Skriftlig Eksamen
Algoritmer og Datastrukturer (dADS)

Datalogisk Institut
Aarhus Universitet

Tirsdag den 27. maj 2003, kl. 9.00–13.00

Opgave 1 (25%)

For konstanten $\pi = 3.141592\ldots$ gælder identiteten

$$\frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots = \sum_{i=1}^{\infty} \frac{1}{i^2}$$

Følgende algoritme beregner summen af de første n led i ovenstående sum, d.v.s

$$\sum_{i=1}^{n} \frac{1}{i^2}$$

<u>**Algoritme** : $\pi^2/6$</u>

Input : heltal $n \geq 1$

Output : heltal p og q, hvor $\dfrac{p}{q} = \sum_{i=1}^{n} \dfrac{1}{i^2}$

Metode : $p \leftarrow 1;$
$q \leftarrow 1;$
$k \leftarrow 1;$
$\left\{ \dfrac{p}{q} = \sum_{i=1}^{k} \dfrac{1}{i^2} \quad \wedge \quad k \leq n \right\}$ **while** $k < n$ **do**
  $k \leftarrow k + 1;$
  $p \leftarrow p * k * k + q;$
  $q \leftarrow q * k * k$

Spørgsmål a: Angiv hvilke bevisbyrder der skal eftervises i et gyldighedsbevis for algoritmen. □

Spørgsmål b: Eftervis bevisbyrderne fra spørgsmål **a**. □

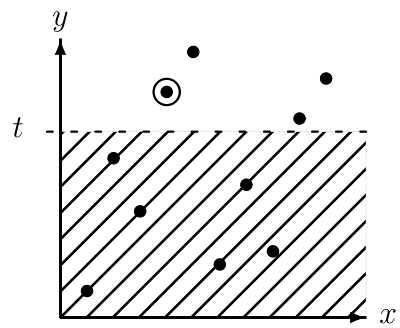
Spørgsmål c: Bevis at algoritmen er korrekt. □

Spørgsmål d: Hvor mange bits kræves til repræsentation af heltallene p henholdsvis q udtrykt som funktion af n i O-notation ? □

Opgave 2 (25%)

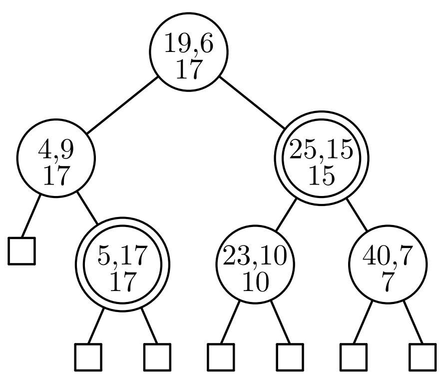
Lad $x_1 < x_2 < \cdots < x_n$ være n heltallige *elementer*, og lad til hvert x_i være tilknyttet en heltallig *værdi* y_i. De n elementer og deres tilknyttede værdier kan opfattes som n punkter $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ i planen.

Opgaven beskæftiger sig med at understøtte operationen $\mathsf{MinElementAbove}(t)$ som returnerer det mindste element x_i, hvis tilhørende værdi y_i er større end eller lig med t, eller meddeler at der ikke findes et sådant element. I nedenstående figur er det returnerede element x-koordinaten af det markerede punkt.



Spørgsmål a: Hvad er $\mathsf{MinElementAbove}(10)$ for punkterne (4,9), (5,17), (19,6), (23,10), (25,15), (40,7) ? □

I det følgende betragtes en udvidelse af rød-sorte søgetræer til opbevaring af elementerne. Lad $v.x$ være elementet gemt i knuden v. Hver knude v udvides til også at gemme værdien $v.y$ knyttet til $v.x$, og den største værdi, $v.y_{\max}$, tilknyttet en knude i undertræet med rod i v. Nedenstående er et udvidet rød-sort søgetræ for punkterne fra spørgsmål a. I knuderne er øverst angivet x, y og nederst $y_{\max}$. Røde knuder er markeret med dobbelt-cirkler.



Spørgsmål b: Beskriv hvordan et udvidet rød-sort søgetræ kan vedligeholdes under indsættelse og sletning af elementer (og tilknyttede værdier) i tid $O(\log n)$. □

Spørgsmål c: Beskriv hvordan $\mathsf{MinElementAbove}(t)$ kan udføres i tid $O(\log n)$. Argumenter for algoritmens udførselstid og korrekthed. □

Opgave 3 (25%)

Givet to strenge $S = S[1]S[2]\cdots S[n]$ og $U = U[1]U[2]\cdots U[k]$, så er U en *super-sekvens* for S hvis S er en delsekvens af U. For eksempel er $\underline{\mathtt{A}}\,\underline{\mathtt{C}}\,\mathtt{A}\,\underline{\mathtt{C}}\,\mathtt{T}\,\mathtt{G}\,\underline{\mathtt{T}}\,\mathtt{A}$ en super-sekvens for $\mathtt{ACCT}$ (understregning angiver en sekvens af matchende positioner).

I det følgende angiver S og T to strenge af længde henholdsvis n og m. En streng U er en *korteste fælles super-sekvens* for S og T, hvis U er en super-sekvens for både S og T, og der findes ikke en kortere streng der er en super-sekvens for både S og T.

Spørgsmål a: Argumenter for at en korteste fælles super-sekvens for S og T har længde højest $n + m$, og for at en korteste fælles super-sekvens ikke altid er entydig. $\square$

Lad $C(i, j)$ betegne længden af en korteste fælles super-sekvens for $S[1]S[2]\cdots S[i]$ og $T[1]T[2]\cdots T[j]$ for $0 \le i \le n$ og $0 \le j \le m$. Det påstås at $C(i, j)$ opfylder følgende rekursionsformel.

$$C(i,j) = \begin{cases} \max\{i,j\} & \text{hvis} \quad (i=0) \vee (j=0) \\ 1 + C(i-1,j-1) & \text{hvis} \quad (i>0) \wedge (j>0) \wedge S[i] = T[j] \\ 1 + \min\{C(i,j-1), C(i-1,j)\} & \text{hvis} \quad (i>0) \wedge (j>0) \wedge S[i] \neq T[j] \end{cases}$$

Spørgsmål b: Argumenter for ovenstående påstand. $\square$

Spørgsmål c: Angiv en algoritme, baseret på dynamisk programmering, der givet S og T finder længden af en korteste fælles super-sekvens. Argumenter for algoritmens udførselstid. $\square$

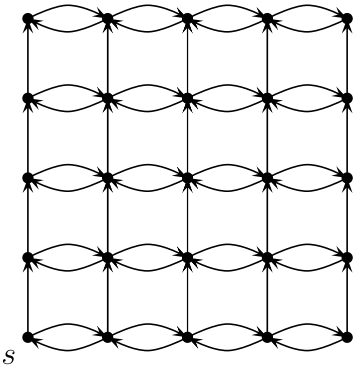
Spørgsmål d: Udvid algoritmen til også at finde en korteste fælles super-sekvens for S og T. $\square$

Opgave 4 (25%)

En *gitter-graf* er en orienteret graf hvor knuderne er arrangeret i k rækker hver indeholdende k knuder, hvor k er et positivt heltal. Lad $v_{i,j}$ betegne den $j$te knude i den $i$te række. Lad $s = v_{1,1}$. En gitter-graf har følgende knuder og kanter:

$$\begin{aligned} V &= \{v_{i,j} \mid 1 \leq i \leq k \wedge 1 \leq j \leq k\} \\ E &= \{(v_{i,j}, v_{i,j+1}) \mid 1 \leq i \leq k \wedge 1 \leq j < k\} \cup \\ &\quad \{(v_{i,j}, v_{i,j-1}) \mid 1 \leq i \leq k \wedge 1 < j \leq k\} \cup \\ &\quad \{(v_{i,j}, v_{i+1,j}) \mid 1 \leq i < k \wedge 1 \leq j \leq k\} \end{aligned}$$

Nedenstående figur viser gitter-grafen for $k = 5$.



I resten af denne opgave antager vi at alle kanter har en ikke-negativ vægt.

Spørgsmål a: Lad n og m betegne henholdsvis antallet af knuder og kanter i en gitter-graf. Udtryk n og m som funktion af k. □

Spørgsmål b: Hvad er udførselstiden for Dijkstra's algoritme for at finde længden af de korteste veje fra $s = v_{1,1}$ til alle de øvrige knuder i en gitter-graf som funktion af k ? □

Spørgsmål c: Beskriv en algoritme der finder længden af de korteste veje fra $s = v_{1,1}$ til alle de øvrige knuder i en gitter-graf i tid $O(m)$. Argumenter for algoritmens udførselstid og korrekthed. □

En *cylinder-graf* er en gitter-graf udvidet med ikke-negative vægtede kanter mellem den venstre og højre knude i hver række, d.v.s. E indeholder også kanterne $(v_{i,1}\ ,\ v_{i,k})$ og $(v_{i,k}\ ,\ v_{i,1})$ for $1 \leq i \leq k$.

Spørgsmål d: Beskriv en algoritme der finder længden af de korteste veje fra $s = v_{1,1}$ til alle de øvrige knuder i en cylinder-graf i tid $O(m)$. Argumenter for algoritmens udførselstid og korrekthed. □