TENTAMEN I
DATASTRUKTURER OCH ALGORITMER DVG B03

140114 kl. 08:15 – 13:15

Ansvarig Lärare: Donald F. Ross

Hjälpmedel: Inga. Algoritmerna finns i de respektive uppgifterna.

***** OBS *****

Betygsgräns:

Kurs:	Max 60p, Med beröm godkänd 50p, Icke utan beröm godkänd 40p, Godkänd 30p **(varav minimum 15p från tentamen, 15p från labbarna)**
Tentamen:	Max 30p, betyg 5: 26p-30p, betyg 4: 21p-25p, betyg 3: 15p-20p
Labbarna:	Max 30p, betyg 5: 26p-30p, betyg 4: 21p-25p, betyg 3: 15p-20p

SKRIV TYDLIGT – LÄS UPPGIFTERNA NOGGRANT

Ange alla antaganden.

---

(1) Ge ett kortfattat svar till följande uppgifter ((a)-(j)).

(a) En definition av en mängd
(b) En definition av en sekvens
(c) En definition av ett träd
(d) En definition av en graf
(e) En definition av ett AVL-träd

Förklara Dina exempel ((f)-(j)) med några meningar

(f) Ett exempel av ett fullt träd som inte är komplett
(g) Ett exempel av ett komplett träd som inte är fullt
(h) Ett exempel av ett free tree
(i) Ett exempel av en algoritm som är $O(n^3)$
(j) Ett exempel av ett B-tree

Totalt 5p

(2) Ge ett kortfattat svar till följande uppgifter.

(a) En definition av rekursion
(b) En definition av abstraktion
(c) Principerna bakom hashning
(d) Ett exempel samt beskrivning av en girig algoritm
(e) En definition av Big-Oh – O(x)

Totalt 5p

(3) Rekursion

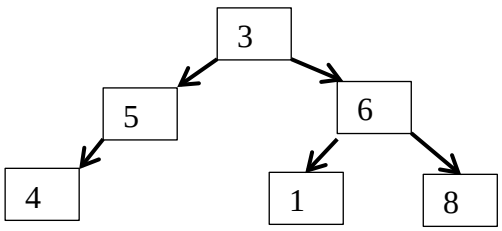
(a) Beskriv vilket **mönster** man hittar hos rekursiva funktioner (1p)

(b) Utifrån din definition i (a) ovan, skriv rekursiv pseudokod för att lägga till ett element i en sekvens i stigande ordning. **Ange alla antaganden.**
Visa hur Din pseudokod fungerar med detta exempel – **lägga till 7 till (1,4,5)**

(2p)

(c) Utifrån din definition i (a) ovan, skriv rekursiv pseudokod för funktionen **T2Q( )** (träd till kö) från trädlabben **Ange alla antaganden.**
Visa hur Din pseudokod fungerar med detta exempel:

(2p)
Totalt 5p



(4) Prims algoritm

a) Tillämpa **Prims algoritm** (nedan) på den **oriktade** grafen:

(a-7-b, a-1-c, a-10-d, b-3-c, b-2-e, c-15-d, c-4-e, c-9-f, d-4-f, e-8-f).

Börja med nod a. Ange alla antaganden och visa alla beräkningar och mellanresultat. Vad representerar resultatet? **Ange varje steg i din beräkning.**

Rita delresultatet efter varje iteration.

(2p)

b) **Förklara principerna bakom Prims** algoritm.

(2p)

c) Visa hur Du skulle använda **principerna** bakom **Kruskals algoritm** för att bekräfta resultatet från Prims algoritm.

(1p)

Totalt 5p

Ange *alla* antaganden och visa *alla* beräkningar och mellanresultat

```
Prim ( node v) -- v is the start node

{   U = {v}; for i in (V-U) { low-cost[i] = C[v,i]; closest[i] = v; }

    while (!is_empty (V-U) ) {
            i = first(V-U); min = low-cost[i]; k = i;
             for j in (V-U-k) if (low-cost[j] < min) { min = low-cost[j]; k = j; }
            display(k, closest[k]);
            U = U + k;
            for j in (V-U) if ( C[k,j] < low-cost[j] ) )  { low-cost[j] = C[k,j]; closest[j] = k; }
   }
}
```

(5) Topologisk sortering

Vid ett universitet har vissa kurser förkunskapskrav. I datavetenskap kräver kompilatorkonstruktion (DAV D02) programspråk (DAV C02) som förkunskap. Datastrukturer och algoritmer (DAV B03) är ett förkunskapskrav till programspråk, avancerad programmering i C++ (DAV C05), samt projektarbete i Java (DAV C08). Datastrukturer och algoritmer kräver diskret matematik (MAA B06) samt programutvecklingsmetodik (DAV A02). Operativsystem (DAV B01) kräver i sin tur programutvecklingsmetodik och datorsystemteknik (DAV A14) och är förkunskapskrav till C och UNIX (DAV C18), tillämpad datasäkerhet (DAV C17) samt realtidssystem (DAV C01). Objektorienterade designmetoder (DAV D11) kräver bägge avancerad programmering i C++ och software engineering (DAV C19).

Hur kan man **visa** att kompilatorkonstruktion och objektorienterade designmetoder kräver diskret matematik?

I vilken ordning ska en student som vill läsa på D-nivå ta alla de ovannämnda kurserna? Metoden som kan användas för att komma fram till en lösning heter topologisk sortering. En variant av topologisk sortering är följande algoritm:

Topological Sort

```
tsort(v) -- prints reverse topological order of a DAG from v
{       mark v visited
        for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```

Vilken är den andra varianten? Tillämpa både varianter i din lösning till problemet ovan.

Vilka begränsningar måste man ta hänsyn till?

5p

(6) Diskussion

Diskutera ingående meningen bakom denna kurs. Vilka var de huvudidéer som studerats under kursens lopp och vilken relevans har dessa idéer för datavetenskap, för andra ämnen inom datavetenskap och för programmering? **Ange gärna exempel** för att stödja Din diskussion.

5p