

# Eksamen i Algoritmer og Datastrukturer (NDAA04010U)

Københavns Universitet

2023-04-11

## Instruktioner

Du har 4 timer til at besvare 22 opgaver, beskrevet nedenfor. Karakteren baseres på det samlede antal point (vægten af hver opgave er angivet) og på en helhedsbedømmelse. Opgavesættet har 14 nummererede sider. CLRS refererer til kursets lærebog, Introduction to Algorithms, 3rd edition.

Skriftlige hjælpemidler samt brug af lokalt installeret software er tilladt. Det er *ikke* tilladt at tilgå internet eller at kommunikere med andre.

Multiple-choice opgaver besvares i ITX systemets multiple-choice modul. I hvert multiple-choice spørgsmål er det angivet hvorvidt der er præcis én korrekt svarmulighed eller der skal angives en *mængde* af ét eller flere korrekte svar. Ved retning af multiple-choice opgaver gives der negative point for forkerte svar. Eksempler:

- Opgave X har 8 svarmuligheder, hvoraf 4 er korrekte. Hvert kryds ved en korrekt svarmulighed giver 1 point, og hvert kryds ved en forkert svarmulighed giver -1 point. Hvis du fx sætter kryds ved alle de korrekte svarmuligheder og én forkert svarmulighed får du 3 point. Hvis du ikke sætter nogen krydser får du 0 point, og sætter du krydser ved alle svarmuligheder får du også 0 point.
- Opgave Y har 5 svarmuligheder, hvoraf 1 er korrekt. Kryds ved den korrekte svarmulighed giver 4 point, og kryds ved en forkert svarmulighed giver -1 point.

Svar på skriftlige opgaver afleveres som et Word dokument i ITX systemet. Der er mulighed for at tegne og skrive på tablet, og indsætte i dokumentet. Efter forsiden skal der laves *én side til hvert delspørgsmål, i samme rækkefølge som i opgavesættet*. Angiv delspørgsmålets nummer og bogstav (fx 10.a) men kopiér ikke opgaveteksten ind i besvarelsen. Totalt skal Word dokumentet have 8 sider — hvis du ikke svarer på et delspørgsmål, så lav en tom side.

# Multiple-choice

Dine svar skal indtastes i ITX-systemets multiple-choice modul.

## 1 MergeSort og rekursion (4%)

Hvis  $\text{MERGESORT}(A, p, r)$ , implementeret som i CLRS sektion 2.3, bliver kaldt på 4 elementer (dvs.  $r - p + 1 = 4$ ) laves det første rekursive kald på en liste af 2 elementer, hvorfra det første rekursive kald laves til en liste med 1 element. På det tidspunkt hvor vi når til bunden af rekursionen er der altså 3 kald af  $\text{MERGESORT}$  på rekursionsstakken samtidig. Hvis  $\text{MERGESORT}(A, p, r)$  bliver kaldt på 5 elementer bliver der op til 4 kald samtidig på rekursionsstakken.

Antag nu at  $\text{MERGESORT}(A, p, r)$  bliver kaldt på 42 elementer (dvs.  $r - p + 1 = 42$ ). Hvor stort bliver det største antal samtidige kald til  $\text{MERGE-SORT}$ , dvs. antal kald på rekursionsstakken? Vælg præcis ét svar.

1. 3 samtidige kald.
2. 4 samtidige kald.
3. 5 samtidige kald.
4. 6 samtidige kald.
5. 7 samtidige kald.

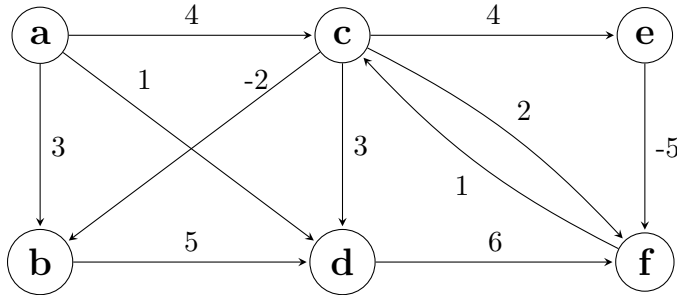
## 2 Korteste-veje træ (4%)

Betragt et korteste veje problem (eng. *single-source shortest-paths problem*) på en orienteret graf  $G = (V, E)$  med vægtfunktion  $w : E \rightarrow \mathbb{R}$  og startknode  $s$ . Hvilke af følgende egenskaber har et korteste-veje træ (eng. *shortest-paths tree*)  $G' = (V', E')$  altid? Vælg ét eller flere korrekte svar.

1. Enhver sti i  $G'$  er en korteste vej i  $G$ .
2. Alle korteste veje i  $G$  svarer til en sti i  $G'$ .
3.  $G'$  er et binært træ med  $s$  som rod.
4.  $G'$  er et mindste udspændende træ (eng. *minimum spanning tree*).
5. Kan beregnes med Dijkstra's algoritme.

### 3 Korteste veje (4%)

Betragt korteste veje problemet (eng. *single-source shortest-paths problem*) med startknode **a** på følgende graf:

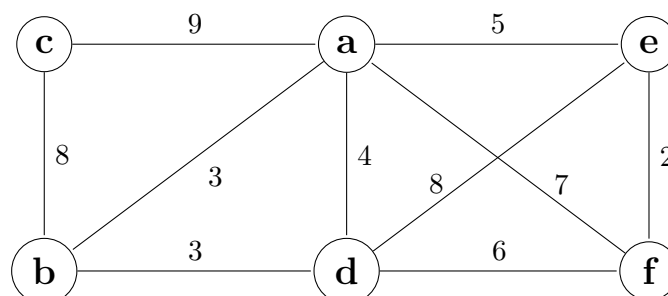


Hvilke kanter er med i korteste-vej træet? Vælg ét eller flere korrekte svar.

1. (a,b)
2. (a,c)
3. (a,d)
4. (b,d)
5. (c,b)
6. (c,d)
7. (c,e)
8. (c,f)
9. (d,f)
10. (e,f)
11. (f,c)

### 4 Prim's algoritme (4%)

Betragt følgende vægtede, uorienterede graf:

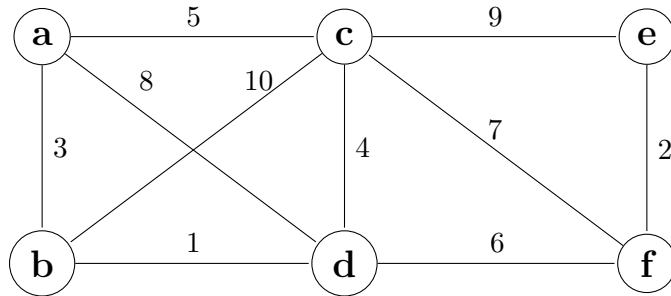


Hvis Prim's algoritme køres på denne graf med start i knude **a**, i hvilken rækkefølge bliver knuderne inkluderet i det mindste udspændende træ? (Med andre ord, i hvilken rækkefølge bliver de taget ud af prioritetskøen?) Vælg præcis ét svar.

1. a, b, d, e, f, c
2. a, b, e, d, f, c
3. a, b, d, c, f, e
4. a, b, e, f, d, c

## 5 Mindste udspændende træ (4%)

Betragt mindste udspændende træ (eng. *minimum spanning tree*) problemet på følgende vægtede, uorienterede graf:

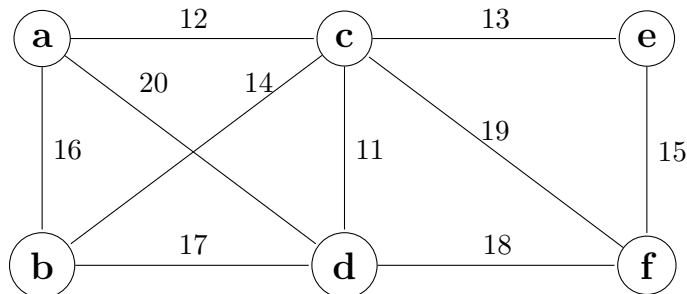


Hvilke kanter er med i et mindste udspændende træ? Vælg ét eller flere korrekte svar.

1. {a,b}
2. {a,c}
3. {a,d}
4. {b,d}
5. {c,b}
6. {c,d}
7. {c,e}
8. {c,f}
9. {d,f}
10. {e,f}

## 6 Snit og sikre kanter (4%)

Betragt følgende uorienterede, vægtede graf med knudemængde  $V = \{a, b, c, d, e, f\}$ :



Betragt kantmængden  $A = \{\{a, c\}, \{c, d\}\}$ , som er en delmængde af et minimalt udspændende træ (eng. *minimum spanning tree*). Vi er interesserede i snit (eng. *cuts*), sikre kanter (eng. *safe edges*) for  $A$ , og snit der respekterer (eng. *respects*) en kantmængde  $A$ . Hvilke af følgende udsagn er sande? Vælg ét eller flere korrekte svar.

1. Lad  $S_1 = \{a, b, c, d\}$ . Snittet  $(S_1, V - S_1)$  respekterer kantmængden  $A$ .
2. Lad  $S_2 = \{a, c, e\}$ . Snittet  $(S_2, V - S_2)$  respekterer kantmængden  $A$ .
3. Lad  $S_3 = \{b, e, f\}$ . Snittet  $(S_3, V - S_3)$  respekterer kantmængden  $A$ .
4.  $\{d, f\}$  er en sikker kant for  $A$ .
5.  $\{c, e\}$  er en sikker kant for  $A$ .
6.  $\{a, d\}$  er en sikker kant for  $A$ .

## 7 Amortiseret analyse (4%)

I denne opgave betragter vi potentialfunktionen der bruges til at analysere en binær tæller i CLRS sektion 17.3. Lad  $\Phi(D_i)$  betegne potentialfunktionens værdi efter  $i$  INCREMENT operationer. Hvilke egenskaber har  $\Phi(D_i)$ ? (Læg mærke til at udsagnene ikke er uafhængige — hvis der gælder  $f(i) = \Omega(i)$  så gælder også  $f(i) = \Omega(\lg i)$ , osv.) Vælg ét eller flere korrekte svar.

1.  $\Phi(D_i) = \Omega(\lg i)$ .
2.  $\Phi(D_i) = \Omega(i)$ .
3.  $\Phi(D_i) = O(1)$ .
4.  $\Phi(D_i) = O(\lg i)$ .
5.  $\Phi(D_i) = O(i)$ .

## 8 Køretid (4%)

Betragt følgende kode i pseudo-kode notationen fra CLRS:

LOOPS( $n$ )

```
1   $r = 0$ 
2  for  $i = 1$  to  $n/2$ 
3      for  $j = n/2 - i$  to  $n$ 
4           $r = r + j$ 
5  return  $r$ 
```

Vi antager at  $n/2$  er et heltal således at den yderste **for**-løkke har  $n/2$  iterationer. Hvilke af følgende er gyldige udsagn om køretiden  $T(n)$  af LOOPS( $n$ )? (Læg mærke til at udsagnene ikke er uafhængige — hvis der gælder  $f(n) = O(n)$  så gælder også  $f(n) = O(n \lg n)$ , osv.) Vælg ét eller flere korrekte svar.

1.  $T(n) = O(n)$ .
2.  $T(n) = O(n \lg n)$ .
3.  $T(n) = O(n^2)$ .
4.  $T(n) = \Omega(n)$ .
5.  $T(n) = \Omega(n \lg n)$ .
6.  $T(n) = \Omega(n^2)$ .

## 9 Rekursionsligninger (4%)

Hvilke af disse rekursionsligninger har løsningen  $T(n) = \Theta(n \lg n)$ ? Antag at  $T(n) = 1$  for  $n \leq 1$ . Vælg ét eller flere korrekte svar.

1.  $T(n) = 2T(\lfloor n/2 \rfloor) + n^2$ .
2.  $T(n) = 2T(\lfloor n/2 \rfloor) + 3n$ .
3.  $T(n) = 2T(\lfloor n/3 \rfloor) + 2n$ .
4.  $T(n) = 3T(\lfloor n/2 \rfloor) + n \lg n$ .
5.  $T(n) = 3T(\lfloor n/3 \rfloor) + n$ .
6.  $T(n) = T(n-1) + \lg n$ .

## 10 Del og hersk (4%)

Antag at du har en rekursiv algoritme  $A$ , og lad  $n$  betegne størrelsen på algoritmens input  $X$ . Hvis  $n = 1$  beregnes resultatet  $A(X)$  i konstant tid. For  $n > 1$  bruger algoritmen først  $O(n \lg n)$  tid på at beregne fire inputs  $X_1, X_2, X_3, X_4$ , hver af størrelse  $\lfloor n/2 \rfloor$ , og  $A(X_1), A(X_2), A(X_3), A(X_4)$  beregnes rekursivt. Endelig kombineres de rekursive svar til et output  $A(X)$  i tid  $O(n^2)$ .

Hvilken af disse rekursionsligninger beskriver køretiden  $T(n)$  af  $A$  på input  $X$ ? Vælg præcis ét svar.

1.  $T(n) = 4T(\lfloor n/2 \rfloor) + O(n \lg n)$ .
2.  $T(n) = 2T(\lfloor n/4 \rfloor) + O(n \lg n)$ .
3.  $T(n) = 4T(\lfloor n/2 \rfloor) + O(n^2)$ .
4.  $T(n) = 2T(\lfloor n/4 \rfloor) + O(n^2)$ .

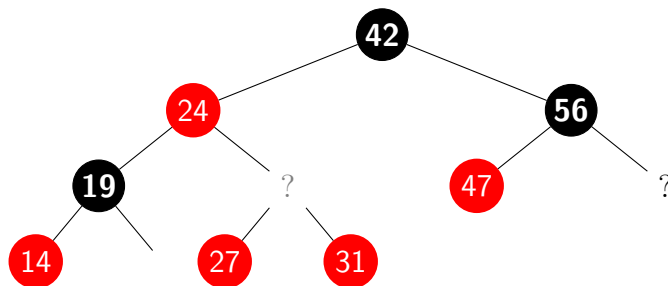
## 11 Fibonacci hobe (4%)

Betragt en Fibonacci hob  $H$  (eng. *Fibonacci heap*), som beskrevet i CLRS kapitel 19, hvorpå der udføres  $n_1$  INSERT operationer,  $n_2$  EXTRACT-MIN operationer, og  $n_3$  DECREASE-KEY operationer. Det totale antal operationer er  $n = n_1 + n_2 + n_3$ . Hvilke af følgende udsagn er altid sande? Vælg ét eller flere korrekte svar.

1. Den samlede worst case tid for operationerne er  $\Omega(n \lg n)$ .
2. Den samlede worst case tid for operationerne er  $O(n \lg n)$ .
3. Den samlede worst case tid for operationerne er  $O(n_1 + n_3 + n_2 \lg n)$ .
4.  $H$  har  $n_1 - n_2 - n_3$  knuder.
5.  $H$  har  $m(H) = n_1 - n_2$  markerede knuder.

## 12 Binære søgetræer (4%)

Betragt et rød-sort binært søgetræ  $T$  (eng. *red-black binary search tree*), som beskrevet i CLRS kapitel 13. Betrakt nedenstående træ, hvor alle blade er udeladt (som på CLRS figur 13.1.c) og to indre knuder er uspecificerede:

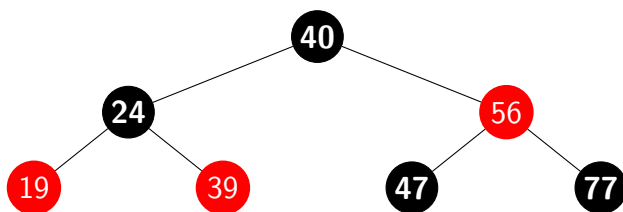


Hvilke knuder er mulige på mindst én af positionerne markeret med ?, hvis træet skal overholde invarianterne for et rød-sort binært søgetræ? Vælg ét eller flere korrekte svar.

1. En rød knude med nøglen 28.
2. En rød knude med nøglen 44.
3. En rød knude med nøglen 58.
4. En sort knude med nøglen 25.
5. En sort knude med nøglen 30.
6. En sort knude med nøglen 66.

### 13 Binære søgetræer (4%)

Betragt dette rød-sort binære søgetræ (igen med bladene udeladt):



Antag at vi bruger indsættelsesalgoritmen RB-INSERT fra CLRS kapitel 13 til at indsætte nøglen 42. Hvad sker med træet? Vælg præcis ét svar.

1. 42 indsættes som en sort knude under knuden 47.
2. 42 indsættes som en rød knude under knuden 47.
3. 42 indsættes som ny sort rodnode og 40 bliver barn til knuden 39.
4. 42 indsættes som ny rød rodnode og 40 bliver barn til knuden 39.
5. Ingen af ovenstående.



## 14 Disjunkte mængder (4%)

Vi betragter trærepræsentationer af disjunkte mængder (eng. *disjoint sets*), som beskrevet i CLRS sektion 21.3, ved brug af heuristikkerne *union by rank* og *path compression*. Antag at MAKE-SET er kørt  $n$  gange og lad  $k$  være en variabel. Hvilke af følgende udsagn er altid sande? Vælg ét eller flere korrekte svar.

1. Tiden for et kald til  $\text{UNION}(x, y)$  er  $O(\lg n)$ .
2.  $k$  kald til  $\text{FIND-SET}$ (42) lige efter hinanden tager tid  $O(k + \lg n)$ .
3. Kaldene  $\text{MAKE-SET}(x_1), \dots, \text{MAKE-SET}(x_k)$  tager total tid  $O(k)$ .
4. En  $\text{LINK}(x, y)$  operation øger den maksimale værdi af *rank* i træerne for  $x$  og  $y$  med 1.
5. I værste fald tager en  $\text{FIND-SET}$  operation  $\Omega(n)$  tid.
6.  $\text{UNION}(x, x)$  tager konstant tid.

## 15 Invarianter (4%)

Betragt følgende funktion, i pseudo-kode notationen fra CLRS, hvor input  $A$  er en tabel af heltal og  $n$  er længden af  $A$ .

$\text{SECOND-SMALLEST}(A, n)$

```
1   $m = A[1]$ 
2   $r = \infty$ 
3  for  $i = 2$  to  $n$ 
4      if  $A[i] < m$ 
5           $r = m$ 
6           $m = A[i]$ 
7      elseif  $A[i] < r$ 
8           $r = A[i]$ 
9  return  $r$ 
```

Hvilke udsagn gyldige løkke-invarianter ved starten af hver iteration i **for**-løkken? Vælg ét eller flere korrekte svar.

1.  $m = \min(A[1], \dots, A[i - 1])$ .
2.  $m = \min(A[1], \dots, A[n])$ .
3.  $r \geq m$ .
4.  $m \geq 0$ .
5. Mindst 1 element i  $A[1], \dots, A[i - 1]$  er mindre end eller lig med  $r$ .
6. Højst 2 elementer i  $A[1], \dots, A[i - 1]$  er mindre end eller lig med  $r$ .

## 16 Valg af datastruktur (4%)

Antag at vi gerne vil skabe en datastruktur `VENTETID`, der indeholder en mængde af  $n$  afgangstidspunkter (ugedag og klokkeslet, fx for en bus). For enkelheds skyld skal der blot holdes styr på én rute og ét stoppested, så den eneste information er afgangstidspunkter. Datastrukturen skal understøtte to operationer: `INSERT( $x$ )`, der indsætter et nyt afgangstidspunkt, og `HVORLÆNGE( $x$ )` der returnerer antal minutter fra tidspunkt  $x$  til den næste afgang.

Hvad er det mest passende valg af datastruktur til løsning af dette problem blandt nedenstående, og hvilken worst-case (amortiseret) køretid for `HVORLÆNGE( $x$ )` bliver resultatet af dette valg? Udtryk svaret som funktion af  $n$ . Vælg præcis ét svar.

1. Disjunkte mængder (eng. *disjoint sets*), tid  $O(1)$ .
2. Disjunkte mængder (eng. *disjoint sets*), tid  $O(\lg n)$ .
3. Fibonacci hob (eng. *Fibonacci heap*), tid  $O(\lg n)$ .
4. Fibonacci hob (eng. *Fibonacci heap*), tid  $O(1)$ .
5. Rød-sort binært søgetræ (eng. *red-black binary search tree*), tid  $O(\lg n)$ .
6. Rød-sort binært søgetræ (eng. *red-black binary search tree*), tid  $O(1)$ .

## 17 Korrekthedsargumenter (4%)

Betragt følgende problem, kaldet **MaxProduct**:

Givet en liste af tal  $x_1, \dots, x_n \in \mathbf{R}$  (kan være både positive og negative), find en delmængde  $I^* \subseteq \{1, \dots, n\}$  hvor produktet  $\prod_{i \in I^*} x_i$  er så stort som muligt, dvs. for alle mængder  $I \subseteq \{1, \dots, n\}$  skal gælde at  $\prod_{i \in I} x_i \leq \prod_{i \in I^*} x_i$ .

**Eksempel.** På input  $x_1 = -4$ ,  $x_2 = -0.5$ ,  $x_3 = 0.5$ ,  $x_4 = 0.5$ ,  $x_5 = 1.5$ ,  $x_6 = 6$  kan vi vælge  $I = \{5, 6\}$  og få et produkt på  $1.5 \cdot 6 = 9$ , men et bedre valg er  $I = \{1, 2, 5, 6\}$  som giver produktet  $(-4) \cdot (-0.5) \cdot 1.5 \cdot 6 = 18$ . Den tomme mængde  $I = \emptyset$  giver per definition  $\prod_{i \in \emptyset} x_i = 1$ .

Hvad gælder altid for en optimal løsning  $I^*$ ? Vælg ét eller flere korrekte svar.

1.  $I^*$  indeholder *ikke* noget indeks  $i$  hvor  $x_i$  er mellem 0 og 1 ( $x_i \in (0, 1)$ ).
2.  $I^*$  indeholder *alle* indekser  $i$  hvor  $x_i < 0$ .
3.  $I^*$  indeholder *alle* indekser  $i$  hvor  $x_i > 1$ .
4.  $I^*$  indeholder *alle* indekser  $i$  hvor  $|x_i| > 1$  (i.e., absolutværdien af  $x_i$  er større end 1).

## 18 Dynamisk programmering (4%)

En *stigende delsekvens* af en tabel  $A$ , der indeholder heltal  $A[1], \dots, A[n]$ , er en delmængde af indekser  $1 \leq i_1 < i_2 < \dots < i_k \leq n$  således at  $A[i_1] < A[i_2] < \dots < A[i_k]$ . Længden af delsekvensen er antal indekser,  $k$ . En *ikke-aftagende delsekvens* defineres på samme måde bortset fra at kravet er  $A[i_1] \leq A[i_2] \leq \dots \leq A[i_k]$ . Betragt følgende funktion, i pseudo-kode notationen fra CLRS, hvor input  $A$  er en tabel af heltal.

MAXIMUMMYSTERY( $A, n$ )

```
1  let  $r[1 \dots n]$  be a new array
2   $r[1] = 1$ 
3  for  $j = 2$  to  $n$ 
4       $q = 1$ 
5      for  $i = 1$  to  $j - 1$ 
6          if not  $A[j] < A[i]$ 
7               $q = \max(q, r[i] + 1)$ 
8       $r[j] = q$ 
9  return  $r[n]$ 
```

Hvad beregner proceduren MAXIMUMMYSTERY? Vælg præcis ét svar.

1. Indekserne i en længste stigende delsekvens.
2. Indekserne i en længste ikke-aftagende delsekvens.
3. Længden af en længste stigende delsekvens.
4. Længden af en længste ikke-aftagende delsekvens.

## Skriftlige opgaver

### 19 Længste fælles delsekvens (4%)

Vi betragter algoritmen til længste fælles delsekvens (eng. *longest common subsequence*) i CLRS sektion 15.4.

a) Tegn tabellen  $c$  som algoritmen udfylder på følgende input:

$$x = \langle N, A, A, N \rangle \text{ og } y = \langle A, N, N, A \rangle.$$

Tilføj pile ( $\uparrow$ ,  $\nwarrow$ ,  $\leftarrow$ ), der forklarer tallene i tabellen, som i CLRS Figur 15.8 og angiv den længste fælles delsekvens. Skriv dit svar på side 2 i Word dokumentet.

### 20 Dynamisk programmering (10%)

Vi betragter følgende variant af længste fælles delsekvens (eng. *longest common subsequence*) problemet fra CLRS sektion 15.4. Input er to vektorer  $x = \langle x_1, \dots, x_m \rangle \in \Sigma^m$  og  $y = \langle y_1, \dots, y_n \rangle \in \Sigma^n$ . I lighed med længste fælles delsekvens er målet at finde en optimal vektor  $z = \langle z_1, \dots, z_k \rangle$ , men i stedet for at maksimere længden af  $z$  skal *vægten* af tegnene i  $z$  maksimeres. Vi antager at vægtene er ikke-negative. Vægten er givet ved en datastruktur  $w$ , således at vægten af  $z_i \in \Sigma$ , der betegnes med  $w[z_i]$ , kan findes i konstant tid. Vægten af  $z$  er summen af vægte,  $\sum_{i=1}^k w[z_i]$ , og målet er et finde en fælles delsekvens med størst mulig vægt.

**Eksempel.** Antag at vi har  $x = \langle A, H, A, A \rangle$ ,  $y = \langle A, A, R, G, H \rangle$ , og  $w[A] = 2$ ,  $w[G] = 8$ ,  $w[R] = 10$ ,  $w[H] = 5$ . Delsekvensen  $\langle A, A \rangle$  med vægt  $2 + 2 = 4$  findes i både  $x$  og  $y$ . Delsekvensen  $\langle A, H \rangle$  med vægt  $2 + 4 = 6$  findes også i både  $x$  og  $y$ , og er den delsekvens, der har størst vægt.

a) Skriv en rekursionsligning for den optimale vægt af en fælles delsekvens af  $x$  og  $y$ . Skriv dit svar på side 3 i Word dokumentet.

b) Lav en analyse af køretiden af den algoritme, der bruger rekursionsligningen og dynamisk programmering til at beregne den optimale vægt. Analysen skal give en øvre grænse på køretiden som funktion af  $n$  og  $m$  i store- $O$  notation, der er så præcis som mulig. Skriv dit svar på side 4 i Word dokumentet.

c) Beskriv hvordan algoritmen kan modificeres til at håndtere det tilfælde hvor vægte kan være negative uden at påvirke den asymptotiske køretid. Argumentér for køretid og korrekthed af din løsning. Skriv dit svar på side 5 i Word dokumentet.

## 21 Induktionsbeviser (6%)

I denne opgave antager vi at  $T$  er en sorteret tabel med  $n > 1$  elementer således at for  $i < j$  gælder  $T[i] \leq T[j]$ . Antag at  $1 \leq p \leq r \leq n$ . Betragt følgende funktion, i pseudo-kode notationen fra CLRS:

WHILEMYSTERY( $x, T, p, r$ )

```
1  low = p
2  high = max(p, r + 1)
3  while low < high
4      mid =  $\lfloor (low + high)/2 \rfloor$ 
5      if  $x \leq T[mid]$ 
6          high = mid
7      else
8          low = mid + 1
9  return high
```

a) Lad  $high_i$  og  $low_i$  betegne værdien af variablene  $high$  og  $low$  efter  $i$  iterationer af **while** løkken. Bevis ved induktion at  $high_i - low_i \leq n/2^i$ . (Du skal bruge antagelsen  $1 \leq p \leq r \leq n$ .) Der lægges vægt på, at argumentationen er klar og koncis. Skriv dit svar på side 6 i Word dokumentet.

## 22 Grådige algoritmer (8%)

Vi betragter igen problemet fra opgave 17, gengivet hér:

Givet en liste af tal  $x_1, \dots, x_n \in \mathbf{R}$  (kan være både positive og negative), find en delmængde  $I^* \subseteq \{1, \dots, n\}$  hvor produktet  $\prod_{i \in I^*} x_i$  er så stort som muligt, dvs. for alle mængder  $I \subseteq \{1, \dots, n\}$  skal gælde at  $\prod_{i \in I} x_i \leq \prod_{i \in I^*} x_i$ .

**Eksempel.** På input  $x_1 = -4$ ,  $x_2 = -0.5$ ,  $x_3 = 0.5$ ,  $x_4 = 0.5$ ,  $x_5 = 1.5$ ,  $x_6 = 6$  kan vi vælge  $I = \{5, 6\}$  og få et produkt på  $1.5 \cdot 6 = 9$ , men et bedre valg er  $I = \{1, 2, 5, 6\}$  som giver produktet  $(-4) \cdot (-0.5) \cdot 1.5 \cdot 6 = 18$ . Den tomme mængde  $I = \emptyset$  giver per definition  $\prod_{i \in \emptyset} x_i = 1$ .

Et mulig tilgang til at løse **MaxProdukt** er at grådigt føje elementer til  $I$ , ét ad gangen. Antag at input er sorteret således at  $x_1 \leq x_2 \leq \dots \leq x_n$ .

a) Argumentér for at for en optimal løsning  $I^*$  vil der altid vil være et lige antal elementer i mængden  $I_-^* = \{i \in I^* \mid x_i < 0\}$ , dvs.  $|I_-^*|$  er deleligt med 2. Skriv dit svar på side 7 i Word dokumentet.

**b)** Foreslå en effektiv algoritme til at finde en optimal løsning  $I^*$ . Argumentér for korrekthed og køretid. Der lægges vægt på, at argumentationen er klar og koncis. (Det er *ikke* et krav at argumentere via “greedy choice property” og “optimal substructure”.) Skriv dit svar på side 8 i Word dokumentet.