Universitetet i Bergen
Det matematisk-naturvitenskapelige fakultet
Eksamen i emnet INF 102 - Algoritmer, datastrukturer og programmering
Mandag 18. februar 2013, kl 09:00 - 12:00
Bokmål

Ingen tillatte hjelpemidler.
Du trenger ikke å skrive java-kode for noen av oppgavene. Det holder med pseudokode eller en beskrivelse av algoritmen.
Alle svar må begrunnes!

Oppgave 1

I denne oppgaven skal vi se på ulike måter å evaluere et polynom $f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1} + a_nx^n$ for en gitt verdi av x. Du kan anta at verdien til a_i er gitt ved a[i], der a[] er en tabell av lengde n+1 (med første indeks 0). Videre har vi kun tilgang til operasjonene + og *, og ikke «opphøyd i».

a) Følgende kode gir en rett-frem måte å beregne f(x) for en gitt verdi av x. Utled en formel som gir det eksakte antall + og * operasjoner som blir utført. Formelen skal være en funksjon av n og du trenger ikke å skille mellom + og * operasjoner, det er summen av antall operasjoner du skal frem til.

```
fx = a[0];
for (i=1;i<=n;i++) {
   prod = a[i];
   for (j=1;j<=i;j++)
     prod = prod * x;
   fx = fx + prod;
}
return fx;
```

b) Vis hvordan du kan skrive om koden slik at du ikke trenger den innerste løkken (j-løkken). Gjør samme analyse som i oppgave a).

c) Dersom n=4 kan vi skrive f(x) som $a_0 + x(a_1 + x(a_2 + x(a_3 + x(a_4))))$. Denne måten å skrive f(x) kalles for *Horners formel* og kan generaliseres for vilkårlig verdi av n. Skriv en kode som bruker denne måten å beregne f(x) og analyser den på samme måte som i a) og b). Hint: Start beregningen fra den innerste parentesen.

Oppgave 2

a) Gitt en heap T og en nøkkelverdi k. Beskriv og gi kjøretiden til en effektiv algoritme for å skrive ut alle verdier i T som har nøkkelverdi større enn k.

b) Forklar hvorfor ingen algoritme som baserer seg på sammenligninger kan bygge et binært søketre (BST) gjennom å foreta færre enn lg(N!) ~N lg N sammenligninger.

Oppgave 3

Anta at vi bruker Union-Find datastrukturen for å representere mengder over tallene {1,2,3,4,5,6,7,8,9}. I utgangspunktet er hvert av tallene en egen mengde. Vi utfører nå Union() operasjoner fortløpende på hvert av følgende par: (4,7), (3,4), (3,8), (2,5), (1,2), (6,9), (6,1), (7,5).

a) Vis hvordan den endelige datastrukturen ser ut dersom vi alltid setter en lavere nummerert rot-node til å peke på en høyere nummerert rot-node.

b) Vis hvordan den endelige datastrukturen ser ut dersom vi alltid setter rot-noden til en mengde med færre elementer til å peke på rot-noden til en mengde med flere elementer. Dersom mengdene har like mange element bruker vi regelen fra a) for å bestemme hvem som peker på hvem.

c) Anta nå at vi representerer mengder over tallene {1,2,3...n-1,n} for en vilkårlig verdi n. Hvor mange sammenligninger kan vi risikere å måtte foreta i verste tilfelle når vi utfører en Find() operasjon dersom vi utfører Union() operasjonen som beskrevet i henholdsvis metode a) og metode b)? Gi svaret som en funksjon av n.

Oppgave 4

La G(V,E) være en vektet urettet graf med positive kant-vekter.

a) Vis at dersom du endrer alle vektene til kantene i G gjennom å multiplisere hver kant-vekt med et positivt tall t så vil det minste utspennende treet til G ikke påvirkes.

b) Vis at dersom alle kant-vektene til G har distinkte verdier så er det minste utspennende treet til G unikt.

c) Anta at alle kantene til G har distinkte verdier.

1. Må den letteste kanten til G være med i det minste utspennende treet til G?
2. Kan den tyngste kanten til G være med i det minste utspennende treet til G?
3. Må den letteste kanten til hver sykel i G være med i det minste utspennende treet til G?