Institutt for datateknologi og informatikk

Eksamensoppgave i TDAT1001 Programmering Grunnkurs

Faglig kontakt under eksamen: Grethe Sandstrak og Bjørn Klefstad

Tlf.: 72 55 95 61 / 73 55 95 72

Eksamensdato: 1. des. 2017

Eksamenstid (fra-til): 09:00 – 13:00

Hjelpemiddelkode/Tillatte hjelpemidler: E / Ingen hjelpemidler tillatt

Annen informasjon:

Merk! Studenter finner sensur i Studentweb. Har du spørsmål om din sensur må du kontakte instituttet ditt. Eksamenskontoret vil ikke kunne svare på slike spørsmål.

**Problemdomene (kilde: www.snl.no)**

Definisjonsordbok: «En ordbok som gir definisjoner av oppslagsordene, kalles en definisjonsordbok. Gjelder den ett språk, er den enspråklig (monolingval).»

I dette eksamenssettet skal dere jobbe med å utvikle en applikasjon for å administrere en sterkt forenklet definisjonsordbok.

**Oppgave 1 – UML – modellering (10 %)**

**Oppgave 1 løses på utdelt ark**

1. Les gjennom hele oppgavesettet og sett opp UML- klassediagram som beskriver de tre klassene i oppgavesettet.
2. Sett opp UML - aktivitetsdiagram for programkoden i Vedlegg 1 – klientprogram. Du skal ikke ta med koden du legger inn som svar i oppgave 4.

**Oppgave 2 – klasen Ord (30 %)**

I denne oppgaven skal du lage klassen Ord. Et objekt av klassen Ord beskrives av et attributt for selve ordet og en tabell av tekststrenger som inneholder ordets ulike definisjoner. Tabellen med definisjoner skal ikke være større enn det antall definisjoner som til enhver tid er registrert . Når nye definisjoner legges til, skal tabellen utvides for å gjøre plass til den nye definisjonen (tabellen utvides fortløpende ved behov). Eksempel på et ord og dets definisjoner:

Ape:

|  |
| --- |
| "Dyr i jungelen" |
| "Herme" |
| "Vrøvle" |

1. Sett opp klassens objektvariabler.
2. Lag en konstruktør som gir verdi til klassens objektvariabler ihh til beskrivelsen gitt i oppgavebeskrivelsen.
3. Sett opp tilgangsmetoder for klassens objektvariabler.
4. Lag en metode for sjekke om to Ord-objekter er like. To ord er like dersom selve ordene er like. – du trenger dermed ikke sjekke om definisjonene er like.
5. Sett opp klassens toString()-metode. For ordet "Ape" skal metoden returnere følgende:

Ape:  
1. Dyr i jungelen

2. Herme

3. Vrøvle

1. Lag en privat hjelpemetode som utvider tabellen med ordets definisjoner med én plass.
2. Lag en metode – **public boolean leggTilDefinisjon(String nyDefinisjon)** -  for å legge til en ny definisjon til et ord. Ny definisjon kan ikke legges til dersom definisjonen er registrert fra før. I eksemplet over betyr dette at dersom en prøver å registrere ny definisjon for ordet Ape, så skal det ikke være mulig å legge til definisjonen "Dyr i jungelen", men det skal være mulig å legge til definisjonen "Dyr". Delvis likhet er ikke nok.

**Oppgave 3 - Klassen Ordbok**

Du skal nå jobbe videre med klassen Ordbok. Denne klassen beskrives av attributter for navn på ordboka og en tabell av Ord definert i oppgave 2.

Gitt følgende start på klassen Ordbok:

class Ordbok implements java.io.Serializable{

private String ordbokNavn;

private Ord[] ordbok;

private int antallReg;

private final int MAKS\_ANTALL\_ORD = 10;

private String filnavn = "ordliste.ser";.

1. Lag en konstruktør til klassen. Konstruktøren skal først sjekke om det finnes eventuelle data lagret på fila "ordliste.ser". Dersom data ikke finnes/ noe går galt med lesing fra fil, skal objektvariabelen ordbok opprettes med størrelse lik *MAKS\_ANTALL\_ORD* og *antallReg* skal være lik 0. Du trenger ikke legge inn mulighet for å endre på maks antall ord.  
   Du kan anta at metoden **public boolean lesOrdbokFraFil(String filnavn)** -  i oppgave f) eksisterer.
2. Lag en metode – **public boolean regNyttOrd (Ord ord)**, som registrerer et nytt ord i ordboka. Nytt ord kan kun registreres dersom det er plass i tabellen ordbok og dersom ordet ikke er registrert fra før.
3. Lag en metode – **public boolean leggTilDefinisjon (String ord, String definisjon)**, som legger til en ny definisjon av ordet. Du trenger ikke ta hensyn til om definisjonen er registrert fra før.
4. Lag en metode, **public Ord[] sorter ()**, som lager en kopi av objektvariabelen ordbok og returnerer kopien sortert alfabetisk på ord.
5. Lag en metode - **public Ord getOrd(String sokeStreng)**, som søker gjennom tabellen ordbok og sjekker om ord lik sokeStreng er registrert i tabellen. Dersom ordet finnes i tabellen skal det returneres. Dersom det ikke finnes skal metoden returnere verdien null.
6. Lag en metode - **public boolean lesOrdbokFraFil(String filnavn),** som leser en ord-tabell fra fil og oppdatere klassens objektvariabler (ordbok og antallReg) med innleste verdier.  Metoden skal returnere *true* om alt går bra, *false* dersom noe feiler.  
   Lag denne metoden på en slik måte at den håndterer eventuelle feil som oppstår.

**Oppgave 4 – klientprogram**

I vedlegg 1 finner du rammeverk til et menystyrt klientprogram som lar bruker

* Registrere nye ord i ordboka
* Legge til ny definisjon for et ord
* Avslutte

1. Lag koden for å registrere et nytt ord i ordboka
2. Lag koden for å registrere en ny definisjon for et ord

**Vedlegg 1 Klientprogram**

class Eksamen\_H2017{

public static void main(String[] args){

String[] muligheter = {"Legg til ord", "Legg til definisjon", "Avslutt"}

final int LEGG\_TIL\_ORD = 0;

final int LEGG\_TIL\_DEFINISJON = 1;

final int AVSLUTT = 2;

int valg = showOptionDialog(null, "Velg", "Eksamen des 2017", YES\_NO\_OPTION, INFORMATION\_MESSAGE, null, muligheter, muligheter[0]);

String navn = "Ordboka";

Ordbok ordbok = new Ordbok(navn, 10);

while (valg != AVSLUTT){

switch (valg){

case LEGG\_TIL\_ORD:

**/\* OPPGAVE 4a) skal inn her \*/**

break;

case LEGG\_TIL\_DEFINISJON:

**/\* OPPGAVE 4b) skal inn her \*/**

break;

default: break;

}

valg = showOptionDialog(null, "Velg", "Eksamen des 2017", YES\_NO\_OPTION, INFORMATION\_MESSAGE, null, muligheter, muligheter[0]);

}

}

}

**Vedlegg 2 – Utdrag Java API:   
FileInputStream. ObjectInputStream, String og JOptionPane**

java.io **Class FileInputStream**

* + - [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) [java.io.InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html) java.io.FileInputStream

**All Implemented Interfaces:**

[Closeable](https://docs.oracle.com/javase/8/docs/api/java/io/Closeable.html), [AutoCloseable](https://docs.oracle.com/javase/8/docs/api/java/lang/AutoCloseable.html" \o "interface in java.lang)

public class **FileInputStream** extends [InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html)

***Constructor Summary***

|  |
| --- |
| **Constructors** |
| [**FileInputStream**](https://docs.oracle.com/javase/8/docs/api/java/io/FileInputStream.html#FileInputStream-java.lang.String-)(**[String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang)** name)  Creates a FileInputStream by opening a connection to an actual file, the file named by the path name name in the file system. |

***Method Summary***

|  |  |
| --- | --- |
| **Modifier and Type** | **Method and Description** |
| |  |  | | --- | --- | | void |  | | [close](https://docs.oracle.com/javase/8/docs/api/java/io/ObjectInputStream.html#close--)()  Closes this file input stream and releases any system resources associated with the stream. |

***Constructor Detail***

**FileInputStream**

public FileInputStream([String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang) name) throws [FileNotFoundException](https://docs.oracle.com/javase/8/docs/api/java/io/FileNotFoundException.html)

Creates a FileInputStream by opening a connection to an actual file, the file named by the path name name in the file system. A newFileDescriptor object is created to represent this file connection. If the named file does not exist, is a directory rather than a regular file, or for some other reason cannot be opened for reading then aFileNotFoundException is thrown.

**Parameters:**

name - the system-dependent file name.

**Throws:**

[FileNotFoundException](https://docs.oracle.com/javase/8/docs/api/java/io/FileNotFoundException.html) - if the file does not exist, is a directory rather than a regular file, or for some other reason cannot be opened for reading.

[SecurityException](https://docs.oracle.com/javase/8/docs/api/java/lang/SecurityException.html) - if a security manager exists and its checkRead method denies read access to the file.

***Method Detail***

#### close

public void close()

throws [IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html)

Closes this file input stream and releases any system resources associated with the stream.

If this stream has an associated channel then the channel is closed as well.

**Throws:**

[IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html) - if an I/O error occurs.

java.io **Class ObjectInputStream**

* + - [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) [java.io.InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html) java.io.ObjectInputStream

public class **ObjectInputStream** extends [InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html)

implements [ObjectInput](https://docs.oracle.com/javase/8/docs/api/java/io/ObjectInput.html), [ObjectStreamConstants](https://docs.oracle.com/javase/8/docs/api/java/io/ObjectStreamConstants.html)

An ObjectInputStream deserializes primitive data and objects previously written using an ObjectOutputStream. Only objects that support the java.io.Serializable or java.io.Externalizable interface can be read from streams.

For example to read from a stream as written by the example in ObjectOutputStream:

FileInputStream fis = new FileInputStream("t.tmp");

ObjectInputStream ois = new ObjectInputStream(fis);

int i = ois.readInt();

String today = (String) ois.readObject();

Date date = (Date) ois.readObject();

ois.close();

***Constructor Summary***

|  |  |
| --- | --- |
| **Constructors** | |
|  | [**ObjectInputStream**](https://docs.oracle.com/javase/8/docs/api/java/io/ObjectInputStream.html#ObjectInputStream-java.io.InputStream-)(**[InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html" \o "class in java.io)** in)  Creates an ObjectInputStream that reads from the specified InputStream. |

***Method Summary***

|  |  |
| --- | --- |
| **Modifier and Type** | **Method and Description** |
| |  |  | | --- | --- | | void |  | | [close](https://docs.oracle.com/javase/8/docs/api/java/io/ObjectInputStream.html#close--)()  Closes the input stream. |
| [**Object**](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) | [**readObject**](https://docs.oracle.com/javase/8/docs/api/java/io/ObjectInputStream.html#readObject--)()  Read an object from the ObjectInputStream. |

***Constructor Detail***

**ObjectInputStream**

public ObjectInputStream([InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html" \o "class in java.io) in)

throws [IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html)

Creates an ObjectInputStream that reads from the specified InputStream. A serialization stream header is read from the stream and verified. This constructor will block until the corresponding ObjectOutputStream has written and flushed the header.

**Parameters:**

in - input stream to read from

**Throws:**

[StreamCorruptedException](https://docs.oracle.com/javase/8/docs/api/java/io/StreamCorruptedException.html) - if the stream header is incorrect

[IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html) - if an I/O error occurs while reading stream header

[SecurityException](https://docs.oracle.com/javase/8/docs/api/java/lang/SecurityException.html) - if untrusted subclass illegally overrides security-sensitive methods

[NullPointerException](https://docs.oracle.com/javase/8/docs/api/java/lang/NullPointerException.html) - if in is null

***Method Detail***

**readObject**

public final [Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) readObject()

throws [IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html), [ClassNotFoundException](https://docs.oracle.com/javase/8/docs/api/java/lang/ClassNotFoundException.html)

Read an object from the ObjectInputStream.

**Returns:**

the object read from the stream

**Throws:c** with a class used by serialization.

[StreamCorruptedException](https://docs.oracle.com/javase/8/docs/api/java/io/StreamCorruptedException.html) - Control information in the stream is inconsistent.

[OptionalDataException](https://docs.oracle.com/javase/8/docs/api/java/io/OptionalDataException.html) - Primitive data was found in the stream instead of objects.

[IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html) - Any of the usual Input/Output related exceptions.

#### close

public void close()

throws [IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html)

Closes the input stream. Must be called to release any resources associated with the stream.

**Specified by:**

[close](https://docs.oracle.com/javase/8/docs/api/java/io/Closeable.html#close--) in interface [Closeable](https://docs.oracle.com/javase/8/docs/api/java/io/Closeable.html" \o "interface in java.io)

**Throws:**

[IOException](https://docs.oracle.com/javase/8/docs/api/java/io/IOException.html) - If an I/O error has occurred.

java.lang **Class String**

* + [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) java.lang.String

**All Implemented Interfaces:**

[Serializable](https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html), [CharSequence](https://docs.oracle.com/javase/8/docs/api/java/lang/CharSequence.html" \o "interface in java.lang), [Comparable](https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html" \o "interface in java.lang)<[String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang)>

public final class **String**

extends [Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html)

implements [Serializable](https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html), [Comparable](https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html)<[String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang)>, [CharSequence](https://docs.oracle.com/javase/8/docs/api/java/lang/CharSequence.html)

***Method Summary***

|  |  |
| --- | --- |
| **Modifier and Type** | **Method and Description** |
| int | [**compareTo**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html#compareTo-java.lang.String-)(**[String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang)** anotherString)  Compares two strings lexicographically. |
| int | [**compareToIgnoreCase**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html#compareToIgnoreCase-java.lang.String-)(**[String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang)** str)  Compares two strings lexicographically, ignoring case differences. |
| int | [**length**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html#length--)()  Returns the length of this string. |

javax.swing **Class JOptionPane**

* + - * + [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) [java.awt.Component](https://docs.oracle.com/javase/8/docs/api/java/awt/Component.html) [java.awt.Container](https://docs.oracle.com/javase/8/docs/api/java/awt/Container.html) [javax.swing.JComponent](https://docs.oracle.com/javase/8/docs/api/javax/swing/JComponent.html) javax.swing.JOptionPane

|  |  |  |
| --- | --- | --- |
| **Modifier and Type** | | **Method and Description** |
| static **[String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html" \o "class in java.lang)** | [**showInputDialog**](https://docs.oracle.com/javase/8/docs/api/javax/swing/JOptionPane.html#showInputDialog-java.lang.Object-)(**[Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html" \o "class in java.lang)** message)  Shows a question-message dialog requesting input from the user. | |
| static void | [**showMessageDialog**](https://docs.oracle.com/javase/8/docs/api/javax/swing/JOptionPane.html#showMessageDialog-java.awt.Component-java.lang.Object-)([**Component**](https://docs.oracle.com/javase/8/docs/api/java/awt/Component.html) parentComponent, [**Object**](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) message)  Brings up an information-message dialog titled "Message". | |

