Høgskolen i Gjøvik
Avdeling for informatikk og medieteknikk

E K S A M E N

EMNENAVN: **Algoritmiske metoder**

EMNENUMMER: **IMT2021**

EKSAMENSDATO: **4. desember 2007**

KLASSE(R): **06HBINDA / 06HBPUA / 06HBISA / ……**

TID: **09.00-14.00**

EMNEANSVARLIG: **Frode Haug**

ANTALL SIDER UTLEVERT: **4 (inkludert denne forside)**

TILLATTE HJELPEMIDLER: **Alle trykte og skrevne.**

- Kontroller at alle oppgavearkene er til stede.
- Innføring med penn, eventuelt trykkblyant som gir gjennomslag. Pass på så du ikke skriver på mer enn ett innføringsark om gangen (da det blir uleselige gjennomslag når flere ark ligger oppå hverandre).
- Ved innlevering skilles hvit og gul besvarelse, som legges i hvert sitt omslag.
- Oppgavetekst, kladd og blå kopi beholder kandidaten til klagefristen er over.
- Ikke skriv noe av din besvarelse på oppgavearkene. Men, i oppgavetekst der du skal fylle ut svar i tegning/tabell/kurve, skal selvsagt dette innleveres sammen med hvit besvarelse.
- Husk kandidatnummer på alle ark. Ta vare på dette nummeret til sensuren faller.

Oppgave 1 (teori, 25%)

Denne oppgaven inneholder tre uavhengige oppgaver fra kap. 8, 11 og 15 i læreboka.

a) Vi skal utføre Shellsort på key'ene "SILLEDILL". Jfr. kode s.109 i læreboka.
For hver gang indre for-løkke er ferdig (etter: `a[j] = v;`):
Tegn opp arrayen og skriv verdiene til 'h' (4 og 1) og 'i' underveis i sorteringen.
Marker spesielt de key'ene som har vært involvert i sorteringen (jfr. fig.8.7 s.108).

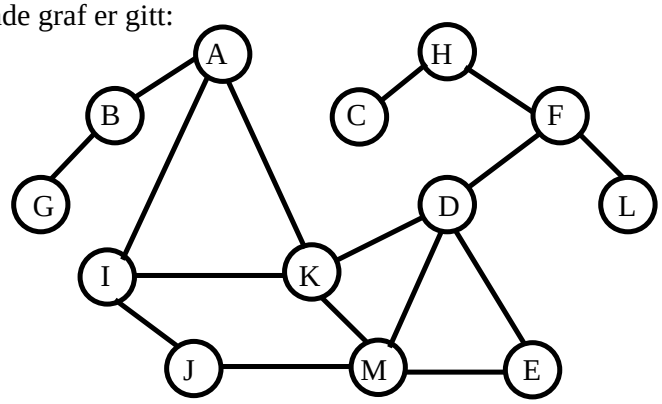
b) Teksten "S I L L E D I L L E N" (i denne rekkefølge fra venstre til høyre, blanke regnes ikke med) skal heap-sorteres vha. bottom-up heap konstruksjon.
(Se fig.11.8, koden s.156 og fig.11.9 i læreboka.)
Tegn opp heapens innhold etterhvert som heapen konstrueres og deretter sorteres etter denne metoden (Dvs. lag en figur etter samme prinsipp som fig.11.9.)

c) Legg key'ene "SILLEDILLSALAT" (i denne rekkefølge fra venstre til høyre) inn i et 2-3-4 tre.
Tegn opp treet etterhvert som bokstavene legges inn.
Gjør også om sluttresultatet til et Red-Black tre.

Oppgave 2 (teori, 25%)

Denne oppgaven inneholder tre uavhengige oppgaver fra kap.30, LZW og kap.33.

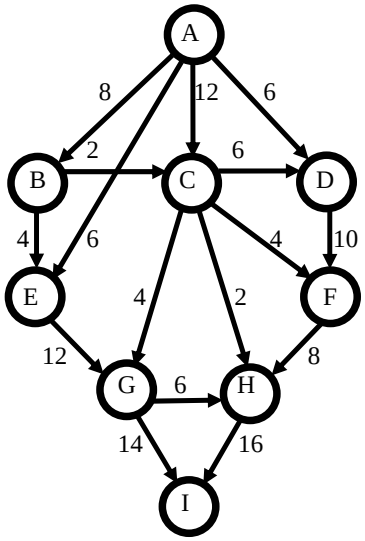
a) Følgende graf er gitt:



Grafen er representert som/i en nabomatrise.
Startende i node 'M' - **tegn opp dybde-først søketreet for denne grafen.**
Forklar ut fra dette treet hvilke node(r) som er *artikulasjonspunkt(er)* i grafen.

b) LZW-teknikken skal brukes på følgende tekststreng: "BARE BANAN BADER BAR"
Katalogen inneholder allerede alle de standard ASCII-tegnene i indeksene 0-255.
Hva blir katalogens innhold f.o.m. indeks nr.256 og oppover?
Hva blir den kodede (output) strengen?

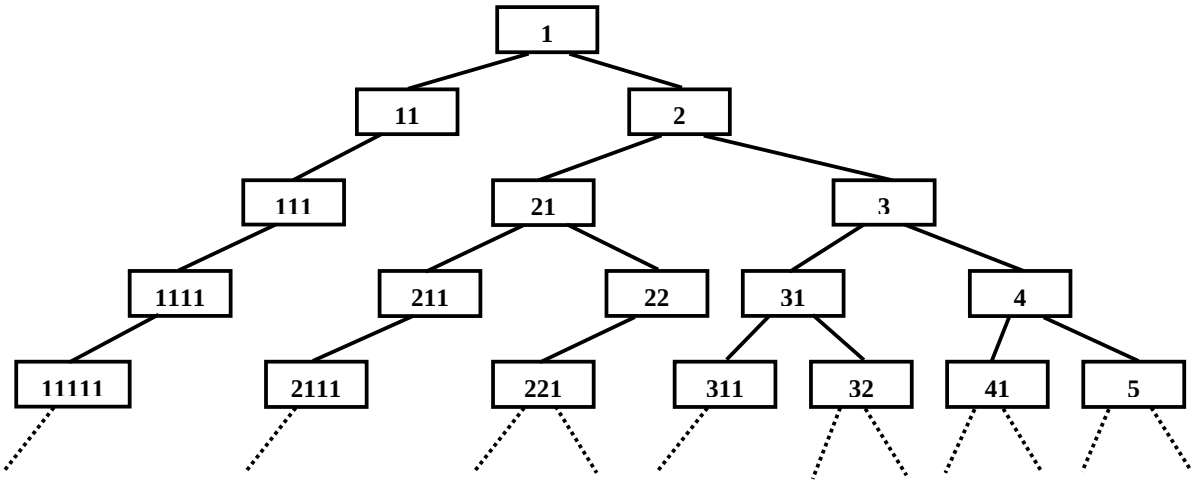
c) Vi har følgende nettverk (dvs. en graf som er både vekted og retted):



'A' er kilden og 'I' er kummen. Kapasiteten for hvert enkelt rør/kant er skrevet på figuren, og flyten i røret går i pilretningen. **Tegn figuren opp igjen, men nå med ny verdi på hver kant, som forteller hva som er max. flyt langs hver av kantene (fra kilde til kum).**

Oppgave 3 (teori & koding, 25%)

Gitt et tre med følgende utseende, node-verdier og egenskaper:



En node er definert som:

```
struct Node {
   int   ID;      //  Nodens ID/key/nøkkel/navn (et tall).
   Node* left;    //  Peker til venstre subtre, evt. z-noden om det er tomt.
   Node* right;   //  Peker til høyre subtre, evt. z-noden om det er tomt.
   Node (int id, Node* l, Node* r)  {  ID = id ;  left = l;  right = r;  }
};
```

Vi har de to globale variablene:

```
Node* z   = new Node(0, NULL, NULL);
Node* rot = new Node(1, z, z);
```

a) **Hvilke verdier blir i de 11 nodene på det neste nivået**

(5%) (Dvs. de på nivå nr.6 – hvor disse vil ligge er angitt av de stiplede linjene.)

b) **Beskriv *kort* reglene dette treet er laget/generert etter.**

(5%) **NB:** z-nodene er ikke påtegnet (disse ligger der høyre subtre mangler, og som både venstre og høyre subtre hos *alle* nodene på det siste/nederste nivået).

c) **Lag den *rekursive* funksjonen** `void bygg_tre(int lev, Node* p)`

(15%) Funksjonen bygger et tre (så lenge `lev` er mindre enn const'en `MAX_NIVA`) etter de reglene du fant frem til i oppgave 3b (og med utseendet ovenfor når `MAX_NIVA` = 5).
Den startes fra `main` med følgende kall: `bygg_tre(1, rot);`

Oppgave 4 (koding, 25%)

Vi har følgende åtte regnestykker:

H	+	I	=	8
A	-	B	=	1
E	-	H	=	1
C	+	E	=	14
F	+	A	=	7
B	-	D	=	1
G	+	F	=	11
I	+	C	=	7

Oppgave går ut på å skrive et komplett program (med hjelpearray(er) og funksjoner) som finner og skriver ut hvilket siffer (1-9) som "skjuler seg" bak hver enkelt bokstav.
Om du oppdager at du kan gjenbruke kode fra emnet/kurset, så holder det å *henvise* til dette.
Programmet *bør* nok bl.a. være basert på en rekursiv løsning, men den trenger ikke å være spesielt effektiv!

Lykke til!

frode@haugianerne.no