Om eksamen

Eksamensinformasjon – digital skoleeksamen

Fakultet: Teknologi, kunst og design
Utdanning: Teknologiske fag
Emnenavn: Algoritmer og Datastrukturer
Emnekode: (ORD) DATS2300 / ITPE2300

Dato: 19.12.2018
Tid: 15.00 - 18.00

Antall oppgaver: 5

Tillatte hjelpemidler: Ingen

Merknad:
Råd og tips:

1. Les gjennom hele oppgavesettet før du begynner og planlegg tiden.
2. Svar utfyllende på oppgavene så du viser at du har forstått pensum.
3. Bruk ikke for lang tid på et punkt. Gå isteden videre til neste punkt og eventuelt tilbake hvis du får god tid.
4. De fem deloppgavene teller likt.
5. Hvis du trenger en hjelpestruktur (liste, stakk, kø o.l.) fra java.util eller fra kompendiet, kan du fritt bruke den uten å måtte kode den selv. Men den må brukes på en korrekt måte. Men du bør si fra om dette i en kommentar.
6. Hvis du har idéer om hvordan ting skal løses, men likevel ikke klarer å få det til, kan du demonstrere idéene dine med ord, tegninger o.l.

1 Oppgave 1: Rekursjon og binærtrær

Oppgave 1: Rekursjon og binærtrær

I denne oppgaven handler om rekursjon og sortering av noder i binærtrær

a. Skriv ut verdien på nodene i treet i vedlegget ved å bruke
    i. Preorden
    ii. Inorden, og
    iii. Postorden.
b. Forklar kort hva en rekursiv funksjon er.
    i. Hvilke til krav stiller vi til en rekursiv funksjon skal virke etter hensikten?
    ii. Hva vil skje dersom ett eller begge disse kravene ikke fylles av funksjonen?
c. Skriv en programkode i Java som rekursivt skriver ut verdien av nodene i et binærtre i inorden sortering. Binærtreet er definert med kildekoden i vedlegget, og du skal fylle ut innholdet i funksjonen printInOrder()
d. Hvordan kan du skrive ut treet inorden uten å bruke rekursjon? Hva må du i så fall gjøre for å få det til? Du skal **IKKE** å skrive kode i denne oppgaven.

Skriv ditt svar her...

Maks poeng: 20

2 # Oppgave 2: Huffmantrær

Oppgave 2: Huffmantrær

I denne oppgaven skal du bruke et Huffmantre til å kode en tekst.

a. Ta utgangspunkt i teksten «BANANKAKE», og skriv opp en tabell med frekvenser for hver bokstav.
b. Hva betyr det venstreorienterte kanoniske Huffmantreet? Hva er kravene til det venstreorienterte kanoniske Huffmantreet?
c. Lag det venstreorienterte kanoniske Huffmantreet ut ifra frekvensene i oppgave a). Om du ikke får til oppgave a) kan du bruke følgende frekvenser: A: 21, B: 7, E: 7, K: 14, N: 14.
    i. Lag en tegning av Huffmantreet, og
    ii. Skriv opp en tabell med Huffmankoder for hver bokstav
d. Bruk så Huffmankodene til å komprimere «BANANKAKE».
    i. Skriv opp den kodede binære meldingen.
    ii. Hvor mange bit bruker du i den kodede meldingen?

Skriv ditt svar her...

Maks poeng: 20

3 Oppgave 3: Maksimumsheap

Oppgave 3: Maksimumsheap

I denne oppgaven skal vi bruke en maksimumsheap og se hvordan den kan brukes til sortering

a. Hva er en maksimumsheap, og hvilke krav stilles for at det skal kunne kalles en maksimumsheap?
b. Start med en tom maksimumsheap. Legg tallene 1, 2, 3, 4, 5, 6, 7 og tegn heapen for hvert tall du legger inn.
c. Vi skal nå ta ut **tre** tall fra heapen vist i vedlegget. Ta ut ett og ett tall og tegn heapen for hvert tall du tar ut. Merk: Ta utgangspunkt i maksimumsheapen som ligger i vedlegget.
d. Forklar hvordan en maksimumsheap kan brukes til sortering uten å bruke ekstra lagringsplass.

Skriv ditt svar her...

Maks poeng: 20

4 Oppgave 4: Korteste vei i en graf

Oppgave 4: Korteste vei i en graf

I denne oppgaven skal vi finne korteste vei i en graf ved hjelp av Dijkstras algoritme.

a. Beskriv med ord hvordan Dijkstras algoritme finner den korteste vei i en graf.
b. Ta utgangspunkt i grafen i vedlegget. Lengden på hver kant står i firkantene.
    i. Hvor lang er korteste vei fra node F til node H? Lag en tegning som viser hvordan du kom frem til svaret ved hjelp av Dijkstras algoritme.
    ii. Hvilke noder passerer du når du følger korteste vei?

Skriv ditt svar her...

Maks poeng: 20

5 Oppgave 5: Lenket liste

Oppgave 5: Lenket liste

I denne oppgaven skal du kode fjerning av en node i en lenket liste.

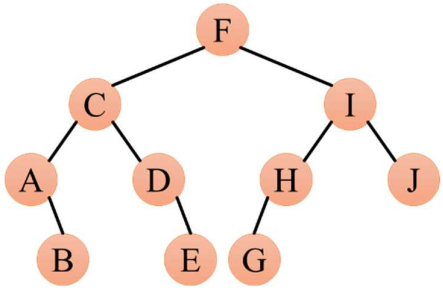
a. Ta utgangspunkt i kildekoden i vedlegget, og anta at alle funksjoner er implementert korrekt. Hva vil utskriften av programmets main-metode være? Forklar hvorfor.
b. Ta utgangspunkt i figuren i vedlegget. Skriv funksjonen remove(Node q) som er markert i kildekodevedlegget.
c. Ta utgangspunkt kildekoden i vedlegget og skriv funksjonen remove(int index) som fjerner noden på nevnte plass.

Skriv ditt svar her...

Maks poeng: 20

Question 1
Attached

Vedlegg 1a: Binærtre

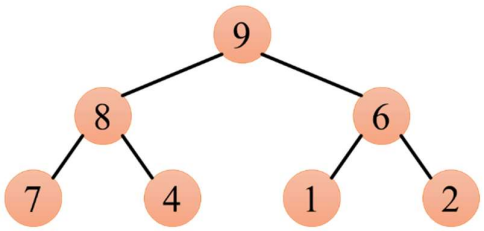


Vedlegg 1b: Kildekode for binærtre

```java
/**
 * Klasse som representerer en node i et binærtre
 * left_child er venstre barn til denne noden
 * right_child er høyre barn til denne noden
 * value er verdien denne noden holder
 */
public static class BTNode {
    BTNode left_child;
    BTNode right_child;
    char value;

    /**
     * Konstruktør som lager en ny node
     * @param value Verdien dene noden skal ha
     */
    BTNode(char value) {
        this.value = value;
    }

    /**
     * Legger til en ny verdi i det binære treet
     * @param value Verdien å legge til
     */
    void add(char value) {
        if (value < this.value) {
            if (this.left_child != null) {
                this.left_child.add(value);
            }
            else {
                this.left_child = new BTNode(value);
            }
        }
        else {
            if (this.right_child != null) {
                this.right_child.add(value);
            }
            else {
                this.right_child = new BTNode(value);
            }
        }
    }

    /**
     * Skriver ut inorden ved hjelp av rekursjon
     */
    public void printInOrder() {
        throw new UnsupportedOperationException("Ikke kodet ennå!");
    }
}

public static void main(String[] args) {
    BTNode root = new BTNode('F');
    char values[] = "CIADHJBEG".toCharArray();

    for (char val : values) {
        root.add(val);
    }
    root.printInOrder();
}
```

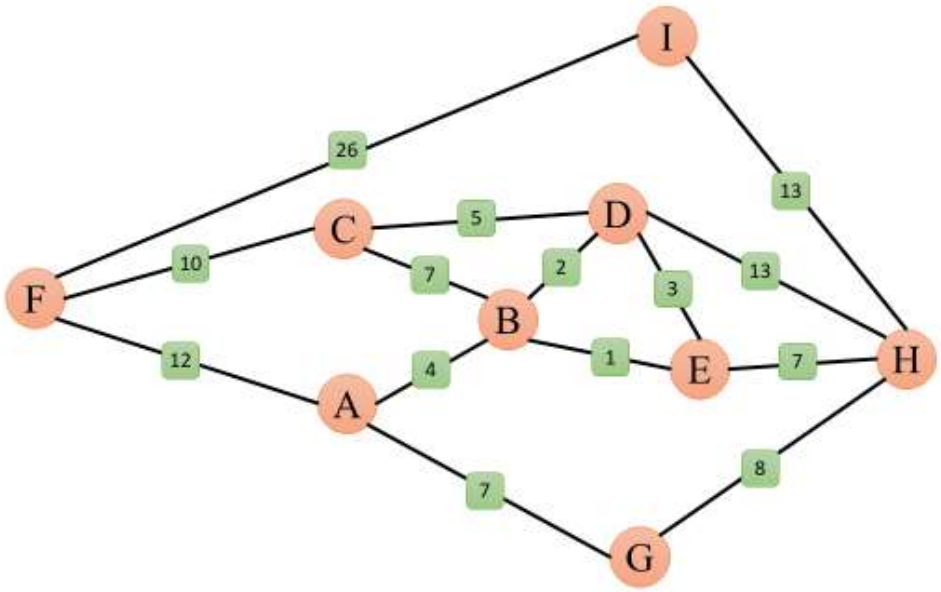
Question 3
Attached

Vedlegg 3a: Heapen



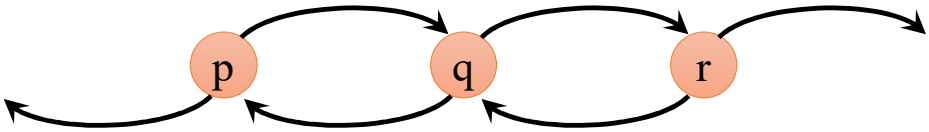
Question 4
Attached

Vedlegg 4a: Graf



Question 5
Attached

Vedlegg 5a: Noder i en dobbelt lenket liste



Vedlegg 5b: Kildekode

```
/**
 * Klasse som representerer en dobbelt lenket liste
 */
public static class DoubleLinkedList {
    int size;
    Node start;
    Node end;

    /**
     * Klasse som representerer en node i en dobbelt lenket liste
     * next er neste node,
     * prev er forrige node, og
     * value er verdien til denne noden
     */
    public static class Node {
        Node next;
        Node prev;
        char value;

        /**
         * Konstruktør som lager en ny node
         * @param value Verdien dene noden skal ha
         */
        Node(char value) {
            this.value = value;
        }
    }

    DoubleLinkedList() {
        this.size = 0;
        this.start = null;
        this.end = null;
    }

    /**
     * Funksjon som legger til en verdi på slutten av listen
     */
    void addBack(char value) {
        Node new_node = new Node(value);

        if (this.start == null) {
            this.start = new_node;
        }
        if (this.end == null) {
            this.end = new_node;
        }

        new_node.prev = this.end;
        this.end.next = new_node;

        new_node.next = this.start;
        this.start.prev = new_node;

        this.end = new_node;
        this.size += 1;
    }
```

```
    /**
     * Funksjon som fjerner noden på plass index
     * @param index Indexen å fjerne
     */
    void remove(int index) {
        throw new UnsupportedOperationException("Ikke kodet ennå!");
    }

    /**
     * Funksjon som fjerner noden q fra en dobbelt lenket liste
     * @param q Noden som skal fjernes
     */
    void remove(Node q) {
        throw new UnsupportedOperationException("Ikke kodet ennå!");
    }

    /**
     * Funksjon som skriver ut den lenkede listen
     */
    void print() {
        Node current = this.start;
        System.out.print(current.value);
        for (int i=1; i<this.size; ++i) {
            current = current.next;
            System.out.print(", " + current.value);
        }
        System.out.println();
    }
}

public static void main(String[] args) {
    DoubleLinkedList list = new DoubleLinkedList();

    char values[] = "ABCDEFGHI".toCharArray();

    for (char val : values) {
        list.addBack(val);
    }

    list.print();

    list.remove(2);
    list.remove(4);
    list.remove(0);
    list.print();

    list.remove(6);
    list.print();
}
```