;
369             }
370             catch (NoSuchElementException ee)// hvis try ikke slår til - feilmelding
371             {
372                 JOptionPane.showMessageDialog(
373                         null,
374                         "",
375                         "Feil input, pr\u00F8v igjen",
376                         JOptionPane.PLAIN_MESSAGE);// feilmelding
377             }
378             // brukeren får valg om å starte innlesing av flere elever på nytt
379             svar = JOptionPane.showOptionDialog(null,
380                     "Vil du lese inn flere elever?",
381                     "Klasseliste",
382                     JOptionPane.YES_NO_OPTION,
383                     JOptionPane.QUESTION_MESSAGE,
384                     null,
385                     null,
386                     null);
387             if (svar == JOptionPane.NO_OPTION)
388             {
389                 visMeny();
390             }
391         }while (svar == JOptionPane.YES_OPTION);
392         System.exit(0);
393     }// slutt på metoden leggTilStudent
394 
395     public void lagDiverseGrupper()
396     {
397         //lokal variabel
398         int igjen;
399         // start en løkke for å finne ut hvilke grupper brukeren vil generere
400         do
401         {
402             // hvis et valg-tre som sender brukeren videre på riktig gren
403             String valgtre = JOptionPane.showInputDialog(null,
404                                 "Skriv inn tallet p\u00E5 hva du vil gj\u00F8re:\n"
405                                 + "1: Sorter etter kj\u00F8nn.\n"
406                                 + "2: Sorter etter fag.\n"
```

```
407                             + "3: Sorter etter studiestart.\n"
408                             + "4: Sorter alfabetisk.\n"
409                             + "5: Lag tilfeldig sammensetting.\n"
410                             + "6: Lag liste over de som er godkjent
    til eksamen.\n"
411                             + "0: Avslutt.");
412             // kaller opp og konstruerer en konstruktør for klassen
    GenererGruppe
413             GenererGruppe kjor = new GenererGruppe(gruppe);
414             // sortere etter kjønn
415             if (valgtre == null){ //Brukeren lukker dialogen.
416                 visMeny(); //GÅ tilbake til hovedmeny
417             }
418 
419             if (valgtre.equals("1"))
420             {
421                 // start metoden sortereKjonn i klassen
    GenererGruppe (Ida)
422                 kjor.sortereKjonn();
423             }
424             // sortere etter fag
425             else if (valgtre.equals("2"))
426             {
427                 // start metoden sortereFag i klassen GenererGruppe
    (Ida)
428                 kjor.sortereFag();
429             }
430             // sortere etter studiestartår
431             else if (valgtre.equals("3"))
432             {
433                 // start metoden sortereStudiestart i klassen
    GenererGruppe (Ida)
434                 kjor.sortereStudiestart();
435             }
436             // sorterer alfabetisk
437             else if (valgtre.equals("4"))
438             {
439                 /* start metoden sortereAlfabetisk i klassen
    GenererGruppe (Ida),
440                 sender inn studenter fra klassen Gruppe*/
441 
    kjor.sortereAlfabetisk(gruppe.hentStudenterAsArray());
442             }
443             // sett sammen tilfeldig
444             else if (valgtre.equals("5"))
```

```
445             {
446                 /* start metoden sortereTilfeldig i klassen
    GenererGruppe (Ida),
447                 sender inn studenter fra klassen Gruppe*/
448 
    kjor.sortereTilfeldig(gruppe.hentStudenterAsArray());
449             }
450             // sorterer etter de som kan gå til eksamen
451             else if (valgtre.equals("6"))
452             {
453                 // start metoden sortereGodkjent i klassen
    GenererGruppe (Ida)
454                 kjor.sortereGodkjent();
455             }
456             // avslutter
457             else if (valgtre.equals("0"))
458             {
459                 System.exit(0);
460             }
461             // hvis det er skrevet feil tall eller en bokstav så
    får bruker ny sjans
462             else
463             {
464                 igjen = JOptionPane.showOptionDialog(null,
465                         "Du skrev inn et valg som ikke finnes, pr
    \u00F8ve p\u00E5 nytt?",
466                         "Lister",
467                         JOptionPane.YES_NO_OPTION,
468                         JOptionPane.QUESTION_MESSAGE,
469                         null,
470                         null,
471                         null);
472                 // hvis brukeren velger ja, start gruppevalg igjen
473                 if (igjen == JOptionPane.YES_OPTION)
474                     lagDiverseGrupper();
475                 // hvis brukeren gjør noe annet start hele
    programmet på nytt
476                 else
477                 {
478                     visMeny();
479                 }
480             }
481             // brukeren får valg om å lage nye lister
482             igjen = JOptionPane.showOptionDialog(null,
483                     "Vil du generere nye lister?",
```

```
484                     "Lister",
485                     JOptionPane.YES_NO_OPTION,
486                     JOptionPane.QUESTION_MESSAGE,
487                     null,
488                     null,
489                     null);
490             // hvis brukeren trykker nei start hele programmet på
    nytt
491             if (igjen == JOptionPane.NO_OPTION)
492             {
493                 visMeny();
494             }
495         }while(igjen == JOptionPane.YES_OPTION);/* fortsett løkke
    med å
496         generere grupper så lenge igjen er ja*/
497         System.exit(0);
498     }// slutt på metoden lagDiverseGrupper
499 }// slutt på klassen Main
500
```

```java
/**********************************************************************
 * @author Øystein
 * Student.class
 *
 * Klassen representerer en student som en del av en gruppe i
 * Studentadministrasjonsprogrammet.
 *
 * Klassen skal inneholde, sette og svare på
 * 1.Studentens navn
 * 2.Studentens studiestart
 * 3.Studentens kjønn
 * 4.Studentens oppgaver (dypere informasjon om oppgavene lagres
 * i et eget oppgave-objekt).
 * 5.Studentens status i forhold til om alle oppgaver er godkjente, og
 * studenten er klar til eksamen.
 * 6.Studentnummer (vi tar ikke denne i bruk, men legger den ved da vi
 * kan se en fremtidig nytte av dette; skille studenter med likt navn og
 * samkjøring mot andre skolesystemer).
 * 7.Studentens fagområde
 *
 **********************************************************************/
import java.io.Serializable;

public class Student implements Serializable{

    //Studentens fornavn
    private String fornavn = "";
    //Studentens etternavn
    private String etternavn = "";
    //Studentens fagområde (f.eks. Matematikk)
    private String fagomrade ="";
    /*Variabel for kjønn; true = mann,
    false = kvinne (tilfeldig valgt, trenger ikke tolkes... :)*/
    private boolean mann = true;
    //Studentens studiestart f.eks. 2014
    private int studiestart = 2014;
    //Studentnummer f.eks. s123456789
    private String studentnummer = "";
    //En vektor som inneholder alle studentens oppgaver.
```

```
44      private Vector<Oppgave> oppgaver = new Vector();
45
46
47      /*En tom konstruktør i tilfellet man trenger å
48      opprette et studentobjekt før en har data til å fylle det.*/
49      public Student(){
50
51      }
52
53      //Konstruktør for å opprette en ny student med alle verdier
    satt.
54      public Student(String forNavn,
55              String etterNavn,
56              int studieStart,
57              boolean kjonnMann,
58              String fagOmrade,
59              Vector<Oppgave> oppgaver){
60          this.fornavn = forNavn;
61          this.etternavn = etterNavn;
62          this.studiestart = studieStart;
63          this.mann = kjonnMann;
64          this.fagomrade = fagOmrade;
65          this.oppgaver = oppgaver;
66      }
67
68
69      //Returnerer studentens fornavn
70      public String getFornavn(){
71          return fornavn;
72      }
73
74      //Setter eller endrer studentens fornavn
75      public void setFornavn(String forNavn){
76          this.fornavn = forNavn;
77      }
78
79      //Returnerer studentens etternavn
80      public String getEtternavn(){
81          return etternavn;
82      }
83
84      //Setter eller endrer studentens etternavn
85      public void setEtternavn(String etterNavn){
86          this.etternavn = etterNavn;
87      }
```

```
 88
 89
 90     /*Returnerer studentens hele navn dersom metoden blir kalt.
 91     Dette skjer f.eks. ved at man System.out.println
    studentobjektet,
 92     eller ved at man legger objektet i en liste. Med andre ord,
 93     denne metoden bestemmer hvordan objektet vises i en liste.*/
 94     public String toString(){
 95         return fornavn + " " + etternavn;
 96     }
 97
 98     //Setter eller endrer studentens fagområde
 99     public void setFag(String fag){
100         this.fagomrade = fag;
101     }
102
103     //Returnerer studentens fagområdet
104     public String getFag(){
105         return fagomrade;
106     }
107
108     //Setter eller endrer studentens studestart
109     public void setStudiestart(String studieStart){
110         try{
111             this.studiestart = Integer.parseInt(studieStart);
112         }catch (Exception e){
113             System.out.println(
114                     "Feil ved lesing av studestart. "
115                     + "Setter studiestart til gjeldene \u00E5r!" +
    e);
116             //Ved feil, sett årstall til innenvÊrende år
117             GregorianCalendar gc = new GregorianCalendar();
118             this.studiestart = gc.get( Calendar.YEAR);
119         }
120     }
121
122     //Returnerer studentens studiestart
123     public int getStudiestart(){
124         return studiestart;
125     }
126
127     //Returnerer true hvis mann, false hvis dame.
128     public boolean isMann(){
129         return mann;
130     }
```

```
131
132     /*Returnerer true hvis dame, false hvis mann
133     (denne metoden er laget i likestillingens navn!)*/
134     public boolean isDame(){
135         return !mann;
136     }
137
138     //Returnerer kjønnet som en streng.
139     public String getKjonn(){
140         if (mann) return "Mann"; else return "Kvinne";
141     }
142
143     //Setter eller endrer studentens kjønn
144     public void setKjonn(boolean mann){
145         this.mann = mann;
146     }
147
148     //Setter eller endrer studentens fornavn
149     public void setStudentnummer(String studentnummer){
150         this.studentnummer = studentnummer;
151     }
152
153     //Returnerer studentnummer
154     public String getStudentnummer(){
155         return studentnummer;
156     }
157
158     //Legger til en oppgave (et oppgaveobjekt) i studentens
    oppgaveliste.
159     public void addOppgave(Oppgave opg){
160         oppgaver.addElement(opg);
161     }
162
163     //Returnerer alle studentens oppgaver som en array av
    oppgaveobjekter.
164     public Oppgave[] getOppgaver(){ //Returnerer alle oppgavene som
    en array.
165         Oppgave[] opg = new Oppgave[oppgaver.size()];
166         oppgaver.toArray(opg);
167         return opg;
168     }
169
170     //Returnerer alle studentens oppgaver som en Vector
171     public Vector<Oppgave> getOppgaverAsVector(){
172         return oppgaver;
```

```
173     }
174 
175     /*Kontrollerer hver eneste oppgave studenten har, og
176     returnerer true hvis alle er godkjente (altså at studenten kan
    ta eksamen)
177     og false hvis ikke.*/
178     public boolean isGodkjent(){
179 
180         /*Gå gjennom alle oppgaver, dersom en ikke er godkjent:
    returner
181         false eller returner true*/
182         for (int i = 0; i < oppgaver.size(); i++){
183             if (oppgaver.elementAt(i).isGodkjent())
184                 continue;
185             else
186                 return false;
187 
188         }
189 
190         //returner FALSE dersom studenten ennå ikke har gjort noen
    oppgaver.
191         if (oppgaver.size() < 1)
192             return false;
193 
194         return true;
195     }
196 }
197 
```

```
 2  * @author Øystein
16 import java.io.Serializable;
18
19 //Serializable for enkelt lagring.
20 public class Gruppe implements Serializable
21 {
22     //Vector til å oppbevare alle studentobjektene (Vector ettersom
   denne er dynamisk).
23     private Vector<Student> studenter = new Vector<Student>();
24
25     public Gruppe() //Tom konstruktør.
26     {
27
28     }
29
30     public void leggTilStudent(Student stud) //Legg til et
   studentobjekt.
31     {
32         studenter.addElement(stud);
33     }
34
35     public Student[] hentStudenterAsArray() //Returnerer studentene
   som en array.
36     {
37         Student[] stud = new Student[studenter.size()];
38         studenter.toArray(stud);
39         return stud;
40     }
41
42     //Returnerer vectoren med alle studentene.
43     public Vector<Student> hentStudenterAsVector()
44     {
45         return studenter;
46     }
47 }
```

```java
/**********************************************************************
 * @author Øystein
 * Filebehandling.class
 *
 * Klassen lagrer og leser et gruppeobjekt med studenter til fil med
 * navn "gruppe.obj" ved å bruke Java sin serializable funksjon.
 **********************************************************************/
import java.io.BufferedInputStream;

public class Filbehandling
{

    /*Filnavnet til gruppefila som blir lagret på harddisken
      (statisk ettersom det ikke skal være flere tilfeller av denne.
       final slik at den er fast og ikke endres).
     Dette gjør det lett å endre filnavnet, uten å måtte gå gjennom hele koden.*/
    public static final String FILNAVN = "gruppe.obj";


    /*Metode for å laste gruppeobjektet, returnerer et Gruppe-objekt.
    OBS! Hvis feil ved lagring, anta at filen er ødelagt eller ikke
    eksisterer. Lag en tomt nytt gruppeobjekt! (Dette er ikke noe
    produksjonsverdig løsning, men i forhold til kriteriene i dette
    prosjektet så holder det).*/
    public static Gruppe lastGruppe()
    {
        File f = new File(Filbehandling.FILNAVN);
        Gruppe gruppe = null;

        if (f.exists())
        {
            /*Filbehandling kjøres i en try-klamme, slik at feil kan
              bli behandlet i stedet for å kræsje programmet.*/
            try
            {
                /*Opprette en strøm fra filen, en buffer for stabil lesing,
```

```
45                  og en strøm for å ta imot objektet som blir lest.*/
46                  FileInputStream fis = new
  FileInputStream(Filbehandling.FILNAVN);
47                  BufferedInputStream bis = new
  BufferedInputStream(fis);
48                  ObjectInputStream ois = new ObjectInputStream(bis);

49
50                  //Leser objektet fra filen
51                  Object obj = ois.readObject();
52
53                  //Kontrollerer om objektet er av typen "Gruppe"
54                  if (obj instanceof Gruppe)
55                  {
56                      gruppe = (Gruppe)obj;
57                  }
58
59                  //Lukker filen.
60                  ois.close();
61                  bis.close();
62                  fis.close();
63              }
64              catch (ClassNotFoundException cnf)
65              {
66                  /*Datafilen inneholder ikke klassen gruppe.
67                  Start programmet med en ny, tom studentgruppe.*/
68                  System.out.println(
69                          "Fant ikke klassen ved lasting \n " +
70                          cnf);
71                  gruppe = new Gruppe();
72              }
73              catch (InvalidClassException ice)
74              {
75                  /*Progammet er ikke lenger kompatibelt med lagret
  data.
76                  Start programmet med en ny, tom studentgruppe.*/
77                  System.out.println(
78                          "Endringer siden sist! Ikke kompatibel med
  gammel brukerdata.\n"
79                          + "Lager ny brukerdata.");
80                  gruppe = new Gruppe();
81              }
82              catch (IOException ioe)
83              {
84                  /*Generelt problem med innlastning.
```

```
85                  Start programmet med en ny, tom studentgruppe.*/
86                  System.out.println("Feil ved lasting \n" + ioe);
87                  gruppe = new Gruppe();
88              }
89          }
90          else
91          {
92              /*Det eksisterer ikke en fil med gruppa.
93               Start programmet med en tom gruppe.*/
94              gruppe = new Gruppe();
95          }
96
97
98          //SKRIVER UT EN LISTE OVER ALLE LAGREDE STUDENTER HVER GANG
    FILEN ≈PNES.
99          //Mest nyttig under utvikling, bare for å se at alt
    stemmer :)
100         for (int i = 0; i < gruppe.hentStudenterAsVector().size();
    i++){
101
    System.out.println( gruppe.hentStudenterAsVector().elementAt(i));
102         }
103
104         return gruppe;
105     }
106
107
108     //Metode for å lagre et Gruppe-objekt, tar i mot et
    gruppeobjekt.
109     public static void lagreGruppe(Gruppe grp)
110     {
111
112         /*Filbehandling kjøres i en try-klamme, slik at feil kan
    bli
113         behandlet i stedet for å kræsje programmet.*/
114         try
115         {
116             /*Opprette en strøm til filen, en buffer for stabil
    skriving,
117             og en strøm for å sende objektet til filen.*/
118             FileOutputStream fos = new
    FileOutputStream(Filbehandling.FILNAVN);
119             BufferedOutputStream bos = new
    BufferedOutputStream(fos);
120             ObjectOutputStream oos = new ObjectOutputStream(bos);
```

```
121             oos.writeObject(grp);
122 
123             //Tømming av alle strømmer (for å hindre delevis
    lagring og korrupte filer.
124             oos.flush();
125             oos.close();
126             bos.flush();
127             bos.close();
128             fos.flush();
129             fos.close();
130         }
131         catch (IOException ioe)
132         {
133             /*Gir brukeren beskjed om at lagringen ikke var
    vellykket,
134             i tillegg vises java sin feilmelding.*/
135             System.out.println("Feil ved lagring! " + ioe);
136         }
137     }
138 }
```

```java
/**********************************************************************
 * @author Anders Lind Johnsen
 *
 *
 * Beskrivelse
 *
 * Klassen oppgave er å teste om studentene har fått sine oppgaver og
 * arbeidskrav godkjent, og følgelig «godkjenner» om de kan gå opp til eksamen.
 *
 *
 * Klassen inneholder informasjon om en enkelt oppgave studenten har gjort.
 * Dato for når oppgaven ble levert, om det er gitt tilbakemelding, om den er
 * godkjent.
 *
 *
 * Klassen kalles opp/får sin input fra Student
 *
 *
 **********************************************************************/

import java.io.Serializable;

public class Oppgave implements Serializable{

    /*variabler levertdato (når oppgaven ble levert), fått
    tilbakemelding (er det gitt tilbakemelding),
    godkjent (er oppgaven godkjent).*/
    private String tilbakeMelding = " ";
    private boolean godkjent = false;
     //Valg string for dato for å gjøre det enklere å håndtere
    private String levertDato = " ";
    private String oppgaveNavn = " ";

    public Oppgave(){ //Tom konstruktør

```

```
30      }
31
32      public Oppgave(String oppgaveNavn,
33              boolean godkjent,
34              String levertDato,
35              String tilbakeMelding){ //konstuktør som samtidig tar i
  mot alle variablene
36          this.godkjent = godkjent;
37          this.levertDato = levertDato;
38          this.tilbakeMelding = tilbakeMelding;
39          this.oppgaveNavn = oppgaveNavn;
40      }
41
42      // setter oppgavenavn
43      public void setOppgaveNavn(String oppgaveNavn){
44          this.oppgaveNavn = oppgaveNavn;
45      }
46
47      // returnerer oppgavenavn
48      public String getOppgaveNavn(){
49          return oppgaveNavn;
50      }
51
52      public String toString(){
53          return oppgaveNavn;
54      }
55
56      //setter dato for oppgavelevert
57      public void setLevertDato(String levertDato){
58          this.levertDato = levertDato;
59      }
60
61      //returnerer dato for oppgavelevert
62      public String getLevertDato(){
63          return levertDato;
64      }
65      /*Returnerer true hvis det er gitt tilbakemelding
66      (hvis tilbakeMelding string inneholder tekst).*/
67      public boolean isTilbakeMeldingOk(){
68          if (tilbakeMelding == null){
69              return false;
70          }else if (tilbakeMelding.isEmpty()){
71              return false;
72          }
73
```

```
74          return true;
75      }
76
77      //Lar deg sette tilbakemeldingsteksten.
78      public void settTilbakeMelding(String tilbakeMelding){
79          this.tilbakeMelding = tilbakeMelding;
80      }
81
82      // Lar deg sette om det er gitt tilbakemelding
83      public String getTilbakeMelding(){
84          return tilbakeMelding;
85      }
86
87      //Lar deg sette om oppgaven er godkjent eller ei
88      public void setGodkjent(boolean godkjent){
89          this.godkjent = godkjent;
90      }
91
92      //returnerer true hvis oppgaven er godkjent.
93      public boolean isGodkjent(){
94          return godkjent;
95      }
96  }
```

```
 2  * Dialog.class
10 import java.awt.BorderLayout;
17
18 public class Dialog extends JDialog implements ActionListener{
19
20     /*Nesten alle vinduer vil ha en lukk knapp, så jeg legger denne
21     til i Dialog-klassen de andre vindusklassene vil arve fra.*/
22     protected JButton btnLukk = new JButton("Lukk");
23
24     protected void setup(){ //Setter standardverider for vinduene
   jeg vil bruke
25         this.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
26         this.setLayout( new BorderLayout());
27
28         btnLukk.addActionListener(this);
29     }
30
31
32     protected void centerScreen() {//Sentrerer vinduet midt på
   skjermen.
33           Dimension dim = getToolkit().getScreenSize();
34           Rectangle abounds = getBounds();
35           setLocation((dim.width - abounds.width) / 2,
36               (dim.height - abounds.height) / 2);
37     }
38
39
40     public void actionPerformed(ActionEvent ae) {//Lytter etter
   tastetrykk
41         if (ae.getSource() == btnLukk){
42             setVisible(false);
43         }
44     }
45
46 }
47
```

```java
/**********************************************************************
 * @author Ida
 *
 * Klasse for å generere ulike gruppesammensetninger av studentgruppen.
 * Klassen kalles opp av Main, og bruker får velge hvilke gruppesammensetning
 * som ønskes.
 * Valgene er gruppering på bakgrunn av kjønn, fag, kull (år for studiestart),
 * alfabetisk liste,
 * tilfeldig gruppering basert på et ønsket antall grupper og studenter
 * kvalifisert til eksamen på bakgrunn av leverte og godkjente arbeidskrav.
 * Grupperingene lagres ikke på noen fil da tanken er at programmet skal
 * brukes hyppig som administrasjonsverktøy.
 * Samtidig vil dette sikre at det hele tiden genereres grupper ut fra oppdatert
 * studentinformasjon,
 * hvilket er avgjørende i blant annet "sortereGodkjent"-gruppen.
 *
 **********************************************************************/
import javax.swing.*;

public class GenererGruppe
{
    //LAGER EN ARRAY FOR ≈ HOLDE P≈ ALLE STUDENTENE
    private static Student[] stud = null;
    static String utskrift = ""; /* utskrift-variabel som får
    verdi basert på valg gruppegenerering. Når verdi er satt skrives
    dette ut i et dialogvindu i slutten av valgt metode. */

    /*KONTRUKTÿR SOM FOR ≈ LAGE EN GENERERGRUPPE-OBJEKT SOM TAR I MOT
    ALLE STUDENTENE OG PLASSERER DISSE I ARRAYEN "stud".*/
    public GenererGruppe(Gruppe grp)
    {
        stud = grp.hentStudenterAsArray();
    }
```

```
37
38      public void sortereKjonn() // start metode sortere kjønn
39      {
40          /* StringBuffer er en type String-variabel som tillater
41          kontinuerlig lagring av ny informasjon*/
42          StringBuffer tekstomradeMann = new StringBuffer();
43          StringBuffer tekstomradeDame = new StringBuffer();
44          StringBuffer tekstomradeSamlet = new StringBuffer("MANN
    \n"); // overskrift Mann
45
46          for (int i = 0; i < stud.length; i++)
47          {
48              if(stud[i].isMann() == true) //DENNE RETURNERER TRUE
    ELLER FALSE
49              {
50                  tekstomradeMann.append(stud[i].getFornavn()
51                          + " "
52                          +stud[i].getEtternavn()
53                          + "\n");
54              }
55              else /* alt annet enn mann, settes i dette programmet
    likt dame.
56                  Vi har med andre ord valgt å ikke ta hensyn til
    eventuelle
57                  studenter som ikke identifiserer seg som mann eller
    kvinne
58                  (og tar i aller høyeste grad kritikk for dette).*/
59              {
60                  tekstomradeDame.append(stud[i].getFornavn()
61                          + " "
62                          + stud[i].getEtternavn()
63                          + "\n");
64              }
65          }// slutt for-løkke
66          // samlet utskrift av mannfolka
67          tekstomradeSamlet.append( tekstomradeMann);
68          // overskrift Dame med linjeskift både før og etter
69          tekstomradeSamlet.append("\n\nDAME\n");
70          // samlet utskrift av kvinnfolka (og eventuelt andre ikke-
    menn)
71          tekstomradeSamlet.append( tekstomradeDame);
72
73          utskrift = tekstomradeSamlet.toString();
74          JOptionPane.showMessageDialog(null, utskrift);
75      } // slutt metode sortereKjønn
```

```
76
77      // start metode sortere fag
78      public void sortereFag ()
79      {
80          // her tar vi hØyde for at det kun er inntak om hØsten
81          StringBuffer tekstomrade1 = new StringBuffer("NORSK\n");
82          StringBuffer tekstomrade2 = new StringBuffer("ENGELSK\n");
83          StringBuffer tekstomrade3 = new StringBuffer("MATEMATIKK
    \n");
84
85          int fag =
    Integer.parseInt(JOptionPane.showInputDialog(null,
86                  "Skriv inn tallet p¬ hvilken fagklasse du vil se\n"
87                  + "1: Norsk \n"
88                  + "2: Engelsk \n"
89                  + "3: Matematikk \n"));
90          /* IF-SETNINGER SOM STARTER SORTERING AV FAGKLASSER. FýRSTE
91          IF-SETNING KOMMENTERT, GANGEN ER LIK I ELSE-IF-
    SETNINGENE.*/
92          if (fag == 1)
93          {
94              // for-løkke som kjører gjennom hele arrayens lengde
95              for (int i = 0; i < stud.length; i++)
96              {
97                  // plukker ut studenter med faget Norsk.
98                  if (stud[i].getFag().toLowerCase().equals("norsk"))
99                  {
100                     /* skriver ut studentenes fornavn og
101                     etternavn etterhvert som de identifiseres som
    norskstudenter.*/
102                     tekstomrade1.append(stud[i].getFornavn()
103                             + " " + stud[i].getEtternavn());
104                     tekstomrade1.append("\n"); // Linjeskift legges
    til
105                 }
106             }
107             // utskrift av utfyllt tekstområde med ferdigsortert
    fagklasse
108             utskrift = tekstomrade1 + "";
109         }
110         else if (fag == 2)
111         {
112             for (int i = 0; i < stud.length; i++)
113             {
114                 // plukker ut studenter med faget Engelsk.
```

```
115                 if
    (stud[i].getFag().toLowerCase().equals("engelsk"))
116                 {
117                     tekstomrade2.append(stud[i].getFornavn()
118                             + " " + stud[i].getEtternavn());
119                     tekstomrade2.append("\n");
120                 }
121             }
122             utskrift = tekstomrade2 + "";
123         }
124         else if (fag == 3)
125         {
126             for (int i = 0; i < stud.length; i++)
127             {
128                 // plukker ut studenter med faget Matematikk.
129                 if
    (stud[i].getFag().toLowerCase().equals("matematikk"))
130                 {
131                     tekstomrade3.append(stud[i].getFornavn()
132                             + " " + stud[i].getEtternavn());
133                     tekstomrade3.append("\n");
134                 }
135             }
136             utskrift = tekstomrade3 + "";
137         } // slutt if-løkke
138         JOptionPane.showMessageDialog(null, utskrift);
139     } // slutt metode sortereFag
140 
141     /* start metode sortere studiestart. Her tar vi høyde for at
    det
142     kun er inntak om høsten*/
143     public void sortereStudiestart()
144     {
145         //Student [] stud = grp.hentStudenterAsArray();
146 
147         // her tar vi hØyde for at det kun er inntak om hØsten
148         StringBuffer tekstomrade1 = new StringBuffer("Høst
    2010\n");
149         StringBuffer tekstomrade2 = new StringBuffer("Høst
    2011\n");
150         StringBuffer tekstomrade3 = new StringBuffer("Høst
    2012\n");
151         StringBuffer tekstomrade4 = new StringBuffer("Høst
    2013\n");
152         StringBuffer tekstomrade5 = new StringBuffer("Høst
```

```
    2014\n");
153
154         String utskrift = "";
155         int valgAar =
    Integer.parseInt(JOptionPane.showInputDialog(null,
156                 "Skriv inn tallet p¬ hvilket ¬rskull du vil se\n"
157                 + "1: 2010\t"
158                 + "2: 2011\t"
159                 + "3: 2012\t "
160                 + "4: 2013\t"
161                 + "5: 2014\t"));
162
163         /* IF-SETNINGER SOM STARTER SORTERING AV ≈RSKULL. FyRSTE
    IF-SETNING
164         KOMMENTERT, GANGEN ER LIK I ELSE-IF-SETNINGENE.*/
165         if (valgAar == 1)
166         {
167             // for-løkke som kjører gjennom hele arrayens lengde
168             for (int i = 0; i < stud.length; i++)
169             {
170                 // plukker ut studenter som startet i 2010
171                 if (stud[i].getStudiestart() == 2010)
172                 {
173                     // studentens fornavn og etternavn legges til i
    tekstområdet
174                     tekstomrade1.append(stud[i].getFornavn()
175                             + " " + stud[i].getEtternavn());
176                     tekstomrade1.append("\n"); // linjeskift legges
    til
177                 }
178             }
179             // utskrift av utfyllt tekstområde med ferdigsortert
    årskull.
180             utskrift = tekstomrade1 + "";
181         }
182         else if (valgAar == 2)
183         {
184             for (int i = 0; i < stud.length; i++)
185             {
186                 if (stud[i].getStudiestart() == 2011)
187                 {
188                     tekstomrade2.append(stud[i].getFornavn()
189                             + " " + stud[i].getEtternavn());
190                     tekstomrade2.append("\n");
191                 }
```

```
192             }
193             utskrift = tekstomrade2 + "";
194         }
195         else if (valgAar == 3)
196         {
197             for (int i = 0; i < stud.length; i++)
198             {
199                 if (stud[i].getStudiestart() == 2012)
200                 {
201                     tekstomrade3.append(stud[i].getFornavn()
202                             + " " + stud[i].getEtternavn());
203                     tekstomrade3.append("\n");
204                 }
205             }
206             utskrift = tekstomrade3 + "";
207         }
208         else if (valgAar == 4)
209         {
210             for (int i = 0; i < stud.length; i++)
211             {
212                 if (stud[i].getStudiestart() == 2013)
213                 {
214                     tekstomrade4.append(stud[i].getFornavn()
215                             + " " + stud[i].getEtternavn());
216                     tekstomrade4.append("\n");
217                 }
218             }
219             utskrift = tekstomrade4 + "";
220         }
221         else if (valgAar == 5)
222         {
223             for (int i = 0; i < stud.length; i++)
224             {
225                 if (stud[i].getStudiestart() == 2014)
226                 {
227                     tekstomrade5.append(stud[i].getFornavn()
228                             + " " + stud[i].getEtternavn());
229                     tekstomrade5.append("\n");
230                 }
231             }
232             utskrift = tekstomrade5 + "";
233         }
234 
235 
236         /* Til slutt sjekkes det om brukeren skriver inn noe feil
```

```
237         eller trykker OK/kryss. Gir isåfall bruker mulighet til å
    starte på nytt*/
238         else
239         {
240                 int ja = JOptionPane.showOptionDialog(null,
241                         "Vil du forsøke å skrive inn riktig år på
    nytt?",
242                         "Error! Error!",
243                         JOptionPane.YES_NO_OPTION,
    JOptionPane.QUESTION_MESSAGE,
244                         null, null, null);
245                 // hvis ja, sett ny verdi til variabelen valgAar.
246                 if (ja == JOptionPane.YES_OPTION)
247                 {
248                     valgAar =
    Integer.parseInt(JOptionPane.showInputDialog(null,
249                             "Skriv inn tallet p¬ hvilket ¬rskull du
    vil se\n"
250                             + "1: 2010\t"
251                             + "2: 2011\t"
252                             + "3: 2012\t "
253                             + "4: 2013\t"
254                             + "5: 2014\t"));
255                 }
256                 else
257                 {
258                     // hvis ikke ja (altså nei i dette tilfellet);
    avslutt programmet.
259                     System.exit(0);
260                 } // slutt else
261         } // slutt else
262
263         JOptionPane.showMessageDialog(null, utskrift);
264     } // slutt metode sortereStudestart
265
266     // Lager liste over studenter som er kvalifisert til eksamen.
267     public void sortereGodkjent ()
268     {
269         StringBuffer tekstomrade = new StringBuffer("Studenter
    kvalifisert til eksamen: \n");
270
271         // for-løkke som kjører gjennom hele arrayens lengde,
    objekt for objekt
272         for (int i = 0; i < stud.length; i++)
273         {
```

```
274             // Plukker ut studentene som er kvalifisert til
    eksamen.
275             if (stud[i].isGodkjent() == true)
276             {
277                 // Fornavn og etternavn legges til i tekstområdet.
278                 tekstomrade.append(stud[i].getFornavn()
279                         + " " + stud[i].getEtternavn());
280                 tekstomrade.append("\n");
281             }
282         }
283         utskrift = tekstomrade + "";
284         // ferdigsortert studentgruppe skrives ut i dialogboks.
285         JOptionPane.showMessageDialog(null, utskrift);
286     } // slutt metode sortereStudiestart
287 
288     // Lager alfabetisk oversikt over studentene (ekstrasnacks og
    særdeles nyttig sortering)
289     public void sortereAlfabetisk (Student[] stud)
290     {
291         /* variabel av typen Stringbuffer, hvor verdien modifiseres
292         kontinuerlig gjennom metoden*/
293         StringBuffer tekstomrade = new StringBuffer("Alfabetisk
    liste over studenter: \n");
294 
295         /* initialisering av ny array for bare studentenes fornavn
    (her kunne vi
296         også laget på etternavn, men vi liker å være uformelle)*/
297         String[] studenter = new String[stud.length];
298         /* for-løkke som kjører gjennom hele arrayens lengde,
    objekt for objekt,
299         og henter ut fornavnet deres*/
300         for (int i = 0; i < studenter.length; i++)
301         {
302             studenter[i] = stud[i].toString();
303         }
304 
305         Arrays.sort(studenter); // metode som sorterer array
    "studenter"
306 
307         // for-løkke som henter ut navnene til alle studentene i
    array.
308         for ( int i = 0 ; i < stud.length ; i++ )
309         {
310             /* skriver ut studentene i et tekstområde etterhvert
    som
```

```
311             de blir sortert i array.*/
312             tekstomrade.append(studenter[i] + "\n");
313         } // slutt for-løkke
314         utskrift = tekstomrade + "";
315         // utskrift av alfabetisk liste i et dialogvindu.
316         JOptionPane.showMessageDialog(null, utskrift);
317     } // slutt metode sortereAlfabetisk
318 
319     // start metode sortere tilfeldig.
320     public void sortereTilfeldig (Student[] stud)
321     {
322         // Ny array som henter inn fornavnene på studentene.
323         List studenter = new ArrayList();
324         // for-løkke som kjører gjennom arrayen og formaterer
    fornavn til String-variabler.
325         for (int i = 0; i < stud.length; i++)
326         {
327             studenter.add(stud[i].toString());
328         }
329         // metode som "shuffler"/stokker om objektene i arrayen.
330         Collections.shuffle(studenter);
331 
332         // bruker blir spurt antall grupper ønsket.
333         int antallGrupper =
    Integer.parseInt(JOptionPane.showInputDialog(null,
334                 "Hvor mange grupper vil du sortere studentene i?
    \n"));
335 
336         StringBuffer[] sb = new StringBuffer[antallGrupper];
337         // for-løkke som kjører gjennom hele arrayens lengde
338         for (int i = 0; i < sb.length; i++)
339         {
340             sb[i] = new StringBuffer();
341         }
342 
343         int counter = 0;
344         int studnr = 0;
345 
346 
347         while (studnr < studenter.size())
348         {
349             /* Gruppefordeling:
350              * Ved hjelp av en teller plasseres studentene i
    antallGrupper grupper.
351              * Når en student er plassert i hver gruppe kjøres
```

```
   counter-variabelen
352             * til 0, og neste student blir plassert i gruppe 1,
   neste etter det i
353             * gruppe 2 */
354            // første student plasseres i første gruppe, andre i
   andre osv.
355            sb[counter].append(studenter.get(studnr) + "\n");
356            counter++;
357            studnr++;
358
359            /* counter-variabel teller oppover så lenge den er
   mindre enn verdien
360            til antallGrupper; blir den større settes den til 0.*/
361            if (counter >= antallGrupper)
362                counter = 0;
363        } // slutt while-løkke
364
365        // skriver ut studentene, ferdig sortert og fordelt i
   grupper.
366        for (int i = 0; i < studenter.size(); i++)
367        {
368            System.out.println(studenter.get(i));
369        }
370
371        utskrift = "";
372
373        /* går gjennom utskriften og formaterer til dem til String,
374        slik at de kan skrives ut i JOptionPane.*/
375        for (int i = 0; i < sb.length; i++)
376        {
377            utskrift += sb[i].toString() + "\n\n";
378        }
379        JOptionPane.showMessageDialog(null, utskrift);
380    }// slutt metode sortere tilfeldig
381 } // slutt klasse
```

```java
/*****************************************************
 * @author Øystein
 * GUIMain.class
 *
 * Hovedvinduet for å utføre endringer på studenter.
 *****************************************************/
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import javax.swing.DefaultListModel;
import javax.swing.JButton;
import javax.swing.JList;
import javax.swing.JScrollPane;

public class GUIMain extends Dialog implements MouseListener, WindowListener{


    //Main
    //Vis valg (legg til studenter ELLER lag lister)
    //Hvis legg til studenter
    //--Spør om informasjon, lag nytt studentobjekt
    //Hvis lag lister
    //--Spør hva slags liste, så skriv ut.

    private Gruppe gruppe = null;

    private JList listStudenter = null;
    private DefaultListModel listModel = new DefaultListModel();
    private JScrollPane scrollStudenter = null;

    private JButton btnRedigerElev = new JButton("Rediger elev og oppgaver");
    private Main m = null;

    public GUIMain(Gruppe grp, Main m){
        gruppe = grp;
        this.m = m;
        setup();

        this.setVisible(true);
    }
```

```
44
45      protected void setup(){ //Ordner vinduet klart til
    førstegangsvisning
46          //Kjører setup-metoden fra Dialog-klassen (som denne
    klassen arver fra)
47          super.setup();
48          this.setTitle("Studentadministrasjon");
49          this.setSize(320, 240);
50
51          /*Legger til en vinduslytter (Dette for å vite når vinduet
52          lukkes, slik at visMeny() fra Main kan kalles opp igjen.*/
53          this.addWindowListener(this);
54
55          //Gjør klar listen med studenter
56          updateListe();
57          listStudenter = new JList(listModel);
58          //Aktiverer en scrollbar dersom det trengs til lista.
59          scrollStudenter = new JScrollPane(listStudenter);
60
61          //Legg komponenter og paneler til i dialogvinduet
62          this.add(scrollStudenter, BorderLayout.CENTER);
63          this.add(btnRedigerElev, BorderLayout.SOUTH);
64
65          //Legg til lytterer etter knappetrykk og museklikk
66          btnRedigerElev.addActionListener(this);
67          listStudenter.addMouseListener(this);
68
69          //Sentrer dialogen på PC-skjermen
70          centerScreen();
71      }
72
73      //Returnerer studentobjektet som tilhører studenten som er valg
    i lista.
74      private Student getValgteStudent(){
75          return gruppe.hentStudenterAsVector().elementAt(
76                  listStudenter.getSelectedIndex());
77      }
78
79      private void updateListe(){//Oppdaterer innholdet i listboksen
80          Student[] stud = gruppe.hentStudenterAsArray();
81
82          listModel.clear(); //tøm lista
83
84          //Legg til alle studenter i lista
85          for (int i = 0; i < stud.length; i++) {
```

```
 86             listModel.addElement(stud[i]);
 87         }
 88 
 89     }
 90 
 91     //Tar hånd om hva som skjer når noe blir klikket på
 92     public void actionPerformed(ActionEvent ae) {
 93 
 94         if (ae.getSource() == btnRedigerElev){
 95             //Hvis knappen "btnRedigerElev" klikkes
 96             //Vis et vindu for redigering av studenten som er valgt
 97             new GUIStudent(gruppe, getValgteStudent());
 98             Filbehandling.lagreGruppe(gruppe);//Lagre
 99             updateListe(); //oppdater lista (i tilfelle
    navnendringer etc.).
100         }
101     }
102 
103 
104 
105     public void mouseClicked(MouseEvent me) { //Hvis brukeren
    dobbeltklikker
106           if (me.getClickCount() == 2) {
107             //Vis et vindu for redigering av studenten som har
    blitt klikket.
108                 new GUIStudent(gruppe, getValgteStudent());
109                 Filbehandling.lagreGruppe(gruppe); //Lagre
110                 updateListe(); //oppdater lista (i tilfelle
    navnendringer etc.).
111              }
112     }
113 
114     public void windowClosed(WindowEvent arg0) {
115         // TODO Auto-generated method stub
116         m.visMeny();
117     }
118 
119     /*Under her er alle (ubrukte) metodene som kreves
120     av ActionListener og WindowsListener*/
121 
122     public void mouseEntered(MouseEvent arg0) {
123     }
124 
125     public void mouseExited(MouseEvent arg0) {
126     }
```

```
127
128     public void mousePressed(MouseEvent arg0) {
129     }
130
131     public void mouseReleased(MouseEvent arg0) {
132     }
133
134     @Override
135     public void windowActivated(WindowEvent arg0) {
136         // TODO Auto-generated method stub
137
138     }
139
140     @Override
141     public void windowClosing(WindowEvent arg0) {
142         // TODO Auto-generated method stub
143
144     }
145
146     @Override
147     public void windowDeactivated(WindowEvent arg0) {
148         // TODO Auto-generated method stub
149
150     }
151
152     @Override
153     public void windowDeiconified(WindowEvent arg0) {
154         // TODO Auto-generated method stub
155
156     }
157
158     @Override
159     public void windowIconified(WindowEvent arg0) {
160         // TODO Auto-generated method stub
161
162     }
163
164     @Override
165     public void windowOpened(WindowEvent arg0) {
166         // TODO Auto-generated method stub
167
168     }
169
170 }
171
```

```java
/*************************************************************
 * @author Øystein
 *
 * GUIOppgave.class
 * Vindu for Â vise informasjon om ett enkelt oppgaveobjekt.
 *************************************************************/
import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import javax.swing.ButtonGroup;
import javax.swing.JButton;
import javax.swing.JFormattedTextField;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.text.MaskFormatter;

public class GUIOppgave extends Dialog{


    private JPanel panelNorth = new JPanel(new GridLayout(0,1));
    private JPanel panelNorthNavn = new JPanel( new
GridLayout(0,1));
    private JPanel panelNorthDato = new JPanel( new
GridLayout(0,1));
    private JPanel panelNorthRadio = new JPanel( new FlowLayout());
    private JPanel panelSouth = new JPanel( new FlowLayout());

    private JButton btnOk = new JButton("Ok");
    private JButton btnAvbryt = new JButton("Avbryt");

    private JRadioButton radioBestatt = new JRadioButton("Best/
u00E5tt");
    private JRadioButton radioIkkeBestatt = new JRadioButton("Ikke
best/u00E5tt");
    private ButtonGroup groupBestatt = new ButtonGroup();

    private JTextArea txtTilbakemelding = new JTextArea();
    private JScrollPane scroll = new
JScrollPane(txtTilbakemelding);

```

```
41      private JFormattedTextField txtLevertDato = null;
42      private JLabel lblLevertDato = new JLabel("Levert dato:");
43
44      private JLabel lblOppgaveNavn = new JLabel("Oppgavenavn:");
45      private JTextField txtOppgaveNavn = new JTextField();
46
47      private Oppgave oppg = null;
48      private Student stud = null;
49
50      //Konstruktur med studentobjekt (for Â opprette ny oppgave)
51      public GUIOppgave(Student stud){
52          this.stud = stud;
53          setup();
54          this.setVisible(true);
55      }
56
57      //Konstruktør med oppgaveobjekt (for redigering)
58      public GUIOppgave(Oppgave oppg){
59          this.oppg = oppg;
60          setup();
61          oppdaterFelt();
62          this.setVisible(true);
63      }
64
65      /*Oppdaterer felt med informasjon fra oppg-objekt
66      (brukt ved endring av eksisterende objekt).*/
67      private void oppdaterFelt(){
68          txtLevertDato.setText( oppg.getLevertDato());
69          txtOppgaveNavn.setText( oppg.getOppgaveNavn());
70          txtTilbakemelding.setText( oppg.getTilbakeMelding());
71          if (oppg.isGodkjent()) radioBestatt.setSelected(true);
72          else radioIkkeBestatt.setSelected(true);
73      }
74
75      private boolean sjekkFelt(){ //Kontrollerer at alle felt er
    utfyllt
76          if (txtLevertDato.getText().isEmpty()) return false;
77          if (txtLevertDato.getText().equals("")) return false;
78          if (txtOppgaveNavn.getText().isEmpty()) return false;
79          if (txtOppgaveNavn.getText().equals("")) return false;
80
81          return true;
82
83      }
84
```

```
85      protected void setup(){
86          super.setup(); //Kjører setup fra moderklassen
87          this.setTitle("Oppgaver");
88          this.setSize(190, 280);
89          this.setModal(true); //Kun dette vinduet aktivt
90
91          //Konfigurere tekstfelt for integer (Årstall)
92          txtLevertDato = new
    JFormattedTextField( datoFormatter("##.##.####"));
93
94          //Ordne radioknapper
95          groupBestatt.add(radioBestatt);
96          groupBestatt.add(radioIkkeBestatt);
97          radioIkkeBestatt.setSelected(true);
98
99          //Legge til komponenter i panelene
100         panelNorthNavn.add(lblOppgaveNavn);
101         panelNorthNavn.add(txtOppgaveNavn);
102         panelNorthDato.add(lblLevertDato);
103         panelNorthDato.add(txtLevertDato);
104         panelNorthRadio.add(radioBestatt);
105         panelNorthRadio.add(radioIkkeBestatt);
106         panelNorth.add(panelNorthNavn);
107         panelNorth.add(panelNorthDato);
108         panelNorth.add(panelNorthRadio);
109         panelSouth.add(btnAvbryt);
110         panelSouth.add(btnOk);
111
112         //Legge til klikklytter
113         btnAvbryt.addActionListener(this);
114         btnOk.addActionListener(this);
115
116
117         //Legge komponentene ut i vinduet/dialogen
118         this.add(panelNorth, BorderLayout.NORTH);
119         this.add(scroll, BorderLayout.CENTER);
120         this.add(panelSouth, BorderLayout.SOUTH);
121
122
123         centerScreen();
124     }
125
126
127     public void actionPerformed(ActionEvent ae) { //Sjekker etter
    knappetrykk
```

```
128
129         if (ae.getSource() == btnOk){
130             clickOk();
131         }else if (ae.getSource() == btnAvbryt){
132             this.setVisible(false);
133         }
134     }
135
136     private void clickOk(){ /*Hva som skjer nÂr knappen "OK"
137         klikker - Setter variablene i oppgaveobjektet.*/
138
139         if (sjekkFelt()){
140             if (oppg != null){ //AltsÂ redigerer en eksisterende
    oppgave
141                 oppdaterOppgaveObjekt();
142             }else{ //AltsÂ oppretter en ny oppgave
143                 oppg = new Oppgave();
144                 oppdaterOppgaveObjekt();
145                 stud.addOppgave(oppg);
146             }
147             this.setVisible(false);
148         }
149
150     }
151
152     private void oppdaterOppgaveObjekt(){ /*Oppdaterer
    oppgaveobjektet ut i
153         fra hva brukeren har skrevet i vinduet.*/
154         oppg.setOppgaveNavn( txtOppgaveNavn.getText());
155         oppg.setLevertDato(txtLevertDato.getText());
156         oppg.settTilbakeMelding(txtTilbakemelding.getText());
157         if (radioBestatt.isSelected()) oppg.setGodkjent(true); else
    oppg.setGodkjent(false);
158     }
159
160     //Sørger for at det kun blir tastet inn datoer i formatet
    XX.XX.XXXX
161     protected MaskFormatter datoFormatter(String datoString) {
162         MaskFormatter formatter = null;
163         try {
164             formatter = new MaskFormatter(datoString);
165           //Hvis feil, si i fra om dette og sett dato til
    01/01/2001.
166         } catch (java.text.ParseException pe) {
167             System.err.println("Feil med datotekstfeltet for
```

```
   innlevering" + pe);
168             txtLevertDato.setText("01.01.2001");
169         }
170         return formatter;
171     }
172 }
```

```java
/*************************************************************
 * @author Øystein
 *
 * GUIOppgaver.class
 *
 * Hovedvindu for å vise alle oppgaver en student har levert.
 *************************************************************/
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import javax.swing.DefaultListModel;
import javax.swing.JButton;
import javax.swing.JList;
import javax.swing.JPanel;
import javax.swing.JScrollPane;

public class GUIOppgaver extends Dialog implements MouseListener{

    private JPanel panelEast = new JPanel(new GridLayout(0,1));

    private JList listOppgaver = null;
    private JScrollPane scrollOppgaver = null;
    private DefaultListModel listModel = new DefaultListModel();

    private JButton btnLeggTilOppgave = new JButton("Legg til...");
    private JButton btnRedigerOppgave = new JButton("Rediger...");
    private JButton btnSlettOppgave = new JButton("Slett");
    private JButton btnLukk = new JButton("Lukk");

    private Student stud = null;

    public GUIOppgaver(Student stud){
        this.stud = stud;
        setup();

        this.setVisible(true);
    }

    //Ordner vinduet klart til førstegangsvisning
    protected void setup(){
        //Kjører setup-metoden fra Dialog-klassen (som denne klassen arver fra)
        super.setup();
```

```
45          this.setTitle(stud.getFornavn() + " -
    Oppgaveadministrasjon");
46          this.setSize(320,240);
47          this.setModal(true);
48
49
50          //Gjør klar listen med oppgaver
51          updateListe();
52          listOppgaver = new JList(listModel);
53          scrollOppgaver = new JScrollPane(listOppgaver);
54
55          /*listOppgaver = new JList(stud.getOppgaver());
56           * //Lar deg kun velge en og en ting i lista.
57
    listOppgaver.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
58          scrollOppgaver = new JScrollPane(listOppgaver); //Skrur på
    scrollbar*/
59
60          //Legg alle komponenter og knapper til paneler
61          panelEast.add(btnLeggTilOppgave);
62          panelEast.add(btnRedigerOppgave);
63          panelEast.add(btnSlettOppgave);
64          panelEast.add(btnLukk);
65
66          //Legg komponenter og paneler til i dialogvinduet
67          this.add(scrollOppgaver, BorderLayout.CENTER);
68          this.add(panelEast, BorderLayout.EAST);
69
70          //Legg til lytterer etter knappetrykk
71          btnLeggTilOppgave.addActionListener(this);
72          btnRedigerOppgave.addActionListener(this);
73          btnSlettOppgave.addActionListener(this);
74          btnLukk.addActionListener(this);
75          listOppgaver.addMouseListener(this);
76
77          //Sentrer dialogen på PC-skjermen
78          centerScreen();
79
80      }
81
82      //Tar hånd om hva som skjer når noe blir klikket på
83      public void actionPerformed(ActionEvent ae) {
84
85          if (ae.getSource() == btnLeggTilOppgave){
86              new GUIOppgave(stud);
```

```
 87             updateListe();
 88         }else if (ae.getSource() == btnRedigerOppgave){
 89             //åpner et oppgavevindu, sender med valgte oppgave.
 90             new GUIOppgave( getValgteOppgave());
 91             updateListe();
 92         }else if (ae.getSource() == btnSlettOppgave){
 93             //Sletter valge oppgaven
 94 
    stud.getOppgaverAsVector().removeElementAt(listOppgaver.getSelected
    Index());
 95             updateListe();
 96         }else if (ae.getSource() == btnLukk){
 97             this.setVisible(false);
 98         }
 99     }
100 
101     //Returnerer oppgaveobjektet som tilsvarer oppgaven som er
    valgt i lista.
102     private Oppgave getValgteOppgave(){
103         return
    stud.getOppgaverAsVector().elementAt( listOppgaver.getSelectedIndex
    ());
104     }
105 
106 
107     private void updateListe(){//Oppdaterer innholdet i listboksen
108         Oppgave[] oppgtemp = stud.getOppgaver();
109         listModel.clear();
110 
111         for (int i = 0; i < oppgtemp.length; i++) {
112             listModel.addElement(oppgtemp[i]);
113         }
114 
115     }
116 
117 
118     public void mouseClicked(MouseEvent me) { //Tar hånd om
    museklikk
119           if (me.getClickCount() == 2) {
120             //åpner et oppgavevindu, sender med valgte oppgave.
121                 new GUIOppgave( getValgteOppgave());
122                 updateListe();
123              }
124     }
125 
```

```
126     public void mouseEntered(MouseEvent arg0) {
127     }
128
129     public void mouseExited(MouseEvent arg0) {
130     }
131
132     public void mousePressed(MouseEvent arg0) {
133     }
134
135     public void mouseReleased(MouseEvent arg0) {
136     }
137 }
```

```java
/*************************************************************
 * @author Øystein
 *
 * GUIStudent.class
 * Hovedvindu for Â redigere informasjon om et studentobjekt.
 *************************************************************/
import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.text.NumberFormat;
import java.util.Vector;
import javax.swing.ButtonGroup;
import javax.swing.JButton;
import javax.swing.JFormattedTextField;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JTextField;
import javax.swing.text.NumberFormatter;

public class GUIStudent extends Dialog{


    private Student stud = null;
    private Gruppe gruppe = null;
    private Vector<Oppgave> oppgaver = new Vector();


    private JPanel panelSouth = new JPanel(new GridLayout(0,3));
    private JPanel panelRadioKjonn = new JPanel(new FlowLayout());
    private JPanel panelCenter = new JPanel(new GridLayout(0,2));

    private JButton btnSlett = new JButton("Slett");
    private JButton btnLagre = new JButton("Lagre");
    private JButton btnAvbryt = new JButton("Avbryt");
    private JButton btnOppgaver = new JButton("Oppgaver");

    private JRadioButton radioMann = new JRadioButton("Mann");
    private JRadioButton radioDame = new JRadioButton("Dame");
    private ButtonGroup groupKjonn = new ButtonGroup();

    private JLabel lblFornavn = new JLabel("Fornavn");
    private JLabel lblEtternavn = new JLabel("Etternavn");
    private JLabel lblFag = new JLabel("Fag");
```

```
46      private JLabel lblOppgaver = new JLabel("Oppgaver");
47      private JLabel lblKjonn = new JLabel("Kj\u00F8nn");
48      private JLabel lblStudieStart = new JLabel("Studiestart");
49
50      private JTextField txtFornavn = new JTextField();
51      private JTextField txtEtternavn = new JTextField();
52      private JTextField txtFag = new JTextField();
53      private JFormattedTextField txtStudieStart = null;
54
55
56      public GUIStudent(Gruppe gruppe){//Konstrukt¯r brukt for Â
  opprette en ny student.
57
58          this.gruppe = gruppe;
59
60          setup();
61          btnSlett.setVisible(false); //Hjemmer knappen ettetsom den
  kun skal vises ved endring av student
62          btnOppgaver.setVisible(false); //Hjemmer knappen ettetsom
  den kun skal vises ved endring av student
63          lblOppgaver.setVisible(false); //Hjemmer knappen ettetsom
  den kun skal vises ved endring av student
64
65          this.setVisible(true);
66      }
67
68      public GUIStudent(Gruppe gruppe, Student stud){ //Konstrukt¯r
  brukt for Â endre en eksisterende student.
69          this.gruppe = gruppe;
70          this.stud = stud;
71          setup();
72          oppdaterFelt();
73
74          this.setVisible(true);
75      }
76
77
78      protected void setup(){ //Ordner vinduet klart til
  f¯rstegangsvisning
79          super.setup(); //Kj¯rer setup-metoden fra Dialog-klassen
  (som denne klassen arver fra)
80          this.setTitle("Studentadministrasjon - Student");
81          this.setSize(300,220);
82          this.setModal(true); //For Â lÂse hovedvinduet mens man har
  dette studentvinduet Âpent.
```

```
83
84          //Konfigurere tekstfelt for integer (Årstall)
85          NumberFormat intFormat = NumberFormat.getNumberInstance();
86          intFormat.setGroupingUsed(false); //slÅ av gruppering av
    tall (noe som f¯rer til mellomrom i tallrekken, og feil i integer
    konvertering fra string.
87          NumberFormatter numberFormatter = new
    NumberFormatter(intFormat);
88          txtStudieStart = new JFormattedTextField(numberFormatter);
89          numberFormatter.setValueClass(Integer.class);
90          numberFormatter.setAllowsInvalid(false); //tillat kun
    nummer
91          txtStudieStart.setToolTipText("/u00C5rstall for
    studiestart. F.eks. \"2014\"");
92
93
94          //Gruppere radioknapper
95          groupKjonn.add(radioMann);
96          groupKjonn.add(radioDame);
97          radioDame.setSelected(true);
98          panelRadioKjonn.add(radioDame);
99          panelRadioKjonn.add(radioMann);
100
101         //Legg alle komponenter og knapper til paneler
102         panelCenter.add(lblFornavn);
103         panelCenter.add(txtFornavn);
104         panelCenter.add(lblEtternavn);
105         panelCenter.add(txtEtternavn);
106         panelCenter.add(lblFag);
107         panelCenter.add(txtFag);
108         panelCenter.add(lblOppgaver);
109         panelCenter.add(btnOppgaver);
110         panelCenter.add(lblKjonn);
111         panelCenter.add(panelRadioKjonn);
112         panelCenter.add(lblStudieStart);
113         panelCenter.add(txtStudieStart);
114
115         panelSouth.add(btnAvbryt);
116         panelSouth.add(btnSlett);
117         panelSouth.add(btnLagre);
118
119
120         //Legg komponenter og paneler til i dialogvinduet
121         this.add(panelCenter, BorderLayout.CENTER);
122         this.add(panelSouth, BorderLayout.SOUTH);
```

```
123
124         //Legg til lytterer etter knappetrykk
125         btnAvbryt.addActionListener(this);
126         btnLagre.addActionListener(this);
127         btnSlett.addActionListener(this);
128         btnOppgaver.addActionListener(this);
129
130         //Sentrer dialogen pÂ PC-skjermen
131         centerScreen();
132
133     }
134
135     public void actionPerformed(ActionEvent ae) { //Tar hÂnd om
    knappetrykk
136
137         if (ae.getSource() == btnAvbryt){
138             this.setVisible(false);
139         }else if (ae.getSource() == btnLagre){
140             klikkLagre();
141         }else if (ae.getSource() == btnOppgaver){
142             System.out.println(stud.getOppgaverAsVector().size());
143             new GUIOppgaver(stud);
144             System.out.println(stud.getOppgaverAsVector().size());
145         }
146     }
147
148
149     private void oppdaterFelt(){ //Oppdaterer alle felt ved endring
150         txtFornavn.setText( stud.getFornavn());
151         txtEtternavn.setText( stud.getEtternavn());
152
    txtStudieStart.setText( Integer.toString(stud.getStudiestart()));
153         if (stud.isDame()) radioDame.setSelected(true); else
    radioMann.setSelected(true);
154         txtFag.setText(stud.getFag());
155     }
156
157
158     private boolean sjekkFelt(){//Kontrollerer at alle felt er
    utfylt
159
160         if (txtFornavn.getText().isEmpty()) return false;
161         if (txtFornavn.getText().equals("")) return false;
162         if (txtEtternavn.getText().isEmpty()) return false;
163         if (txtEtternavn.getText().equals("")) return false;
```

```
164         if (txtFag.getText().isEmpty()) return false;
165         if (txtFag.getText().equals("")) return false;
166         if (txtStudieStart.getText().isEmpty()) return false;
167         if (txtStudieStart.getText().equals("")) return false;
168 
169 
170         return true;
171 
172     }
173 
174     private void klikkLagre(){ //Endrer/oppretter og lagrer
    studentobjektet.
175 
176 
177         if (sjekkFelt()){
178             if (stud == null){//Ny student
179                 stud = new Student(txtFornavn.getText(),
    txtEtternavn.getText(), Integer.parseInt(txtStudieStart.getText()),
    radioMann.isSelected(), txtFag.getText(), oppgaver);
180                 gruppe.leggTilStudent(stud);
181             }else{ // Endre student
182                 stud.setFornavn(txtFornavn.getText());
183                 stud.setEtternavn(txtEtternavn.getText());
184                 stud.setStudiestart(txtStudieStart.getText());
185                 stud.setKjonn(radioMann.isSelected());
186                 stud.setFag(txtFag.getText());
187             }
188 
189             Filbehandling.lagreGruppe(gruppe); //Lagrer endringer
    til disk.
190             this.setVisible(false);
191         }
192     }
193 
194 
195 }
196 
```

```java
/*********************************************
 * @author Øystein
 *********************************************/
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class GUIVisGruppe extends Dialog
{
    private String navnliste = null;

    //private JButton btnLukk = new JButton("Lukk");
    private JTextArea txtListe = new JTextArea();
    private JScrollPane scroll = new JScrollPane(txtListe);




    public GUIVisGruppe(String navnliste)
    {
        this.navnliste = navnliste;

        setup();

        setVisible(true);
    }

    protected void setup()
    {
        super.setup();

        this.setModal(true);
        this.setSize(320, 240);



        //Sette opp trykklyttere
        btnLukk.addActionListener(this);

        //Legge til navn i lista - Sette klar tekstboksen
        txtListe.setText(navnliste);
        txtListe.setEditable(false);

        //Legge ut komponenter
```

```
46          this.add(scroll, BorderLayout.CENTER);
47          this.add(btnLukk, BorderLayout.SOUTH);
48
49          centerScreen();
50      }
51
52      public void actionPerformed(ActionEvent ae)
53      {
54          super.actionPerformed(ae);
55
56      }
57
58  }
```