Høgskolen i Telemark

EKSAMEN

5610 ALGORITMAR OG DATASTRUKTURAR

06.05.2011

Tid:	9-14 (5 timar)
Målform:	Bokmål og nynorsk
Sidetal:	9 (forside + 4 + 4)
Hjelpemiddel:	Alle trykte og skrivne
Vedlegg:	Ingen

Eksamensresultata blir offentliggjort på nettet via Studentweb

Råd og retningslinjer. Les oppgaveteksten godt før du går i gang med å løse oppgava. Deloppgavene er uavhengige av hverandre i den forstand at om du ikke får til en oppgave, kan du likevel gjøre neste, som om den første var løst. Fordél tida godt på alle oppgavene. Om du mener en oppgave er upresis, så skriv din egen presisering.

Oppgave 1: Liste

1a (30%)

For et par tiår siden var det ikke uvanlig å lage pekerkjeder på følgende måte. En Node-klasse inneholdt bare nestepeker. Liste-klassa inneholdt førstepeker og evt. sistepeker av type Node. De klassene som skulle legges i liste måtte lages som subklasse av Node-klassa, se f. eks. Person-klassa i eksempelet under. Man kan ennå treffe på slik kode som må vedlikeholdes.

```
public class Node { Node neste; }
public class Liste {
  private Node første, siste;
  private int antall;
  ...
}
class Person extends Node { ... }

Liste l = new Liste();
l.settInnFørst(new Person("Ola Nordmann", ...));
```

Du kan anta at Node og Liste ligger i same pakke. Bruk class Node slik den er over, og lag class Liste med følgende metoder:

```
public Liste()                      // Konstruktørmetode
public void settInnFørst(Node n)    // Sett inn objektet først i lista
public void settInnSist(Node n)     // Sett inn objektet sist i lista
public Node finn(int i)             // Returner objekt nr. i dersom det
                                    // finnes, start telling på 0
public Node slett(int i)            // Slett og returner objekt nr. i
public boolean erTom()              // Returner true dersom tom liste
public void gjørTom()               // Gjør lista tom
public int antall()                 // Antall objekt i lista
```

Liste skal altså ha første- og sistepeker, og en antall-variabel. Du skal ikke bruke listehode og -hale. Du skal skrive klassa fra grunnen av, uten å bruke noe i Collections API'et.
Nevn både fordeler og ulemper med dette systemet i forhold til dagens.

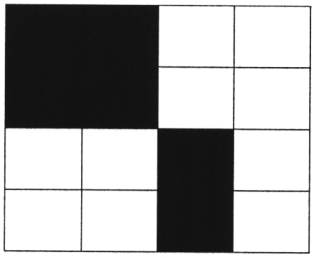
1b (10%)

Lag class ListeIterator som iterator-klasse for Liste-klassa over, også den ligger i same pakke. Den skal implementere Iterator-grensesnittet i Collections-API'et, den må altså ha hasNext() og next(). Klassa må også ha remove(), men den metoden skal ikke ha noen funksjon.
Lag også iterator()-metode i class Liste.

1c (5%)
Lag metoden toString() i class Liste. Den skal returnere en String med alle objektene kommaseparert og omsluttet av [ ].

Oppgave 2: Quadtre

Et ***quadtre*** er et tre der hver node har null eller fire barn. Vi skal her bruke quadtre til å representere bilde med bare svart og hvitt, for eksempel et dokument. Hver node i treet representerer et kvadratisk område i bildet. Rotnoden representerer hele bildet, dens fire barn representerer hver sin fjerdepart av bildet osv. Dersom området som en node representerer er homogent, dvs helt hvitt eller helt svart, skal noden ikke ha barn. Vi sier da at noden er hvit (svart). Dersom området er inhomogent, dvs. inneholder både svart og hvitt, skal noden ha fire barn, vi sier at noden er grå. Vi begrenser oss til bilder som er kvadratiske, og der størrelsen er en toerpotens, f.eks. 256x256, 512x512, 1024x1024.



Figur 1: Bilde på 4x4. Kvadrant én opp til venstre er homogent svart og trenger ikke splittes. Kvadrant to opp til høyre og tre ned til venstre er homogent hvite og trenger ikke splittes. Kvadrant fire ned til høyre er inhomogen og må ha fire barn.

2a (5%)
Tegn QuadTreet som representerer bildet i figur 1. Sett en S inni svarte noder, en H inni hvite, og en G i grå/inhomogene noder. Tegn barna i rekkefølge 1..4 slik forklart i figur 1.

Følgende uferdige klasser er laget. Et QuadTre-objekt representerer et helt bilde. Et Piksel-objekt representerer et homogent område. IndreNode representerer et inhomogent område. Node er abstrakt superklasse for Piksel og IndreNode.

```
public class QuadTre {
  Node rot;
  int høyde; // Maksimal høyde på treet

  public float verdi() { ... }
  public int størrelse() { ... }
  public void roter() { ... }
  public void rydd() { ... }
}
```

Variablen høyde forteller hvor høyt treet maksimalt kan være. Dermed gir den også størrelsen på bildet. Dersom høyde er 3 har vi et 8x8 bilete. Treets faktiske høyde kan være mindre. Dersom bildet er helt homogent vil faktisk høgde være 0, treet har bare rotnode.

```
abstract class Node {
  float verdi;
  float verdi() { return verdi; }
  abstract float beregnVerdi();
}
```

Nodens verdi er på skalaen 0..1 der 0 er svart og 1 er hvit.

```
class Piksel extends Node {
  float beregnVerdi() {
    return verdi;
  }
}
```

Et piksel er her et homogent område med pikselverdi 0 (svart) eller 1 (hvit).

```
class IndreNode extends Node {
  Node ov, oh, nv, nh;

  float beregnVerdi() { ... }
}
```

En indre node er et inhomogent område som er delt i fire:
ov – opp opp til venstre
oh – opp til høyre
nv – ned til venstre
nh – ned til høyre

2b (10%)
Metoden verdi() i class QuadTre skal returnere gjennomsnittsverdien i biletet, verdien er lagret i rotnoden. Metoden størrelse() i class QuadTre skal returnere størrelsen på biletet, f.eks. 8 når høyde er 3. Metoden beregnVerdi() i class IndreNode skal – rekursivt nedover i treet – beregne verdi til noden som gjennomsnittet av de fire barna, legge verdien i verdi-variablen, og returnere verdien. Du kan anta at alle IndreNode-objekter har fire barn, ingen er null.
Lag disse tre metodene.

2c (10%)
Når et bilde skal roteres 90 grader mot høyre, er det bare å flytte alle kvadrantane 90 grader mot høyre (med klokka) på alle nivå nedover i treet. Kvadranten opp til venstre skal plasseres opp til høyre, osv. Lag metoden
`public void roter()`
i class QuadTre som utfører dette. Lag de hjelpemetoder du måtte ønske i de klasser du måtte ønske.

2d (15%)
For framvisning av slike bilder er det laget egen framvisningsklasse:

```
public class QTDisplay {
  private QuadTre qt;
  private Graphics g;

  public QTDisplay(QuadTre qt,
      Graphics g) {
    this.qt = qt;
    this.g = g;
  }

  public void draw(int x, int y)
    { ... }
  public void draw(int x, int y,
    int size) { ... }
}
```

Ved kall på draw skal bildet representert ved qt tegnes med øvre, venstre hjørne i posisjon (x,y).
size-parameteren forteller hvor stort bildet skal tegnes på skjermen.
Uten size-parameter skal bildet tegnes så stort som størrelsen på bildet tilsier.

Lag ferdig de to draw-metodene i class QTDisplay. Lag de hjelpemetoder du måtte ønske i class QTDisplay. QTDisplay ligger i samme pakke som QuadTre.

2e (15%)

Anta at under oppbygging av et quadtre blir først alle noder på alle nivå laget. Bildet i figur 1 vil altså generere et tre med rotnode, fire indre noder på neste nivå, og 4x4=16 pikselnoder på nederste nivå. Det gjenstår da å «rydde» i strukturen. Dersom en indre node har fire barn som alle er pikselnoder, og alle pikselnodene har samme verdi, kan den indre noden (med sine fire barn) erstattes med en pikselnode. Når dette er utført rekursivt i treet, kommer vi fram til det du tegnet i 2a.

Lag metoden rydd() i class QuadTre som utfører dette. Metoden skal også sørge for at alle indre noder får rett verdi. Lag de hjelpemetoder du måtte ønske i de klasser du måtte ønske.

Lykke Til!

Råd og retningsliner. Les oppgåveteksten godt før du går i gang med å løyse oppgåva. Deloppgåvene er uavhengige av kvarandre i den meining at om du ikkje får til ei oppgåve, kan du likevel gjere neste, som om den fyrste var løyst. Fordél tida godt på alle oppgåvene. Om du meiner ei oppgåve er upresis, så skriv din eigen presisering.

Oppgåve 1: Liste

1a (30%)

For eit par tiår sidan var det ikkje uvanleg å lage peikarkjeder på fylgjande vis. Ei Node-klasse inneheldt berre nestepeikar. Liste-klassa inneheldt fyrstepeikar og evt. sistepeikar av type Node. Dei klassene som skulle leggast i liste måtte lagast som subklasse av Node-klassa, sjå t.d. Person-klassa i dømet under. Ein kan enno treffe på slik kode som må vedlikehaldast.

```
public class Node { Node neste; }
public class Liste {
  private Node fyrste, siste;
  private int antal;
  ...
}
class Person extends Node { ... }

Liste l = new Liste();
l.settInnFørst(new Person(«Ola Nordmann», ...));
```

Du kan anta at Node og Liste ligg i same pakke. Bruk class Node slik den er over, og lag class Liste med fylgjande metoder:

```
public Liste()                      // Konstruktørmetode
public void settInnFørst(Node n)    // Set inn objektet først i lista
public void settInnSist(Node n)     // Set inn objektet sist i lista
public Node finn(int i)             // Returner objekt nr. i om det
                                    // finst, start tellinga på 0
public Node slett(int i)            // Slett og returner objekt nr. i
public boolean erTom()              // Returner true dersom tom liste
public void gjerTom()               // Gjer lista tom
public int antal()                  // Antal objekt i lista
```

Liste skal altså ha fyrste- og sistepeikar, og ein antal-variabel. Du skal ikkje nytte listehovud og -hale. Du skal skrive klassa frå grunnen av, utan å nytte noko i Collections API'et. Nemn både føremuner og ulemper med dette systemet i høve til dagens.

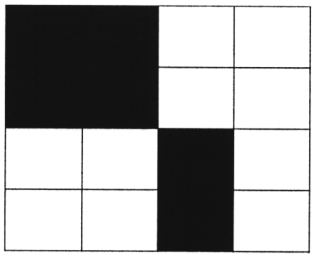
1b (10%)

Lag class ListeIterator som iterator-klasse for Liste-klassa over, også den ligg i same pakke. Den skal implementere Iterator-grensesnittet i Collections-API'et, den må altså ha hasNext() og next(). Klassa må også ha remove(), men den metoden skal ikkje ha nokon funksjon. Lag også iterator()-metode i class Liste.

1c (5%)
Lag metoden toString() i class Liste. Den skal returnere ein String med alle objekta kommaseparert og omslutta av [ ].

Oppgåve 2: Quadtre

Eit ***quadtre*** er eit tre der kvar node har null eller fire barn. Vi skal her nytte quadtre til å representere bilete med berre svart og kvitt, til dømes eit dokument. Kvar node i treet representerer eit kvadratisk område i biletet. Rotnoden representerer heile biletet, dens fire barn representerer kvar sin fjerdepart av biletet osv. Dersom området som ein node representerer er homogent, dvs. heilt kvitt eller heilt svart, skal noden ikkje ha barn. Vi seier då at noden er kvit (svart). Dersom området er inhomogent, dvs. inneheld både svart og kvitt, skal noden ha fire barn, vi seier at noden er grå. Vi begrensar oss til bilete som er kvadratiske, og der storleiken er ein toerpotens, t.d. 256x256, 512x512, 1024x1024.



Figur 1: Bilete på 4x4. Kvadrant ein opp til venstre er homogent svart og treng ikke splittast. Kvadrant to opp til høgre og tre ned til venstre er homogent kvite og treng ikkje splittast. Kvadrant fire ned til høgre er inhomogen og må ha fire barn.

2a (5%)
Teikn QuadTreet som representerer biletet i Figur 1. Set ein S inni svarte noder, ein K inni kvite, og ein G i grå/inhomogene noder. Teikn barna i rekkefylgje 1..4 slik forklart i figur 1.

Fylgjande uferdige klasser er laga. Eit QuadTre-objekt representerer eit heilt bilete. Eit Piksel-objekt representerer eit homogent område. IndreNode representerer eit inhomogent område. Node er abstrakt superklasse for Piksel og IndreNode.

```
public class QuadTre {
  Node rot;
  int høgde; // Maksimal høgde på treet

  public float verdi() { ... }
  public int storleik() { ... }
  public void roter() { ... }
  public void rydd() { ... }
}
```

Variablen høgde fortel kor høgt treet maksimalt kan vere. Dermed gjev den også storleiken på biletet. Dersom høgde er 3 har vi eit 8x8 bilete. Treets faktiske høgde kan vere mindre. Dersom biletet er heilt homogent vil faktisk høgde vere 0, treet har berre rotnode.

```
abstract class Node {
  float verdi;
  float verdi() { return verdi; }
  abstract float beregnVerdi();
}
```

Noden sin verdi er på skalaen 0..1 der 0 er svart og 1 er kvit.

```
class Piksel extends Node {
  float beregnVerdi() {
    return verdi;
  }
}
```

Eit piksel er her eit homogent område med pikselverdi 0 (svart) eller 1 (kvit).

```
class IndreNode extends Node {
  Node ov, oh, nv, nh;

  float beregnVerdi() { ... }
}
```

Ein indre node er eit inhomogent område som er delt i fire:
ov – opp opp til venstre
oh – opp til høgre
nv – ned til venstre
nh – ned til høgre

2b (10%)

Metoden verdi() i class QuadTre skal returnere gjennomsnittsverdien i biletet, verdien er lagra i rotnoden. Metoden storleik() i class QuadTre skal returnere storleiken på biletet, t.d. 8 når høgde er 3. Metoden beregnVerdi() i class IndreNode skal – rekursivt nedover i treet – berekne verdi til noden som gjennomsnittet av dei fire barna, legge verdien i verdi-variablen, og returnere verdien. Du kan anta at alle IndreNode-objekt har fire barn, ingen er null.
Lag desse tre metodane.

2c (10%)

Når eit bilete skal roterast 90 grader mot høgre, er det berre å flytte alle kvadrantane 90 grader mot høgre (med klokka) på alle nivå nedover i treet. Kvadranten opp til venstre skal plasserast opp til høgre, osv. Lag metoden

```
public void roter()
```

i class QuadTre som utfører dette. Lag dei hjelpemetoder du måtte ynskje i dei klassar du måtte ynskje.

2d (15%)

For framvisning av slike bilete er det laga eigen framvisningsklasse:

```
public class QTDisplay {
  private QuadTre qt;
  private Graphics g;

  public QTDisplay(QuadTre qt,
      Graphics g) {
    this.qt = qt;
    this.g = g;
  }

  public void draw(int x, int y)
    { ... }
  public void draw(int x, int y,
    int size) { ... }
}
```

Ved kall på draw skal biletet representert ved qt teiknast med øvre, venstre hjørne i posisjon (x,y).
size-parameteren fortel kor stort biletet skal teiknast på skjermen. Utan size-parameter skal biletet teiknast så stort som storleiken på biletet tilseier.

Lag ferdig dei to draw-metodane i class QTDisplay. Lag dei hjelpemetodar du måtte ynskje i class QTDisplay. QTDisplay ligg i same pakke som QuadTre.

2e (15%)

Anta at under oppbygging av eit quadtre vert fyrst alle noder på alle nivå laga. Biletet i figur 1 vil altså generere eit tre med rotnode, fire indre noder på neste nivå, og 4x4=16 pikselnoder på nederste nivå. Det gjenstår då å «rydde» i strukturen. Dersom ein indre node har fire barn som alle er pikselnoder, og alle pikselnodane har same verdi, kan den indre noden (med sine fire barn) erstattast med ein pikselnode. Når dette er utført rekursivt i treet, kjem vi fram til det du teikna i 2a.

Lag metoden rydd() i class QuadTre som utfører dette. Metoden skal også syte for at alle indre noder får rett verdi. Lag dei hjelpemetodar du måtte ynskje i dei klasser du måtte ynskje.

Lykke Til!