Høgskolen i Østfold

EKSAMEN

Emnekode: ITF20006 000	Emne: Algoritmer og datastrukturer
Dato: 19. mai 2010	Eksamenstid: 09:00 til 13:00
Hjelpemidler: 8 A4-sider (4 ark) med egne notater	Faglærer: Gunnar Misund

Oppgavesettet består av 5 sider inklusive denne forsiden. Kontroller at oppgaven er komplett før du begynner å besvare spørsmålene.

I implementasjonsoppgaver gir Java-kode best uttelling, men godt dokumentert pseudo-kode kan også brukes. Gjør dine egne forutsetninger hvis du føler behov for det.

Oppgavesettet består av 4 oppgaver, med tilsammen 13 deloppgaver. Hver hovedoppgave er vektet, og alle deloppgavene i en oppgave teller likt. Alle oppgavene skal besvares. Ved sensur teller hver deloppgave like mye. Karakteren fastsettes dog på basis av en helhetsvurdering av besvarelsen.

SKRIV TYDELIG :-)

Sensurdato: 16. juni 2010

Karakterene er tilgjengelige for studenter på studentweb senest 2 virkedager etter oppgitt sensurfrist. Følg instruksjoner gitt på: www.hiof.no/studentweb

Oppgave 1: 20%

A) *Hva er effektiviteten i O-notasjon til følgende funksjon (begrunn svaret):*

$F(n) = 13 + (14n^2 + \log_3 n) / n$

B) *Forklar kort hva binærsøk er, og redegjør for effektiviteten til slike søk.*

C) *Implementer metoden*

```
void delete(int[] arr, int idx)
```

som sletter element nr `idx` *i tabellen* `arr`. *Angi metodens effektivitet, inkludert begrunnelse.*

D) *Implementer metoden*

```
void delete(ListNode head, ListNode tail, ListNode del)
```

som fjerner noden med referansen `del` *i en dobbeltlenket liste der første node er* `head`*, og siste node er* `tail`*. Du kan anta at* `del` *ikke er første eller siste node. Angi metodens effektivitet, inkludert begrunnelse. Se vedlegg.*

E) *Betrakt funksjonen*

```
void func (int[] arr) {
      int i = arr.length-1;
      int j = 0;
      while(i >= 0) {
            for (int k = 0; k <= i; k++) {
                  j += arr[k];
            }
            i--;
      }
}
```

Angi funksjonens effektivitet, inkludert begrunnelse.

Oppgave 2: 30%

A) *Foreleser forteller begeistret at han har pønsket ut en genial sorteringsmetode, som ved en kløktig kombinasjon av trær og grafer viser seg å være av* ***lineær*** *orden. Hva tror du om dette?*

B) *Forklar hvordan utvalgssortering foregår. Anvend metoden på følgende sekvens, og illustrer hvert trinn.*

13 - 8 - 56 - 1 - 9 - 41 - 18

Diskuter metodens effektivitet.

C) *Implementer metoden* `ListNode sort(ListNode head)` *som returnerer den første noden i en sortert, dobbeltlenket liste, der nodene er hentet fra listen med start* `head`*. Sorteringen skal foregå etter utvalgsprinsippet. Forklar hva som eventuelt skjer med den originale input-lista. Se vedlegg.*

Oppgave 3: 20%

A) *Bygg opp en MaxHeap ved å sette inn disse elementene (i gitt rekkefølge):*

B - G - Y - F - A - Z

Fjern deretter det største elementet. Illustrer de ulike trinnene ved både innsettingene og fjerningen.

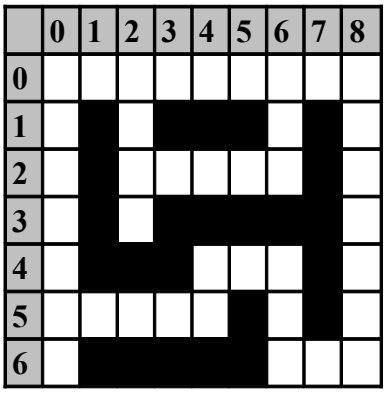
B) *Bygg opp et AVL-tre (binært, balansert søketre) ved å sette inn følgende elementer (i den gitte rekkefølgen):*

K - F - R - T - V - M

Illustrer de ulike trinnene i prosessen, og redegjør for hvilke regler du bruker.

Oppgave 4: 30%

Vi har gitt følgende labyrint på matriseform (9 kolonner i x-retning, 7 rader i y-retning, svart markerer blokkerte ruter, grå markerer indeksene for kolonnene og radene):



Den rekursive metoden **`move`** kan benyttes for å finne en vei gjennom labyrinten (se vedlegg).

A) *Hva blir skrevet ut til skjerm ved kallet* **`move`**`(maze, 0, 0, 8, 6)`*, der* `maze` *er labyrinten over? Du kan stoppe når du har kommet fram til mål. Forklar kort hva som skjer videre etter at man har funnet fram til mål første gang. Bruk gjerne illustrasjoner.*

B) *Forklar med tekst og illustrasjoner hvordan labyrinten over kan modelleres som et nettverk. Diskuteter spesielt hvordan vi kan minimere antallet noder.*

C) *Vi definerer lengden på en vei gjennom labyrinten som antallet flyttinger fra en celle til en nabocelle. Bruk Dijkstras algoritme for å finne korteste vei gjennom labyrinten over (fra* `0,0` *til* `8,6`*), basert på nettverksmodellen du har foreslått i forrige deloppgave. Forklar hva som skjer i hvert steg, gjerne ved å bruke en passende tabell.*

Vedlegg

```
public class ListNode {
      public int val;
      public ListNode prev;
      public ListNode next;
}

public void move(boolean[][] maze, int sx, int sy, int dx, int dy) {
      // Blocked cells are initially marked as "false";
      // sx/sy represents the current move,
      // and dx/dy is the destination cell
      if (!valid(maze, sx, sy)) {
            // Return if we are outside the labyrint
            // or the cell value is false
            return;
      }

      // Assume we have a global counter, initially set to zero
      moveNo++;
      System.out.println(""+moveNo+": "+sx+","+sy);
      if (sx == dx && sy == dy) {
            System.out.println("Found!");
            return;
      }
      maze[sx][sy] = false;
      move(maze,  sx,     sy + 1, dx, dy);
      move(maze,  sx - 1, sy,     dx, dy);
      move(maze,  sx,     sy - 1, dx, dy);
      move(maze,  sx + 1, sy,     dx, dy);
}
```

Slutt på oppgavesettet.