UNIVERSITETET I OSLO

KANDIDAT

15477

PRØVE

# IN1010 1 Objektorientert programmering

| | |
|---|---|
| Emnekode | IN1010 |
| Vurderingsform | Individuell skriftlig prøve |
| Starttid | 04.06.2025 13:00 |
| Sluttid | 04.06.2025 17:00 |
| Sensurfrist | -- |
| PDF opprettet | 27.06.2025 06:17 |

**Seksjon 1**

| Oppgave | Tittel | Oppgavetype |
|---|---|---|
| ℹ | Informasjon | Informasjon eller ressurser |
| 1 | Oppgave 1 | Programmering |
| 2 | Oppgave 2 | Programmering |
| 3 | Oppgave 3 (tegning) | Langsvar |
| 4 | Oppgave 4 | Programmering |
| 5 | Oppgave 5 | Programmering |
| 6 | Oppgave 6 | Programmering |
| 7 | Oppgave 7 | Programmering |
| 8 | Oppgave 8 | Programmering |
| 9 | Oppgave 9 | Programmering |
| 10 | Oppgave 10 | Programmering |

# 1 Oppgave 1

### Skriv ditt svar her

```java
//Har endret litt på rekkefølgen, altså strukturen i klassen
class Samling<E>{
    private class Node {
        E data;
        Node neste;

        Node (E data){
            this.data = data;
            neste = null;
        }
    }

    private final Node START;       //Antar at START er final (konstant), da den

    public Samling (int n){
        START = new Node(null);
        Node peker = START;
        for (int i = 0; i < n; i ++){
            Node ny = new Node(null);
            peker.neste = ny;
            peker = peker.neste;        //Kunne også ha skrevet peker = ny
        }
    }

    //Antar at det alltid finnes en ledig Node
    public void settInn(E ny){
        Node peker = START.neste;           //Starter fra noden etter START
        //Iterer meg gjennom til den riktige noden
        while (peker.data != null){
            peker = peker.neste;
        }
        //Nå har vi kommet til en Node som har datareferanse null, og data settes
        peker.data = ny;
    }

    public void fjern (E ref) { /* Skal ikke skrives */}
}
```

Maks poeng: 10

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**0 2 4 5 4 5 9**

# 2 Oppgave 2

### Skriv ditt svar her

```java
//Har bare kopiert koden fra Oppgave 1 hit
class Samling<E> implements Iterable<E>{
    private class Node {
        E data;
        Node neste;

        Node (E data){
            this.data = data;
            neste = null;
        }
    }

    private final Node START;       //Antar at START er final (konstant), da den

    public Samling (int n){
        START = new Node(null);
        Node peker = START;
        for (int i = 0; i < n; i ++){
            Node ny = new Node(null);
            peker.neste = ny;
            peker = peker.neste;        //Kunne også ha skrevet peker = ny
        }
    }

    //Antar at det alltid finnes en ledig Node
    public void settInn(E ny){
        Node peker = START.neste;
        //Iterer meg gjennom til den riktige noden
        while (peker.data != null){
            peker = peker.neste;
        }
        //Nå har vi kommet til en Node som har datareferanse null
        peker.data = ny;
    }

    public void fjern (E ref) { /* Skal ikke skrives */}

    /* Oppgave 2 */
    @Override
    public Iterator<E> iterator(){
        return new ListeIterator();
    }

    class ListeIterator implements Iterator<E> {
        Node denne = START.neste;

        @Override
        public boolean hasNext (){
            return denne.data != null;
        }

        @Override
        public E next(){
            if (!hasNext()){
                throw new NoSuchElementException();
            }
            E midlertidig = denne.data;
            denne = denne.neste;
            return midlertidig;
        }
    }
}
```

Maks poeng: 10

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**5 2 6 4 6 0 6**

# 3 Oppgave 3 (tegning)

*I denne oppgaven kan du svare med digital tegning i svarfeltet under, eller tegne for hånd på utdelte skisseark. Se instruksjon for utfylling av skisseark i lenken under oppgavelinjen.*

Tegnet for hånd.

Ord: 3

Maks poeng: 8

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

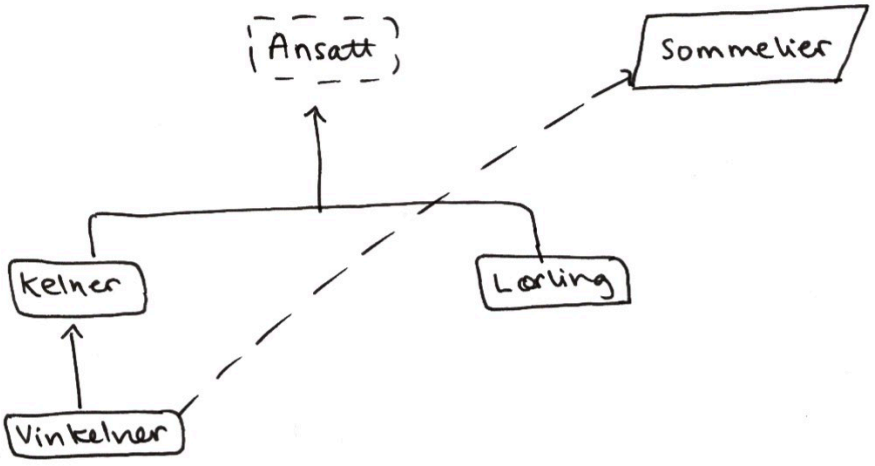
**0 8 1 7 8 5 9**

Håndtegning 1 av 1

**Fill out Question Code and Test Information on every sheet.** Fyll inn oppgavekode og emneinformasjon på alle skisseark.

| Question Code<br>Oppgavekode | Date<br>Dato | Subject code<br>Emnekode | Candidate ID<br>KandidatID | Question nr<br>Oppgavenr | Page number<br>Sidetall |
|---|---|---|---|---|---|
| 0 8 1 7 8 5 9 | 04.06.25 | IN1010 | 15477 | 3 | 1 |

**Drawing area** Tegneområde




Antar at Ansatt er en abstrakt klasse, da alle ansatt enten er kelnere eller lærlinger.

# 4 Oppgave 4

### Skriv ditt svar her

```java
abstract class Ansatt {
    final String NAVN;            //Antar at det er konstant

    //Antar at navn er en parameter til konstruktøren til klassen
    public Ansatt (String navn){
        NAVN = navn;
    }
}

class Lærling extends Ansatt {
    public Lærling (String navn){
        super(navn);
    }
}

class Kelner extends Ansatt {
    Samling<Bord> bord = new Samling<>(10);     //Legger til en Samling på 10 bor
        denne skal tas inn som parameter til konstruktøren eller ikke -> så den i
    public Kelner (String navn){
        super(navn);
    }
}

interface Sommelier {
    String spesialitet();
}

class Vinkelner extends Kelner implements Sommelier {
    final String SPESIALITET;

    public Vinkelner (String navn, String spesialitet){
        super(navn);
        SPESIALITET = spesialitet;
    }

    @Override
    public String spesialitet(){
        return SPESIALITET;
    }
}

class Bord {
    final int ANTALL_PLASSER;
    //kelner og gruppe er null når bordet ikke er i bruk
    Kelner kelner = null;
    Gjestgruppe gruppe = null;
    //Tar med en boolsk-verdi for å sjekke om bord er opptatt eller ikke
    boolean ledig = true;            //Setter den true i starten

    public Bord (int antall){
        ANTALL_PLASSER = antall;
    }
}

class Gjest {
    boolean ønskerVin;

    public Gjest (boolean ønskerVin){
        this.ønskerVin = ønskerVin;
    }
}

class Gjestegruppe {
    Samling<Gjest> gjester = new Samling<>(20);
```

```
64      //Antar at man bruker settInn()-metoden fra Samling for å sette inn og dermed
            20 gjester -> ingenting er nevnt i oppgaveteksten
65  }
66
67  class Restaurant {
68      Samling<Bord> alleBord = new Samling<>(15);
69      Samling<Ansatte> ansatte = new Samling<>(25);
70  }
```

Maks poeng: 10

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**3 5 8 7 4 4 2**

# 5 Oppgave 5

**Skriv ditt svar her**

```
1   //Kopierer deler av koden fra oppgave 4 inn her
2   class Gjest {
3       boolean ønskerVin;
4
5       public Gjest (boolean ønskerVin){
6           this.ønskerVin = ønskerVin;
7       }
8
9       public boolean ønskerVin(){
10          return ønskerVin;           //Det har ikke noe å si om metoden og instans
11      }
12  }
13
14  class Gjestegruppe {
15      Samling<Gjest> gjester = new Samling<>(20);
16      //Antar at man bruker settInn()-metoden fra Samling for å sette inn og dermed
            20 gjester -> ingenting er nevnt i oppgaveteksten
17
18      public boolean noenØnskerVin(){
19          for (Gjest g : gjester){
20              if (g.ønskerVin()){
21                  return true;
22              }
23          }
24
25          //Ellers retunerer metoden false -> ingen som ønsker vin i gruppa
26          return false;
27      }
28  }
```

Maks poeng: 8

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**0 1 6 9 3 6 9**

# 6 Oppgave 6

### Skriv ditt svar her

```java
//Kopiere deler av koden fra oppgavee 4
class Restaurant {
    Samling<Bord> alleBord = new Samling<>(15);
    Samling<Ansatte> ansatte = new Samling<>(25);

    public Kelner finnMinstÅGjøre (boolean erVinkelner){
        int minst = Math.MAX_VALUE;  //Setter det til et stort tallet, er litt us
            den henter en stor verdi som jeg bruker for å sammenligne.
        Kelner minstÅGjøre = null;

        //Kunne ha skrevet en hjelpemetode, da det er kode som skrives om igjen,
        if (!erVinkelner){      //Her er vi interessert i vanlige Kelnere
            for (Ansatt a : ansatte){
                if (ansatt instanceof Kelner && !ansatt instanceof Sommelier){
                    int antall = 0;
                    Kelener kelner = (Kelner) a;
                    for (Bord b : kelner.bord){
                        antall += b.gruppe.gjester.antall();  //bruker antall()-m
                            oppgave 7
                    }
                    if (antall < minst){
                        minst = antall;
                        minstÅGjøre = kelner;
                    }
                }
            }
        }
        else {      //Her er kun Vinkelner-e interessante
            for (Ansatt a : ansatte){
                if (a instanceof Vinkelner){ //eller a instanceof Sommelier
                    int antall = 0;
                    Vinkelner vinkelner = (Vinkelner) a;
                    for (Bord b : vinkelner.bord){
                        antall += b.gruppe.gjester.antall();  //bruker antall()-m
                    }
                    if (antall < minst){
                        minst = antall;
                        minstÅGjøre = vinkelner;
                    }
                }
            }
        }

        return minstÅGjøre;
    }
}
```

Maks poeng: 8

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**7 2 3 6 9 9 0**

# 7 Oppgave 7

### Skriv ditt svar her

```java
//Har endret litt på rekkefølgen, altså strukturen i klassen
class Samling<E>{
    private class Node {
        E data;
        Node neste;

        Node (E data){
            this.data = data;
            neste = null;
        }
    }

    private final Node START = new Node(null);        //Antar at START er final (
        samme

    public Samling (int n){
        Node peker = START;
        for (int i = 0; i < n; i ++){
            Node ny = new Node(null);
            peker.neste = ny;
            peker = peker.neste;        //Kunne også ha skrevet peker = ny
        }
    }

    //Antar at det alltid finnes en ledig Node
    public void settInn(E ny){
        Node peker = START.neste;
        //Iterer meg gjennom til den riktige noden
        while (peker.data != null){
            peker = peker.neste;
        }
        //Nå har vi kommet til en Node som har datareferanse null
        peker.data = ny;
    }

    public void fjern (E ref) { /* Skal ikke skrives */}


    /* Oppgave 7 */
    public int antall (){
        if (START.neste.data == null){   //Ingen elementer i listen
            return 0;
        }
        return antallHjelper (START.neste, 0);        //Starter fra den noden ett
            lista
    }

        //Skriver en privat rekursiv hjelpemetode
    private int antallHjelper (Node n, int ant){
        //Basistilfellet
        if (n.data == null){
            return ant;
        }
        ant ++;
        return antallHjelper(n.neste, ant);     //Rekursive steget
    }
}
```

Maks poeng: 10

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**4 8 4 6 2 2 3**

# 8 Oppgave 8

**Skriv ditt svar her**

```
class Restaurant {
    Samling<Bord> alleBord = new Samling<>(15);
    Samling<Ansatte> ansatte = new Samling<>(25);

    public void taImotGjester (Gjestegruppe gruppe){
        //Her aksessere jeg alt direkte, men kunne også hatt hentmetoder og settm
        int antallGjester = gruppe.gjester.antall();
        Bord ledigBord = finnLedigBord(antallGjester);
        boolean ønskerVin = gruppe.noenØnskerVin();
        Kelner kelner = finnMinstÅGjøre(ønskerVin);
        kelner.bord.settInn(ledigBord);
        ledigBord.kelner = kelner;
        ledigBord.gruppe = gruppe;
        ledigBord.ledig = false;
    }

    //hjelpemetode for å finne et ledig bord
        //bruker iteratoren og returnere et ledig bord med nok plasser og som er
    //står at man alltid kan anta at det finnes et ledig bord
    private Bord finnLedigBord (int antallGjester){
        Bord ledigBord = null;
        for (Bord b : alleBord){
            if (b.ledig && ANTALL_PLASSER >= antallGjester){
                ledigBord = b;
                break;      //ledig bor er funnet og går ut av løkken
            }
        }
        return ledigBord;
    }
}
```

Maks poeng: 12

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**4 1 4 8 5 9 9**

# 9 Oppgave 9

### Skriv ditt svar her

```
class Restaurant {
    Samling<Bord> alleBord = new Samling<>(15);
    Samling<Ansatte> ansatte = new Samling<>(25);

    //Instansvariabler for å lage GUI-vinduet
    JFrame vindu;
    JPanel panel;
    JLabel tekst;
    JButton knapp;

    public void taImotGjester (Gjestegruppe gruppe){
        int antallGjester = gruppe.gjester.antall();
        Bord ledigBord = finnLedigBord(antallGjester);
        //Dersom bord er null, så vises  GUI-vinduet
        if (ledigBord == null){
            lagVindu();
        }
        else{
            boolean ønskerVin = gruppe.noenØnskerVin();
            Kelner kelner = finnMinstÅGjøre(ønskerVin);
            kelner.bord.settInn(bord);
            ledigBord.kelner = kelner;
            ledigBord.gruppe = gruppe;
            ledigBord.ledig = false;
        }
    }


    private Bord finnLedigBord (int antallGjester){
        Bord ledigBord = null;
        for (Bord b : alleBord){
            if (b.ledig && ANTALL_PLASSER >= antallGjester){
                ledigBord = b;
            }
        }
        return ledigBord;       //Dersom det ikke er noen ledig bord, så er den r
    }

    //Lager en hjelpemetode for å sette opp vinduet som kalles dersom det ikke fi
        hjelpemetoden finnLedigBord returnere null)
    private void lagVindu(){
        try{
            UIManager.setLookAndFeel(UIManager.setCrossPlatfromLookAndFeelClassNa
        }
        catch (Exception e){
            System.exit(1);
        }

        vindu = new JFrame("Melding");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        panel = new JPanel();
        tekst = new JLabel("Beklager, restauranten er full!");
        knapp = new JButton("OK");

        class Avslutt implements ActionListener {
            @Override
            public void actionPerformed(ActionEvent ae){
                System.exit(0);     //Avslutter uten feil
            }
        }

        knapp.addActionListener(new Avslutt());
```

```
64          panel.add(tekst);
65          panel.add(knapp);
66
67          vindu.add(panel);
68
69          vindu.pack();
70          vindu.setLocationRelativeTo(null);
71          vindu.setVisible(true);
72      }
73  }
```

---

Maks poeng: 10

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**5 2 5 4 4 9 5**

# 10 Oppgave 10

### Skriv ditt svar her

```
//Monitor klassen
class Restaurant {
    Samling<Bord> bord = new Samling<>(15);     //Beholder
    Samling<Ansatte> ansatte = new Samling<>(25);

    //Instansvariabler for å håndtere synkronisering i monitoren
    Lock lås = new ReentrantLock(true);
    Condition ventPåLedig = lås.newConditon();

    //Metoden returnere her det bordet gjesten har fått utdelt -> slik at det ka
        dette har jeg kun gjort for denne oppgaven. I de tidligere oppgavene ret
        har en void som returtype
    public Bord taImotGjester (Gjestegruppe gruppe){
        lås.lock();

        try{
            int antallGjester = gruppe.gjester.antall();
            Bord ledigBord = finnLedigBord(antallGjester);
            while (ledigBord == null){
                ventPåLedig.await();
            }
            //nå er det et ledig bord -> gruppen blir plassert på bordet og får
            boolean ønskerVin = gruppe.noenØnskerVin();
            Kelner kelner = finnMinstÅGjøre(ønskerVin);
            kelner.bord.settInn(ledigBord);
            ledigBord.kelner = kelner;
            ledigBord.gruppe = gruppe;
            ledigBord.ledig = false;
            return ledigBord;
        }
        catch (InterruptedException e){
            return null;
        }
        finally {
            lås.unlock();
        }
    }

    public void forlatBordet (Bord bord){
        lås.lock();

        try{
            bord.gjestegruppe = null;
            bord.kelner = null;
            bord.ledig = true;
            ventPåLedig.signalAll();    //står at man skal "signalere ventende g
                å bruke signal() da den som har ventet lengst får et bord først,
                Condition-køen (ventPåLedig) ->  her har jeg valgt å følge det o
        }
        finally{
            lås.unlock();
        }
    }

    //Hjelpemetode som brukes i taImotGjester()
    private Bord finnLedigBord (int antallGjester){
        Bord ledigBord = null;
        for (Bord b : bord){
            if (b.ledig && ANTALL_PLASSER >= antallGjester){
                ledigBord = b;
            }
        }
        return ledigBord;
    }
```

```
61  }
62
63  //Trådklassen
64  class Restaurantbeøsk implements Runnable{
65      Restaurant restaurant;
66      Gjestegruppe gruppe;
67
68      public Restaurantbesøk (Restaurant r, Gjestegruppe g){
69          restaurant = r;
70          gruppe = g;
71      }
72
73      @Override
74      public void run(){
75          Bord b = restaurant.taImotGjester(gruppe);      //Lagrer referansen til
                den kan brukes i forlatBordet()-metoden
76          //Kunne også hatt Thread.sleep(5000) for å vente på at gjestene spiser f
                realistisk)
77          restaurant.forlatBordet(b);
78      }
79  }
```

---

Maks poeng: 14

**Knytte håndtegninger til denne oppgaven?**
Bruk følgende kode:

**5 8 4 6 0 9 0**