



KANDIDAT
10037

PRØVE

TDAT1001 A Programmering grunnkurs

Emnekode	TDAT1001
Vurderingsform	Skriftlig eksamen
Starttid	01.12.2017 08:00
Sluttid	01.12.2017 12:00
Sensurfrist	22.12.2017 00:00
PDF opprettet	22.11.2018 09:18

Oppgave		
Oppgave	Tittel	Oppgavetype
i	Forside	Dokument
i	Informasjon og Problembeskrivelse	Dokument
1	Oppgave 1 - UML-diagram	Muntlig
2	Oppgave 2- Klassen Ord	Programmering
3	Oppgave 3 - klassen Ordbok	Programmering
4	Oppgave 4 - Klientprogram	Programmering
i	Vedlegg 1 - Klientprogram	Dokument
5	Vedlegg 2 - Java API	Muntlig

1 **Oppgave 1 - UML-diagram**

Oppgave 1 løses på utdelt ark

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

Knytte håndtegninger til denne oppgaven?
Bruk følgende kode:

3 3 1 5 9 2 1

Question Code
Oppgavekode

Date
Dato

Subject code
Emnekode

Candidate ID
KandidatID

Question nr
Oppgavenr

Page number
Sidetall

3315921

01.12.17

TDAT1001_A

10037

1

1

00000000

11111111

22222222

33333333

44444444

55555555

66666666

77777777

88888888

99999999

a)

ORD

- STRING ORD

- STRING[] DEFINISJONER

+ORD (STRING ORD)

+STRING GETORD()

+STRING[] GETDEFINISJONER()

+BOOLEAN EQUALS (OBJECT O)

+STRING TO STRING()

-VOID UTVIDTABELL()

+BOOLEAN LEGGTILDEFINISJON (STRING NDEFINISJON)

+INT COMPARETO (ORD O)

ORDBOK

- STRING ORDBOKNAVN

- ORD[] ORDBOK

- INT ANTALLREG

- INT MAKS_ANT_ORD {read-only}

- STRING FILNAVN

+ORDBOK (STRING NAVN, INT MAKSANTORD)

+BOOLEAN REGNXTTORD (ORD ORD)

+BOOLEAN LEGGTILDEFINISJON (STRING ORD, STRING DEFINISJON)

+ORD[] SORTER()

+ORD GETORD (STRING SOMESTRING)

+BOOLEAN LESORDPÅFIL (STRING FILNAVN)

EKSAMEN_H2017

+MAIN (STRING[] ARGS)

-VOID REGNXTTORD ()

-VOID REGNXTDEF ()

SIDEN DET IKKE ER OPPGITT VELGER JEG Å HA KOMPOSISJON MELLOR ORDBOKEN OG ORD.

SIDEN DET IKKE ER NØDVENDIG VELGER JEG Å IKKE HA SET-METODER I ORD-KLASSEN.

I TILLEGG VELGER JEG Å LAGE ET UTVIDET KLASSE DIAGRAM

3/9

Question Code
Oppgavekode

3315921

Date
Dato

01.12.17

Subject code
Emnekode

TDAT1001-A

Candidate ID
KandidatID

10037

Question nr
Oppgavenr

1

Page number
Sidetall

2

0000000

1111111

2222222

3333333

4444444

5555555

6666666

7777777

8888888

9999999

b)

AKTIVITETS DIAGRAM AV MAN (SINIGD) ANG

SETTEN OPP RELEVANTE
VARIABLER. SETTER OPP
MULIGHETENE SOM EN
STRING-TABELL

LESER INN VALG FRA
BRUKER

SETTER NAVN OG OPP-
RETTER ORDBOK-OBJEKTET

FALSE

TRUE

VALG = AVSLUTT

VALG == 0

LEGGES TIL
ORD

VALG == 1

LEGGES TIL
DEFINISJON

DEFAULT

LESER INN VALG
FRA
BRUKER

2

Oppgave 2- Klassen Ord

Oppgave 2

4/9

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 en hver tid er registrert . Når nye definisjoner legges til, skal tabellen utvides for å gjøre plass til den nye definisjonen (tabellen utvides fortløpende etter behov). Eksempel på et ord og dets definisjoner:

Ape:

"Dyr i jungelen"
"Herme"
"Vrøvle"

- a. Sett opp klassens objektvariabler.
- b. Lag en konstruktør som gir verdi til klassens objektvariabler ihh til oppgavebeskrivelsen.
- c. Sett opp tilgangsmetoder for klassens objektvariabler.
- d. Lag en metode for å sjekke om to Ord-objekt er like. To ord er like dersom selve ordet er like. – du trenger dermed ikke sjekke om definisjonene er like.
- e. Sett opp klassens toString()-metode. For ordet "Ape" skal metoden returnere følgende:

Ape:

- 1. Dyr i jungelen
- 2. Herme
- 3. Vrøvle

- f. Lag en privat hjelpemetode som utvider tabellen med ordets definisjoner med én plass.
- g. 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 allerede er registrert. I eksempelet gitt over vil dette si at om 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 likt er ikke nok.

Skriv ditt svar her...

```
1  import java.io.*;
2
3  public class Ord implements Serializable{
4
5      private String ord;
6      private String[] definisjoner;
7
8      public Ord(String ord){
9          this.ord = ord;
10         definisjoner = new String[0];
11     }
12
13     public String getOrd(){return ord;}
14     public String[] getDefinisjoner(){return definisjoner;}
15     //Ser ikke nødvendigheten med å lage set-metoder, og lager derfor ikke det
16     /*Inkluderer fortsatt et eksempel på en set-metode hvis det faktisk var nødvendig
17     public void setOrd(String ord){this.ord = ord;}*/
18
19     public boolean equals(Object o){
20         //Sjekker om input er et ord
21         if(!(o instanceof Ord)){
22             return false; //Var ikke et Ord
23         }
24
25         //Sjekker om input refererer til det samme objektet
26         if(o == this){
27             return true;
28         }
29
30         //Setter opp Ord objekt av input
31         Ord input = (Ord)o;
32
33         //Sjekker om ordene er like
34         //Vil at ordene skal være like selv om at det er forskjeller mellom store og små bokstaver
35         if(ord.toLowerCase().equals(input.getOrd().toLowerCase())){
36             return true; //De er like
37         }
38
39         return false; //Ordene er ikke like
40     }
41     public String toString(){
42         String output = ord + ":";
43         for(int i = 0; i < definisjoner.length; i++){
44             output += "\n" + String.valueOf(i) + ". " + definisjoner[i];
45             //Bruker String.valueOf() bare for å vise at det er en funksjon som
```

```

46         }           konverterer input til en string, men er ikke nødvendig å bruke her
47         return output;
48     }
49
50     //Utvider definisjons-tabellen med 1
51     private void utvidTabell(){
52         //Lager en kopi som har en ekstra plass
53         String[] kopi = new String[definisjoner.length + 1];
54         for(int i = 0; i < definisjoner.length; i++){
55             kopi[i] = definisjoner[i];
56         }
57         definisjoner = kopi;
58     }
59
60     public boolean leggTilDefinisjon(String nyDefinisjon){
61         //Velger å ikke skille mellom store og små bokstaver
62         String nyDef = nyDefinisjon.toLowerCase();
63         //Sjekker om den nye definisjonen er allerede registrert
64         for(int i = 0; i < definisjoner.length; i++){
65             //Skal ikke skille mellom stor og små bokstaver
66             String def = definisjoner[i].toLowerCase();
67             if(def.equals(nyDef)){
68                 return false; //Er allerede registrert
69             }
70         }
71
72         //Utvider tabell
73         utvidTabell();
74         //Registrerer - legger til den nye definisjonen bakerst i den utvidete
75         //definisjons-tabellen.
76         definisjoner[definisjoner.length - 1] = nyDefinisjon;
77         return true; //Registrert
78     }
79
80     //En metode som sjekker om et annet ord er større, mindre, eller lik
81     public int compareTo(Ord o){
82         String inputOrd = o.getOrd();
83         int forskjell = ord.compareTo(inputOrd);
84         return (forskjell > 0)? 1 : (forskjell < 0)? -1 : 0;
85         //Kan også returnere bare forskjell, men viser at hvis "ord" er størst
86         //returner man vanligvis 1, -1 hvis mindre, og 0 hvis lik.
87     }
88 }
```

Knytte håndtegninger til denne oppgaven?

Bruk følgende kode:

3 1 2 7 6 2 9

3 Oppgave 3 - klassen Ordbok

Oppgave 3

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";
```

- a. 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 3 f) eksisterer.
- b. 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.
- c. Lag en metode – public boolean leggTilDefinisjon (String ord, String definisjon), som legger

- til en ny definisjon av ordet. Dersom det går greit å legge til en definisjon skal metoden returnere verdien true, dersom det ikke går greit skal metoden returnere verdien false.
- Du kan anta t metoden **public boolean leggTilDefinisjon(String nyDefinisjon)** fra oppgave 2g) eksisterer.
- d. Lag en metode, **public Ord[] sorter ()**, som lager en kopi av objektvariabelen ordbok og returnerer kopien sortert alfabetisk på ord.
- e. 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.
- f. 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.

Skriv ditt svar her...

```
1  import java.io.*;
2
3  public class Ordbok implements java.io.Serializable{
4      private String ordbokNavn;
5      private Ord[] ordbok;
6      private int antallReg;
7      private final int MAKS_ANTALL_ORD = 10;
8      private String filnavn = "ordliste.ser";
9
10     //Ifølge vedlegg 1 tas det også en int som paramenter i konstruktøren, og siden
11     //det ikke står noe spesifikt om å ha med dette antar jeg at det er
12     //MAKS_ANTALL_ORD det er snakk om.
13     public Ordbok(String ordbokNavn, int maksAntOrd){
14         this.ordbokNavn = ordbokNavn;
15         boolean status = lesOrdbokFraFil(filnavn); //Prøver å lese fra fil
16         MAKS_ANTALL_ORD = maksAntOrd;
17
18         //Hvis systemet ikke klarte å lese fra fil
19         if(!status){
20             ordbok = new Ord[MAKS_ANTALL_ORD];
21             antallReg = 0;
22         }
23     }
24
25     public boolean regNyttOrd(Ord ord){
26         //Sjekker om det er plass i tabellen
27         if(antallReg == ordbok.length || ord == null){
28             return false; //Ikke plass eller input er null
29         }
30
31         //Sjekker om ordet er registrert fra før
32         for(int i = 0; i < antallReg; i++){
33             if(ordbok[i].equals(ord)){
34                 return false; //Ordet er registrert fra før
35             }
36         }
37
38         //Registrerer
39         //Lager en dyp-kopi på grunn av prinsippet komposisjon
40         ordbok[antallReg] = new Ord(ord.getOrd());
41         antallReg++;
42         return true;
43     }
44
45     public boolean leggTilDefinisjon(String ord,String definisjon){
46         //Finner ordet i tabellen
47         Ord o = null;
48         for(int i = 0; i < antallReg; i++){
49             //Velger å implementere en mulighet å finne ordet selv om det er forskjell
50             //på
51             //små og store bokstaver
52             if(ordbok[i].getOrd().toLowerCase().equals(ord.toLowerCase())){
53                 o = ordbok[i]; //Fant ordet
54                 break;
55             }
56
57             //Sjekker om man fant ordet
58             if(o == null){
59                 return false; //Fant ikke ordet
60             }
61
62             //Prøver å registrere ny definisjon
63             return o.leggTilDefinisjon(definisjon);
64         }
65
66         public Ord[] sorter(){
67             //Sjekker om ordboken er null eller har ingen ord
68             if(ordbok == null || ordbok.length == 0){
69                 return null; //Ingenting å sortere
```

```
70         }
71     }
72     //Lager en kopi
73     Ord[] kopi = new Ord[antallReg];
74     for(int i = 0; i < kopi.length; i++){
75         kopi[i] = new Ord(ordbok[i].getOrd()); //Lager dyp-kopi på grunn av
            komposisjon
76     }
77
78     //Sorterer - bruker select-sort (n^2-algoritme)
79     for(int i = 0; i < kopi.length; i++){
80         int minst = i;
81         for(int k = i + 1; k < kopi.length; k++){
82             if(kopi[minst].compareTo(kopi[k]) > 0){
83                 minst = k; //kopi[k] er den minste hittil
84             }
85         }
86         //Bytter plass
87         Ord temp = kopi[minst];
88         kopi[minst] = kopi[i];
89         kopi[i] = temp;
90     }
91     return kopi;
92     //Det er også mulig å bruke Arrays.sort(kopi), men det viser ikke forståelse
93     //for compareTo-metoden
94 }
95
96 public Ord getOrd(String sokeStreng){
97     //Finner ordet i tabellen
98     sokeStreng = sokeStreng.toLowerCase();
99     for(int i = 0; i < antallReg; i++){
100         //Velger å ikke skille mellom store og små bokstaver
101         //Bruker .toLowerCase() for å oppnå dette
102         if(ordbok[i].getOrd().toLowerCase().equals(sokeStreng)){
103             return new Ord(ordbok[i].getOrd()); //Fant ordet
104             //Returnerer en dyp-kopi på grunn av komposisjon
105             //Vil bare at ordbok-objektet skal håndtere ordene sine
106         }
107     }
108
109     return null; //Fant ingen ord
110 }
111
112 public boolean lesOrdbokFraFil(String filnavn){
113     //Bruker try-catch syntaks for å håndtere eventuelle feil
114     try{
115         FileInputStream fs = new FileInputStream(filnavn);
116         ObjectInputStream os = new ObjectInputStream(fs);
117         Ord[] ob = (Ord[])os.readObject();
118         os.close();
119         ordbok = ob; //Setter orbok lik den leste ord-tabellen
120         //Sjekker om ordboken inneholder null
121         //Kan hende at tabellen fra filen inneholder objekter som er lik null
122         antallReg = 0;
123         for(int i = 0; i < ordbok.length; i++){
124             if(ordbok[i] != null){
125                 antallReg++;
126             }
127         }
128         return true;
129     }
130     catch(IOException | ClassNotFoundException e){
131         e.printStackTrace();
132         System.out.println("Feil under lesing av filen!");
133         return false;
134         //IOException viser til en feilen der filen ikke eksisterer (og mye mer)
135         //ClassNotFoundException viser til en feil der lesingen av objektet
136         //ikke er en Ord[].
137         //Det er også mulig å skrive "catch(Exception e)" siden Exception viser til
138         //all form av feil.
139     }
140 }
141 }
```

Knytte håndtegninger til denne oppgaven?
Bruk følgende kode:

8797309

4 Oppgave 4 - Klientprogram

Oppgave 4
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

- a. Lag koden for å registrere et nytt ord i ordboka
- b. Lag koden for å registrere en ny definisjon for et ord

Skriv ditt svar her...

```
1  import static javax.swing.JOptionPane.*;
2
3  public class Eksamen_H2017{
4      public static void main(String[] args){
5          String[] muligheter = {"Legg til ord","Legg til definisjon","Avslutt"}
6          final int LEGG_TIL_ORD = 0;
7          final int LEGG_TIL_DEFINISJON = 1;
8          final int AVSLUTT = 2;
9          int valg = showOptionDialog(null,"Velg","Eksamen des 2017",YES_NO_OPTION
10             ,INFORMATION_MESSAGE,null,muligheter,muligheter[0]);
11          String navn = "Ordboka";
12          Ordbok ordbok = new Ordbok(navn,10);
13
14          while(valg != AVSLUTT){
15              switch(valg){
16                  case LEGG_TIL_ORD:
17                      regNyttOrd(ordbok); //Lager metoder i stedet for å skrive inn direkte
18                      //Velger å gjøre det for ryddighetens skyld
19                      break;
20                  case LEGG_TIL_DEFINISJON:
21                      regNyDefinisjon(ordbok); //Registrerer ny definisjon
22                      break;
23                  default:
24                      break;
25              }
26              int valg = showOptionDialog(null,"Velg","Eksamen des 2017",YES_NO_OPTION
27                  ,INFORMATION_MESSAGE,null,muligheter,muligheter[0]);
28          }
29
30          //Klientprogrammet som registrerer nytt ord
31          private static void regNyttOrd(Ordbok ordbok){
32              String ord = showInputDialog("Skriv inn det nye ordet");
33              Ord o = new Ord(ord);
34              boolean status = ordbok.regNyttOrd(o); //Prøver å registrere det nye ordet
35              String message = (status)? "Ordet ble registrert!" : "Feil! Kunne ikke
36                  registrere ordet!";
37              showMessageDialog(null,message); //Tilbakemelding
38
39          //Klientprogrammet som registrerer ny definisjon
40          private static void regNyDefinisjon(Ordbok ordbok){
41              String ord = showInputDialog("Skriv inn et ord du skal ha en ny definisjon for"
42                  );
43              String definisjon = showInputDialog("Skriv inn en ny definisjon");
44              boolean status = ordbok.leggTilDefinisjon(ord,definisjon); //Prøver å registrere
45                  definisjonen
46              String message = (status)? "Ny definisjon ble registrert!" : "Kunne ikke
47                  registrere ny definisjon";
48              showMessageDialog(null,message); //Tilbakemelding
49          }
50      }
```

Knytte håndtegninger til denne oppgaven?

Bruk følgende kode:

1975612

5

Vedlegg 2 - Java API

Vedlegg 2 - Java API

Knytte håndtegninger til denne oppgaven?

Bruk følgende kode:

9923471