Institut for Matematik og Datalogi
Syddansk Universitet, Odense

21. september 2019
LMF/RF

Eksaminatorier DM534 Uge 39

Studiegrupperne handler om at bruge hinanden til at arbejde med stoffet og sammen forsøge at løse ugens opgaver inden eksaminatorietimerne. I denne uge kræver opgave 1 og (især) opgave 2 gode ideer, og er derfor velegnede til at løse sammen med andre i en studiegruppe. Mange af resten af opgaverne er formuleret som spørgsmål, hvilket også naturligt kan danne baggrund for at arbejde i en studiegruppe.

Husk at læse de relevante slides før du/I forsøger at løse en opgave.

1. I opgave 11 fra uge 37 blev beskrevet følgende alternative metode til at skifte fortegn på heltal repræsenteret i two's complement:

   Invertér alle bits i tallet og læg derefter 1 til tallet.

   Implementer denne metode i et program til CPU-simulatoren fra labtimerne.

   Mere præcist, lav et program som læser et heltal x (i two's complement) fra RAM celle 20 og skriver tallet $-x$ (i two's complement) i celle 22.

   [Hint: bits i x kan inverteres ved bitwise XOR af x med et bestemt bitmønster (hvilket?).]

2. [Lidt svær] CPU-simulatoren har kommandoer til addition, men ikke multiplikation. Find på en metode til at lave multiplikation ud fra de eksisterende kommandoer.

   Mere præcist, lav et program som læser to heltal x og y (i two's complement) fra RAM celle 20 og 22, og derefter beregner $x \cdot y$ og skriver resultatet i RAM celle 24.

Dit program behøver kun fungere for ikke-negative tal, dvs. $x, y \geq 0$. Resultatet kan også kun forventes at være korrekt når $x \cdot y \leq 127$, eftersom CPU-simulatoren bruger 8-bits heltal i two's complement.

[Hint: Én metode kan baseres på at multiplikation pr. definition er en masse additioner. En anden metode kan baseres på opgave 15 fra uge 37 (udfordrende at implementere, men giver et hurtigere program).]

3. Er nedenstående en algoritme?

$$
\begin{array}{l}
i = 0 \\
\textbf{While } i \neq 5 \\
\quad i = i + 2
\end{array}
$$

4. Betragt listen $L = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]$. I nedenstående spørgsmål tæller vi sammenligninger, som involverer elementer i listen.

   (a) Hvor mange sammenligninger foretages der med SequentialSearch$(L, 7)$?

   (b) Hvor mange sammenligninger foretages der med BinarySearch$(L, 7)$?

   Antag nu, at L indeholder 10.000 elementer.

   (c) Hvor mange sammenligninger foretager man i værste tilfælde med en sekventiel søgning i L?

   (d) Hvor mange sammenligninger foretager man i værste tilfælde med en binær søgning i L?

5. Udfyld de manglende felter (undtagen øverste række) i tabellen på side 11 i forelæsningsnoterne om algoritmer.

6. Husk på algoritmerne til, ciffer for ciffer, at addere eller gange to tal i hånden (også nævnt på slides om repræsentation af tal). Hvis du ikke helt kan huske algoritmerne, er her et eksempel:

$$
\begin{array}{rr}
 & 321 \\
+ & 281 \\
\hline
 & 602
\end{array}
\qquad\qquad
\begin{array}{r}
321 \times 281 \\
\hline
281 \\
562 \\
843 \\
\hline
90201
\end{array}
$$

   (a) Hvad er køretiden for at addere to tal med n cifre hver? Hvad er den karakteristiske operation?

(b) Hvad er køretiden for at gange to tal med n cifre hver? Hvad er den karakteristiske operation?

7. Hvilke af følgende udsagn er sande?

   (a) $n \in O(n)$

   (b) $2n + 5 \in O(n)$

   (c) $\sqrt{n} - \log(n) \in O(n)$

   (d) $(\log(n))^2 \in O(n \log n)$

   (e) $n^2 \in O(n)$

   (f) $n \in O(n^2)$

   (g) $n \log(n) \in O(n^2)$

   (h) $n \log(n) \in O(n)$

   (i) $3n^2 + 2n + 1 \in O(n^2)$

   (j) $3n^2 + 2n + 1 \in O(n)$

8. Betragt følgende algoritme til at finde det mindste tal i listen L.

   <u>MIN</u>(L)
   $n = L$.length
   min $= L[1]$
   For $i = 2$ **to** n
       If $L[i] <$ min
           min $= L[i]$
   Return min

   (a) Hvad er algoritmens køretid?

   (b) Opskriv en løkke-invariant for algoritmen, og bevis, at den altid finder det mindste element i L.

   (c) Omskriv algoritmen, så den bruger en while-løkke i stedet for en for-løkke.

   (d) Bemærk, at algoritmen er iterativ. Skriv en rekursiv version af algoritmen.