Avdeling for allmenne fag

EKSAMEN

801 OBJEKTBASERT PROGRAMMERING

14.05.2004

Tid:	3 timar / 4 timar
Målform:	Bokmål/nynorsk
Sidetal:	8 med framsida og vedlegg
Hjelpemiddel:	Alt skriftleg materiale
Merknader:	Ingen
Vedlegg:	A, B og C (inkludert i oppgavesettet)

Eksamensresultata blir offentliggjort på lister med kandidatnummer, lagt i hyllene ved eksamensoppslagstavla, eller du kan gå inn på Arena høgskole – da trenger du heller ikkje kandidatnummeret.

Bokmål

Noen deloppgaver skal kun besvares av de som har fritak for midtprøven / hjemmeoppgaven, og dermed får utvidet sluttprøve. Disse deloppgavene er merket: "Skal kun besvares av de som har 4 timers sluttprøve".

Vekting av deloppgaver i forhold til endelig karakter i emnet går fram av følgende tabell:

Deloppgave	Vekting 3 timer	Vekting 4 timer
1-a	10 %	10 %
1-b	10 %	10 %
1-c	Skal ikke gjøres	10 %
2-a	20 %	20 %
2-b	20 %	20 %
2-c	Skal ikke gjøres	30 %
Innlevering	20 %	Fritatt
Midtprøve	20 %	Fritatt

Les oppgaveteksten nøye, og prøv å svare på så mange deloppgaver som mulig. Lykke til!

Oppgave 1

1-a)

Du skal lage en metode for å lese inn heltall fra bruker:

public static int lesHeltall (int fra, int til)

Hvis brukeren skriver noe som ikke er et lovlig heltall fra og med parameteren *fra* og til parameteren *til* skal metoden gjenta innlesingen helt til inndata er korrekt.

Innlesing skal gjøres ved hjelp av metoden *showInputDialog*, som er definert i klassen *JOptionPane*:

public static String showInputDialog(String ledetekst)

Denne metoden viser et dialogvindu med meldingen *ledetekst* og en tekstboks der brukeren kan skrive inndata. Når brukeren trykker *enter* blir innholdet i tekstboksen returnert.

1-b)

Du skal programmere en såkalt krypteringsalgoritme:

public static void kryptere(int nøkkel, int[] data)

Alle verdiene i tabellen *data* skal være et av tallene 0...255, og du kan tenke deg at hvert tall representerer et tegn (for eksempel en bokstav). Hele tabellen representerer dermed en tekst. Parameteren *nøkkel* er et positivt tall. Metoden skal oppdatere hvert element i tabellen ved å legge til *nøkkel*. Resultatverdiene skal imidlertid også være blant tallene 0...255. Verdier over 255 må dermed "reduseres" ved at man "starter på 0 igjen". Eksempel: Med *nøkkel* lik 2 skal verdi 0 bli til 2, 1 bli til 3, 2 bli til 4, ..., 253 bli til 255, 254 bli til 0 og 255 bli til 1.

1-c) Skal kun besvares av de som har 4 timers sluttprøve

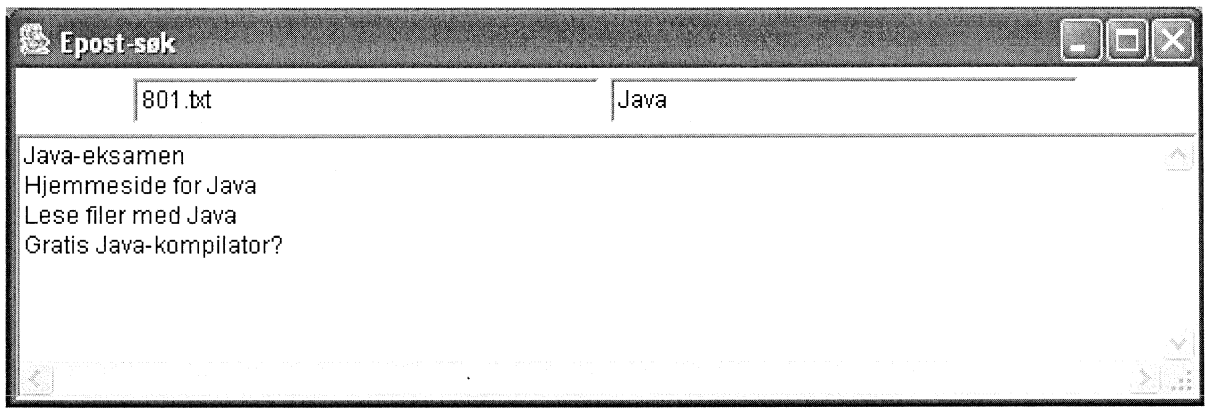
Du skal nå lage en litt mer sofistikert krypteringsalgoritme.

```
public static void kryptere2( int[] nøkkel, int[] data )
```

Tabellen *data* inneholder verdier blant tallene 0...255, og vi kan tenke oss at den representerer en tekst som beskrevet i deloppgave 1-b. Heltallstabellen *nøkkel* inneholder alle verdiene 0...255 nøyaktig 1 gang, og rekkefølgen i denne tabellen bestemmer hvordan verdiene i *data* skal krypteres: Alle forekomster av en verdi X i tabellen *data* skal endres til den verdien som er lagret på posisjon X i tabellen *nøkkel*. Hvis for eksempel data[20] inneholder 14 og nøkkel[14] inneholder 3, så skal data[20] bli satt til 3.

Oppgave 2

Du skal lage utvalgte deler av et program for å søke i tekstfiler som inneholder e-post'ene til en bruker. Et eksempel på en e-postfil er vedlagt. Tittelen til en e-post forekommer på linjer som begynner med Subject, og er den eneste informasjonen fra filen som dette programmet skal bruke. Brukergrensesnittet skal se ut som følger:



Brukeren skriver inn et filnavn i venstre tekstboks og et søkeord i høyre tekstboks. Det store tekstområdet skal hele tiden vise de titlene i filen som inneholder søkeordet. Programmet skal bygges opp fra en brukergrensesnitt-klasse *EpostGUI* og en modellklasse *EpostTitler*. Deler av begge klassene er vedlagt.

2-a)

Du skal programmere ferdig brukergrensesnittklassen *EpostGUI* (se vedlegg A), som innebærer å definere konstruktøren og metoden *actionPerformed*. Få med kall på metoder i modellklassen!

2-b)

Du skal programmere metoden *lesFil* i klassen *EpostTitler* (se vedlegg B). Tittelen til en e-post står på linjen som starter med Subject (se vedlegg C). Metoden får navnet på filen som parameter, skal lese inn alle titlene som forekommer i filen, og lagre disse i tabellen *tittelTab* (instansvariabel i klassen). Følgende to metoder er begge definert i klassen *String* og kan brukes hvis du ønsker det:

```
public String substring( int fra, int til )   // Trekk ut delstreng fra og med posisjon fra til posisjon til
public String substring( int fra )            // Trekk ut delstreng fra og med posisjon fra og ut
```

2-c) Skal kun besvares av de som har 4 timers sluttprøve

Du skal programmere metoden *hentTittel* i modellklassen *EpostTitler*. Metoden tar et heltall *nr* som parameter og skal returnere en bestemt tittel. Vi forklarer med et eksempel: Anta instansvariabelen *søkeord* er lik "Java" og at tabellen *tittelTab* har titler med "Java" i seg på posisjon 0, 2, 16 og 29. Hvis *nr* er lik 3 skal tittelen i posisjon 16 returneres (den er nummer 3 av de som inneholder søkeordet). Hvis *nr* er større enn 4 skal metoden returnere **null**. Følgende metode i klassen *String* kan være nyttig:

```
public int indexOf( String s )        // Gir posisjonen til første forekomst av s, eller -1 hvis s ikke finnes
```

Nynorsk

Nokre deloppgåver skal berre løysast av dei som har fritak for midtprøva / heimeoppgåva, og av den grunn får ei utvida sluttprøve. Desse deloppgåvene er merkte: "Skal berre løysast av dei som har 4 timars sluttprøve".

Vekting av deloppgåver i høve til endeleg karakter i kurset går fram av følgjande tabell:

Deloppgåve	**Vekting 3 timar**	**Vekting 4 timar**
1-a	10 %	10 %
1-b	10 %	10 %
1-c	Skal ikkje gjerast	10 %
2-a	20 %	20 %
2-b	20 %	20 %
2-c	Skal ikkje gjerast	30 %
Innlevering	20 %	Fritatt
Midtprøve	20 %	Fritatt

Les oppgåva nøye, og freist å svare på så mange deloppgåver som mogeleg. Lukke til!

Oppgåve 1

1-a)

Du skal lage ein metode for å lese inn heiltal frå brukar:

public static int lesHeiltal (int frå, int til)

Viss brukaren skriv noko som ikkje er eit lovlig heiltal frå og med parameteren *frå* og til parameteren *til* skal metoden gjenta innlesinga heilt til inndata er korrekt.

Innlesing skal gjerast ved hjelp av metoden *showInputDialog*, som er definert i klassa *JOptionPane*:

public static String showInputDialog(String ledetekst)

Denne metoden syner eit dialogvindauge med meldinga *ledetekst* og ein tekstboks der brukaren kan skrive inndata. Når brukaren trykkjer *enter* blir innhaldet i tekstboksen returnert.

1-b)

Du skal programmere ei såkalla krypteringsalgoritme:

public static void kryptere(int nøkkel, int[] data)

Alle verdiane i tabellen *data* skal vere eit av tala 0...255, og du kan tenkje deg at kvart tal representerer eit teikn (til dømes ein bokstav). Heile tabellen representerer dermed ein tekst. Parameteren *nøkkel* er eit positivt tal. Metoden skal oppdatere kvart element i tabellen ved å leggje til *nøkkel*, men resultatverdiane skal og vere blant tala 0...255. Tal over 255 må dermed "reduserast" ved at ein "startar på 0 att". Døme: Med *nøkkel* lik 2 skal verdi 0 bli til 2, 1 bli til 3, 2 bli til 4, ..., 253 bli til 255, 254 bli til 0 og 255 bli til 1.

1-c) Skal berre løysast av dei som har 4 timars sluttprøve

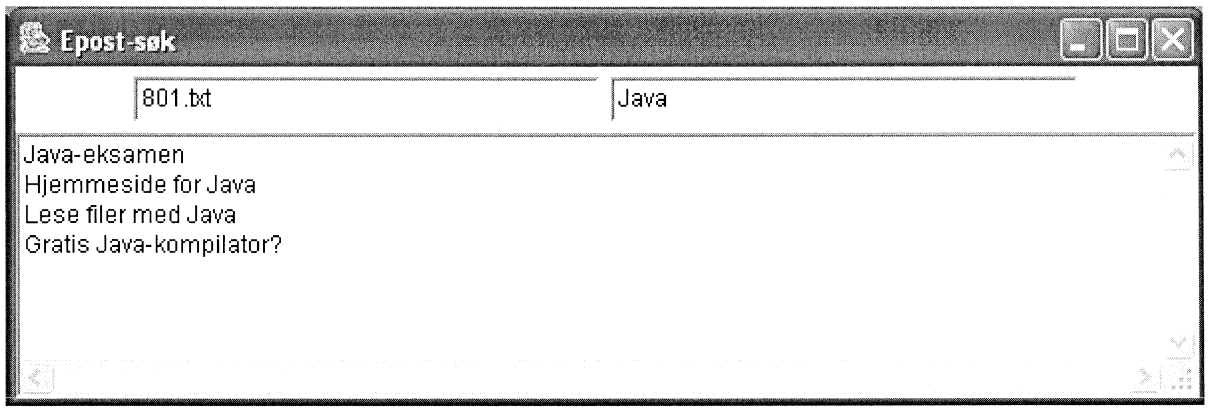
Du skal nå lage ei litt mer sofistikert krypteringsalgoritme.

```
public static void kryptere2( int[] nøkkel, int[] data )
```

Tabellen *data* inneheld verdiar blant tala 0...255, og vi kan tenkje oss at han representerer ein tekst som forklart i deloppgåve 1-b. Heiltalstabellen *nøkkel* inneheld alle verdiane 0...255 nøyaktig 1 gong, og rekkefølgja i denne tabellen bestemmer korleis verdiane i *data* skal krypterast: Alle verdiar X i tabellen *data* skal endrast til den verdien som er lagra på posisjon X i tabellen *nøkkel*. Viss til dømes data[20] inneheld 14 og nøkkel[14] inneheld 3, så skal data[20] bli satt til 3.

Oppgåve 2

Du skal lage utvalde delar av eit program for å søkje i tekstfiler som inneheld e-post'ane til ein brukar. Eit døme på ein e-postfil er vedlagt. Tittelen til ein e-post står på liner som startar med Subject, og er den einaste informasjonen frå fila som dette programmet skal nytte. Brukargrensesnittet skal sjå ut slik:



Brukaren skriv inn eit filnavn i venstre tekstboks og eit søkeord i høgre tekstboks. Det store tekstområdet skal heile tida syne dei titlane i fila som inneheld søkjeordet. Programmet skal byggjast opp frå ei brukargrensesnitt-klasse *EpostGUI* og ei modellklasse *EpostTitler*. Delar av begge klassene er lagt ved.

2-a)

Du skal programmere ferdig brukargrensesnittklassa *EpostGUI* (sjå vedlegg A), som inneber å definere konstruktøren og metoden *actionPerformed*. Få med kall på metodar i modellklassa!

2-b)

Du skal programmere metoden *lesFil* i klassa *EpostTitler* (sjå vedlegg B). Tittelen til ein e-post står på lina som startar med Subject (sjå vedlegg C). Metoden får namnet på fila som parameter, skal lese inn alle titlane som finst i fila, og lagre desse i tabellen *tittelTab* (instansvariabel i klassa). Følgjande to metodar er begge definert i klassa *String* og kan nyttast viss du ønskjer det:

```
public String substring( int frå, int til )  // Trekk ut delstreng frå og med posisjon frå til posisjon til
public String substring( int frå )           // Trekk ut delstreng frå og med posisjon frå og ut
```

2-c) Skal berre løysast av dei som har 4 timars sluttprøve

Du skal programmere metoden *hentTittel* i modellklassa *EpostTitler*. Metoden tar eit heiltal *nr* som parameter og skal returnere ein bestemt tittel. Vi forklarer med eit døme: Tenk at instansvariabelen *søkeord* er lik "Java" og at tabellen *tittelTab* har titlar med "Java" i seg på posisjon 0, 2, 16 og 29. Hvis *nr* er lik 3 skal tittelen i posisjon 16 returnerast (den er nummer 3 av dei som inneheld søkeordet). Hvis *nr* er større enn 4 skal metoden returnere **null**. Følgjande metode i klassa *String* kan vere nyttig:

```
public int indexOf( String s )        // Gir posisjonen til første førekomst av s, eller -1 viss s ikkje finst
```

Vedlegg A. Kodeskjelett til brukargrensesnittklassen

```
import java.awt.*;
import java.awt.event.*;

public class EpostGUI extends Frame implements ActionListener, WindowListener
{
	private TextField txtFilnavn, txtSøkeord;
	private TextArea txtTitler;
	private EpostTitler ET;

	public static void main( String[] args )
	{
		EpostGUI gui = new EpostGUI();
		gui.setSize( 600, 600 );
		gui.setVisible( true );
	}

	public EpostGUI()
	{
		// Kode til oppgave 2-a
	}

	public void actionPerformed( ActionEvent e )
	{
		// Kode til oppgave 2-a
	}

	public void windowClosing( WindowEvent e )
	{
		System.exit( 0 );
	}

	public void windowIconified( WindowEvent e ) { /* Skal være tom */ }
	public void windowOpened( WindowEvent e ) { /* Skal være tom */ }
	public void windowClosed( WindowEvent e ) { /* Skal være tom */ }
	public void windowDeiconified( WindowEvent e ) { /* Skal være tom */ }
	public void windowActivated( WindowEvent e ) { /* Skal være tom */ }
	public void windowDeactivated( WindowEvent e ) { /* Skal være tom */ }
}
```

Vedlegg B. Kodeskjelett til modellklassen

```
import java.io.*;

public class EpostTitler extends Object
{
        private String[] tittelTab; // Alle titlene; dvs teksten etter SUBJECT: i hver e-post
        private int nesteLedige;    // Neste ledige posisjon i tittelTab
        private String søkeord;     // Kun titler som inneholder søkeord blir vist

        public EpostTitler( int max )
        {
                tittelTab = new String[max];
                søkeord = null;
                tømTabell();
        }

        public boolean lesFil( String filnavn )
        {
                // Kode til oppgave 2-b
                // Leser inn data fra filen med navn filnavn, trekker ut epost-titlene og lagrer disse i tabellen
                // tittelTab, og setter nesteLedige til neste ledige posisjon i tabellen.
        }

        public void settSøkeord( String s )
        {
                søkeord = s;        // Setter søkeordet til tekststrengen s.
        }

        public String hentTittel( int nr )
        {
                // Kode til oppgave 2-c
                // Returnerer epost-tittel nummer nr blant de titlene som inneholder søkeordet.
                // Hvis det ikke finnes så mange "matchende" titler returnerer metoden null.
        }

        private void tømTabell()
        {
                for ( int i=0; i<tittelTab.length; i++ )
                        tittelTab[i] = "";
                nesteLedige = 0;
        }
}
```

Vedlegg C. Eksempel på e-postfil

Subject: Java-eksamen
To: Bjorn.Kristoffersen@hit.no
From: s123@student.hit.no
Date: Tue, 6 Apr 2004 11:48:55 +0100

Når er det eksamen?
Hilsen Lise
.....
Subject: Hva er pensum?
To: Bjorn.Kristoffersen@hit.no
From: s456@student.hit.no
Date: Tue, 13 Apr 2004 16:13:07 +0100

Heisann!
Vet du hvilke sider som er pensum? Og kan vi ha med boka?
- Ola
.....
Subject: Hjemmeside for Java
To: Bjorn.Kristoffersen@hit.no
From: s789@student.hit.no
Date: Tue, 13 Apr 2004 21:29:01 +0100

Hei Bjørn

Jeg prøver å finne siste versjon av Java-kompilatoren på
hjemmesiden til Java. Har du noen tips?

Mvh Jens
.....
Subject: Re: Re: Hva er pensum?
To: Bjorn.Kristoffersen@hit.no
From: s456@student.hit.no
Date: Tue, 13 Apr 2004 17:42:13 +0100

Hei igjen!

Betyr det at jeg kan ta med utskrift av programkode?

- Ola
.....
Subject: Repetisjon
To: Bjorn.Kristoffersen@hit.no
From: s012@student.hit.no
Date: Tue, 13 Apr 2004 23:59:57 +0100

Blir det repetisjonsforelesning om Java neste mandag?
Hilsen Anne
.....

OG SÅ VIDERE - RESTEN AV FILEN ER IKKE TATT MED