

## i Information

- Försök på alla uppgifter! Uppgifterna är inte ordnade på något speciellt sätt.
- Det är ditt ansvar att övertyga oss att du besitter den kunskap som efterfrågas.
- Det är viktigt att du löser den *givna* uppgiften!
- På sista "frågan" i denna tentamen finns plats att skriva kommentarer om du vill förtydliga något kring dina svar på någon fråga. "Frågan" ger inga poäng.
- Lärarna kommer att besöka tentasalen nån gång under provets gång för att svara på eventuella oklarheter. Det går även nå oss på telefon.
- Endast de som är underkända på tidigare tentamina får skriva denna tentamen. Plussning (dvs att skriva om efter du blivit godkänd i syfte att få högre betyg) är inte tillåtet.

**Betyg:** max poäng är 40

- För betyg **3** (godkänt) krävs **20** poäng.
- För betyg **4** krävs **26** poäng.
- För betyg **5** krävs **32** poäng.

**Lycka till!**

**Ola, Niclas, Vicenç**

## 1 Terminologi (5p)

Nedan kommer 10 sant/falskt frågor. Rätt svar ger 1p, felaktigt svar -1p och inget svar alls ger 0p. Du kan få 0-10 p på denna uppgift (alltså inte minusresultat). Med datatyp avses en abstrakt datatyp med definition enligt boken.

**En ordnad datatyp kan vara osorterad.**

☐ Sant



☐ Falskt

**Kö är en heterogen datatyp.**

☐ Sant

☐ Falskt



**En datatyp vars objekt inte består av element som i sin tur är datatypsobjekt kallas konstruerad datatyp. Tex heltal.**

☐ Sant

☐ Falskt



**En konkret datatyp kan vara implementerad.**

☐ Sant



☐ Falskt

**Man kan prata om en lista som innehåller tre element och ett värde.**

☐ Sant



☐ Falskt

**En heterogen datatyp är en sammansatt datatyp där elementen är av olika datatyper.**

☐ Sant



☐ Falskt

**En datatyp som är konstruerad av en implementerad datatyp är implementerad.**

☐ Sant



☐ Falskt

**Mängd är en oordnad datatyp.**

☐ Sant



☐ Falskt

**En datatyp som finns tillgänglig i en given maskinvara eller programspråk kallas abstrakt datatyp.**

☐ Sant

☐ Falskt



**Enkel datatyp är en term som används när man beskriver, diskuterar eller använder en datatyp utan att ta hänsyn till om eller hur den är realiserad i ett programspråk/hårdvara.**

☐ Sant

☐ Falskt



---

Totalpoäng: 10

## 2 Sorteringsalgoritmer (4p)

Nedanstående frågor ger +0.5p för rätt svar, -0.5 för felaktigt. Lägsta poäng är 0p. Alla rätt ger 4p.

Vilken/vilka av följande sorteringsalgoritmer har en **medelfallskomplexitet** på  $O(n \log n)$ ? Välj ett eller flera alternativ.

- ☐ Bubbelsortering
- ☐ Quicksort ✓
- ☐ Instickssortering
- ☐ Urvalssortering
- ☐ Heapsort ✓
- ☐ Mergesort ✓

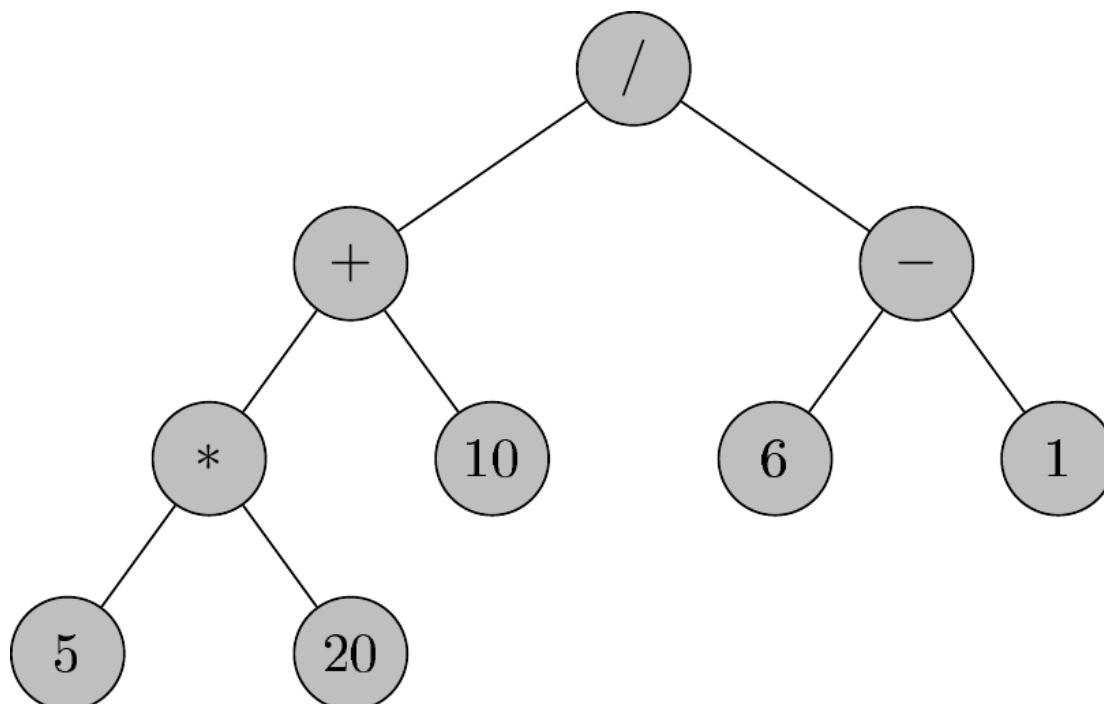
Vilken/vilka av följande sorteringsalgoritmer har en **värstafallskomplexitet** på  $O(n^2)$ ? Välj ett eller flera alternativ.

- ☐ Instickssortering ✓
- ☐ Urvalssortering ✓
- ☐ Bubbelsortering ✓
- ☐ Heapsort
- ☐ Mergesort
- ☐ Quicksort ✓

---

Totalpoäng: 4

3



Detta träd representerar ett algebraiskt uttryck som innehåller heltal och binära operationer. För att evaluera en nod  $n$  med en binär operator som etikett måste vi först evaluera vänster delträd (delträdet vars rot är vänsterbarn till  $n$ ) och höger delträd (delträdet vars rot är högerbarn till  $n$ ). Detta ger operanden  $a$  från vänsterbarnet och  $b$  från högerbarnet. Till slut evaluerar vi  $a \text{ op } b$ , där  $\text{op}$  är operatoren som är etikett till  $n$ . Evaluering av ett löv  $p$  ger heltalet som är etikett till  $p$ .

### 3.1 Evaluering (1p)

Vilket numeriskt värde får du om du evaluerar det binära trädet till ovan?

(22).

Totalpoäng: 1

### 3.2 Traversering för utskrift (1p)

Vilken typ av traversering av trädet behöver vi använda om vi vill **skriva ut** det aritmetiska uttrycket som en sträng (dvs. som vi vanligtvis ser aritmetiska uttryck)?

- ☐ Bredden-först
- ☐ Djupet-först, preorder
- ☒ Djupet-först, inorder
- ☐ Djupet-först, postorder



---

Totalpoäng: 1

### 3.3 Utskrift (2p)

Vad får vi för utskrift om vi applicerar traverseringsalgoritmen från förra uppgiften på trädet?

1. Ert svar ska innehålla paranteser för att indikera i vilken ordning operationerna ska genomföras.
2. Skriv inga paranteser runt etiketterna i löven.
3. Med undantag för punkt 2, skriv alltid ut paranteserna, även om dom inte "behövs" pga prioritetsordningen mellan operationerna.

Skriv resultatet här:


$((5*20)+10)/(6-1), (((5*20)+10)/(6-1)))$

---

Totalpoäng: 2

### 3.4 Unära operationer (1p)

Trädet i vårt exempel innehåller bara binära operationer. Antag att vi också vill kunna representera *unära* operationer (unärt minus,  $e^x$ ,  $\sin x$ , osv.). Kan vi behålla samma datatyp eller måste vi byta till någon annan?

- ☐ Vi måste byta datatyp till ett ordnat träd.
- ☐ Vi måste byta datatyp till en DAG.
- ☒ Vi kan behålla det binära trädet som datatyp, om vi låter den unära operanden representeras av vänster delträd. 

---

Totalpoäng: 1

**Gränssnitt för Riktad lista:**

$s = \text{Empty}()$	Returnera en tom riktad lista.
$p = \text{First}(s)$	Returnera den första positionen i den riktade listan $s$ .
$q = \text{Next}(p, s)$	Returnera positionen $q$ omedelbart efter positionen $p$ i $s$ .
$\text{Isend}(p, s)$	Returnera <i>True</i> om positionen $p$ är sista positionen i $s$ .
$v = \text{Inspect}(p, s)$	Returnera värdet lagrat på positionen $p$ i listan $s$ .
$(p, s) = \text{Insert}(v, p, s)$	Stoppa in värdet $v$ i listan $s$ på positionen omedelbart före $p$ . Returnera den modifierade listan samt positionen för det nya elementet i den nya listan.
$(p, s) = \text{Remove}(p, s)$	Ta bort elementet på positionen $p$ i listan $s$ . Returnera den modifierade listan samt positionen omedelbart efter det borttagna elementet.



## 4.1 Algoritmer i pseudokod (4p+4p)

I denna sektion ska ni skriva två versioner av en algoritm/funktion som filtrerar bort upprepade värden ur en Riktad lista. Gränssnittet för den abstrakta datatypen Riktad lista beskrivs i panelen bredvid. Använd gränssnittet för Riktad lista till att skriva en funktion som tar en riktad lista  $s$  som parameter och filtrerar bort alla dubletter. Listan  $s$  är osorterad och kan innehålla samma värde en eller flera gånger.

Det finns en funktion  $Equal(a,b)$  som returnerar  $True$  om värdet  $a$  är lika med värdet  $b$ , annars  $False$ . Använd funktionen för att testa för likhet mellan elementvärden i er algoritm.

Den första versionen *Filter1* av algoritmen ska lämna  $s$  oförändrad och returnera en riktad lista  $t$  med alla värden i  $s$ , upprepade endast en gång. Algoritmen ska ha följande huvud:

**Algorithm Filter1(s: Directed list)**

*// Return a copy of s with all duplicate values removed*

Den andra versionen *Filter2* av algoritmen ska modifiera  $s$  så att  $s$  bara innehåller en kopia av varje värde och sedan returnera den modifierade listan  $s$ . Algoritmen ska ha följande huvud:

**Algorithm Filter2(s: Directed list)**

*// Modify and return s with all duplicate values removed*

Det är möjligt att *modularisera* lösningen, dvs. att identifiera en eller flera delalgoritmer och beskriva dem separat (som t.ex. Algorithm ZZZ). Ge dom i så fall lämpliga namn och kommentarer. Det är tillåtet att anropa samma delalgoritm från *Filter1* och *Filter2*. Ni behöver i så fall bara lista delalgoritmen en gång.

Skriv pseudokod för *Filter1*, *Filter2* och eventuella delalgoritmer/funktioner i fältet längst ner.

Det som kommer att bedömas är

- att pseudokoden löser uppgiften under de givna förutsättningarna,
- att koden är korrekt indenterad,
- att koden är rimligt kommenterad,
- om koden är modulariserad, och
- om koden har optimal komplexitet.

För fulla poäng ska koden vara modulariserad och ha optimal komplexitet, inklusive att listorna inte traverseras onödigt mycket.

1

## Lösningssförslag

```
Algorithm FindFrom(v: Value, p: Pos, s: Directed list)
// Return True if the value v is stored in the directed list s from
// the position p and onwards.

while not Isend(p, s) do
    if Equal(v, Inspect(p, s)) then
        // We found the element
        return True
    // Otherwise, advance in the list
    p = Next(p, s)
// We've reached the end without finding the element
return False

Algorithm Filter1(s : Directed list)
// Return a copy of s with all duplicate values removed

// Start with empty output list
t = Empty()
// Iterate over input list
p = First(s)
while not Isend(p, s) do
    v = Inspect(p, s)
    // Check if the value is already in output list
    if not FindFrom(v, First(t), t) then
        // If not, insert it first in output list
        t = Insert(v, First(t), t)
    // Advance in input list
    p = Next(p, s)
// Return the list copy
return t

Algorithm Filter2(s: Directed list)
// Modify and return s with all duplicate values removed

// Traverse input list
p = First(s)
while not Isend(p, s) do
    v = Inspect(p, s)
    // Check if the value reappears somewhere later in the input list
    if FindFrom(v, Next(p, s), s) then
        // If yes, remove the value.
        // The position p is returned as the position immediately after the removed element.
        (s, p) = Remove(s, p)
    else
        // If the value does not reappear, advance in the input list
        p = Next(p, s)
// Return the modified list
return s
```

## 4.2 Frågor Filter1 (1.5p)

Nedan följer frågor kring din implementation av **Filter1**. För att få poäng krävs normalt att Filter1 uppfyller huvudkraven i förra frågan.

Om  $n$  är antalet element i in-listan, vad har din algoritm **Filter1** för komplexitet?

- ☐  $O(1)$
- ☐  $O(\log n)$
- ☐  $O(n)$
- ☐  $O(n \log n)$
- ☒  $O(n^2)$
- ☐  $O(n^3)$



Om listan som skickas in saknar dubletter, hur är utlistan som returneras av din **Filter1** ordnad?

- ☐ Samma ordning som i inlistan.
- ☒ Omvänd ordning som i inlistan.
- ☐ Sorterad.
- ☐ Inget av de övriga alternativen.



Om listan som skickas in har en tripplett, dvs. ett värde som repeteras tre gånger, vilken kopia av värdet återfinns i listan som returneras av din **Filter1**?

- ☒ Den första
- ☐ Den andra
- ☐ Den tredje
- ☐ Det varierar



---

Totalpoäng: 1.5

### 4.3 Frågor Filter2 (1.5p)

Nedan följer frågor kring din implementation av **Filter2**. För att få poäng krävs normalt att Filter2 uppfyller huvudkraven i förra frågan.

Om  $n$  är antalet element i in-listan, vad har din algoritm **Filter2** för komplexitet?

- ☐  $O(1)$
- ☐  $O(\log n)$
- ☐  $O(n)$
- ☐  $O(n \log n)$
- ☐  $O(n^2)$
- ☐  $O(n^3)$



Om listan som skickas in saknar dubletter, hur är utlistan som returneras av din **Filter2** ordnad?

- ☐ Samma ordning som i inlistan.
- ☐ Omvänd ordning som i inlistan.
- ☐ Sorterad.
- ☐ Inget av de övriga alternativen.



Om listan som skickas in har en tripplett, dvs. ett värde som repeteras tre gånger, vilken kopia av värdet återfinns i listan som returneras av din **Filter2**?

- ☐ Den första
- ☐ Den andra
- ☐ Den tredje
- ☐ Det varierar



---

Totalpoäng: 1.5

5

 $M$  $P$ 

	A	B	C	D	E	F	G	H
A	0	12	10	2	6	$\infty$	$\infty$	$\infty$
B	12	0	12	10	8	$\infty$	$\infty$	$\infty$
C	10	12	0	12	4	$\infty$	$\infty$	$\infty$
D	2	10	12	0	8	$\infty$	$\infty$	$\infty$
E	6	8	4	8	0	$\infty$	$\infty$	$\infty$
F	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	0	10	9
G	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	10	0	1
H	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	9	1	0

A	B	C	D	E	F	G	H
-	D	E	A	A	-	-	-
D	-	E	B	B	-	-	-
E	E	-	A	C	-	-	-
D	D	E	-	A	-	-	-
E	E	E	A	-	-	-	-
-	-	-	-	-	-	H	F
-	-	-	-	-	H	-	G
-	-	-	-	-	H	H	-

Matriserna  $M$  och  $P$  visar resultatet av Floyd:s algoritm på en graf. Matrisen  $M$  innehåller kortaste avståndet mellan noderna,  $P$  är föregångarmatrisen. Använd  $M$  och  $P$  till att besvara frågorna nedan.

### 5.1 Kortaste vägen (3p)

Nedan följer frågor kring kortaste vägen mellan två noder baserat på matriserna  $M$  och  $P$ . För varje fråga, svara med nodernas namn, separerade med streck/minustecken. Exempel: Om den kortaste vägen mellan noderna X och Y går via noderna P och Q, skriv **X-P-Q-Y**. Ange också totallängden på den efterfrågade vägen. Om väg saknas, skriv ett streck/minustecken (-) för vägen och **inf** för längden.

Vilken är kortaste vägen mellan noderna C och D?  (C-E-A-D)

Hur lång är kortaste vägen mellan noderna C och D?  (12)

Vilken är kortaste vägen mellan noderna E och F?  (-)

Hur lång är kortaste vägen mellan noderna E och F?  (inf)

Vilken är kortaste vägen mellan noderna F och G?  (F-H-G)

Hur lång är kortaste vägen mellan noderna F och G?  (10)

Totalpoäng: 3

## 5.2 Lägsta gradtal (2p)

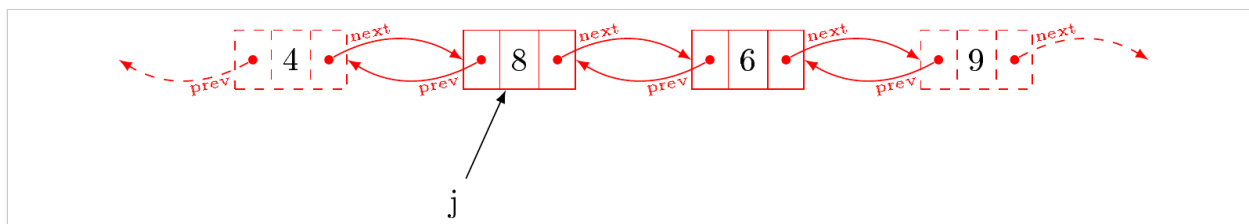
Alla bågar i grafen har unika heltalsvikter. Största vikten är 10.

Vilket är det lägsta gradtalet för en nod i grafen?  (1)

Ange namnet på **en** nod med detta gradtal:  (C, F, G)

Totalpoäng: 2

## 6.1 Antal länkar (1p)



En fältbaserad Bubblesort och motsvarande list-baserade algoritm beskrivs ~~till vänster~~ <sup>på nästa sida</sup>. Den list-baserade implementationen använder 2-celler för att bygga upp en dubbel-länkad struktur och har extraceller i början och slutet av listan, se figur ovan. Du ska skriva funktionen *List-Swap*(*a*, *j*) som byter plats på *elementen* i listan genom att manipulera länkarna direkt. I detta fall ska du alltså *inte* använda gränsytan till Lista. Du får heller inte byta plats på elementvärdena.

Hur många länkar måste ändras? Skriv svaret här:  (6).

Totalpoäng: 1

bubble-list.pse

```
Algorithm List-Swap(a: List, j: Pos)
// Swap elements at positions j and its successor. Return the
// modified list and the lowest position after the swap.

// Your code goes here, ends with return something.

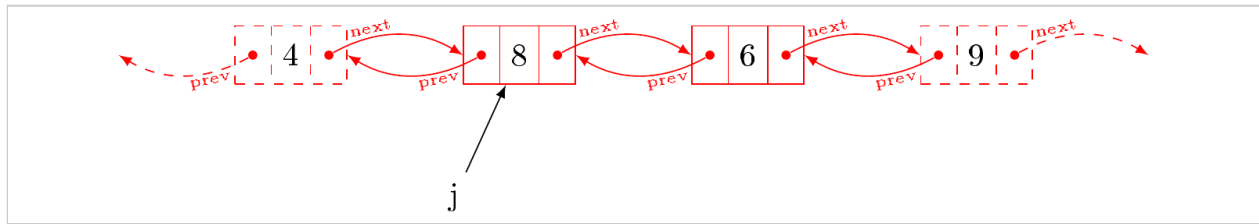
Algorithm List-Bubblesort(a: List)
// Perform bubblesort on the list a.
// Assumes a has at least two elements.
repeat
    swapped <- False
    j <- First(a)
    // Next(j, a) = End(a) corresponds to stop at High(a)-1
    while not Next(j, a) = End(a) do
        if Inspect(j, a) > Inspect(Next(j, a), a) then
            // Swap values at j and its successor
            (a, j) = List-Swap(a, j)
            swapped <- True
        j = Next(j, a)
until not swapped
return a
```

bubble-array.pse

```
Algorithm Array-Swap(a: Array, j: Pos)
// Swap values at positions j and its successor. Return the
// modified array and the lowest position after the swap.
temp <- a[j+1]
a[j+1] <- a[j]
a[j] <- temp
return (a, j)

Algorithm Array-Bubblesort(a: Array)
// Perform bubblesort on the Array a.
// Assumes a has at least two elements.
repeat
    swapped <- False
    for j from Low(a) to High(a)-1 do
        if a[j] > a[j+1] then
            // Swap values at j and its successor
            (a, j) = Array-Swap(a, j)
            swapped <- True
until not swapped
return a
```

## 6.2 Swap för länkad lista (3p)



Skriv pseudokod för funktionen *List-Swap* i fältet nedan. Positionstypen *Pos* är en referens/pekare till ett element. Om *e* är av typen *Pos* så refererar *e.next* till elementet som följer efter *e* i listan. På samma sätt refererar *e.prev* till föregående element. Om du vill får din funktion *List-Swap()* använda ytterligare en referens/pekare som lokal variabel.

Svaret ska vara max 10 rader, exklusive algoritmhuvud, där varje pseudokodsrad utom den sista ska bestå av en tilldelning. Den sista kodraden ska innehålla en **return**-sats liknande den i *Array-Swap*. Även om svaret endast ska bestå av pseudokod så rekommenderar vi starkt att du ritar figurer liknande den ovan som stöd för dig själv.

Det som kommer att bedömas är

- att du löser uppgiften enligt förutsättningarna ovan utan att tappa bort några länkar och
- att du använder syntaxen som beskrivs ovan.

Du behöver inte kommentera koden.

Räkna med att funktionen har huvudet:

**Algorithm** List-Swap(a: List, j: Pos)

// Swap elements at positions *j* and its successor *k*. Return the

// modified list and the lowest position of *j* and *k* after the swap.

1	
---	--



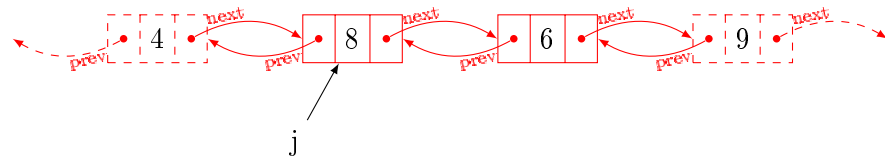
### 12.1 Lösningsförslag 1

```
1) k <- j.next
2) j.prev.next <- k
3) k.prev <- j.prev
4) k.next.prev <- j
5) j.next <- k.next
6) k.next <- j
7) j.prev <- k
8) return (a,k)
```

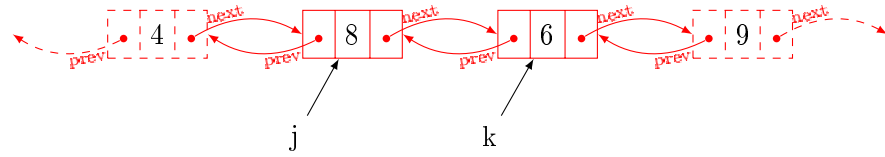
### 12.2 Lösningsförslag 2

```
1) j.prev.next <- j.next
2) j.next.prev <- j.prev
3) j.next.next.prev <- j
4) j.next <- j.next.next
5) j.prev.next.next <- j
6) j.prev <- j.prev.next
```

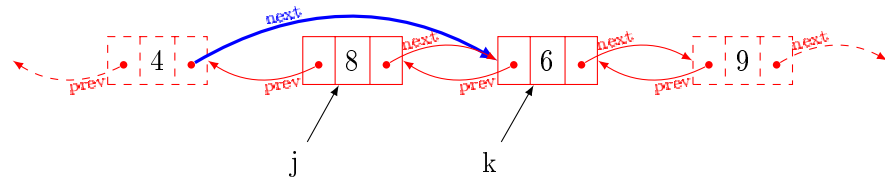
Lösningsförslag 1: Vid start



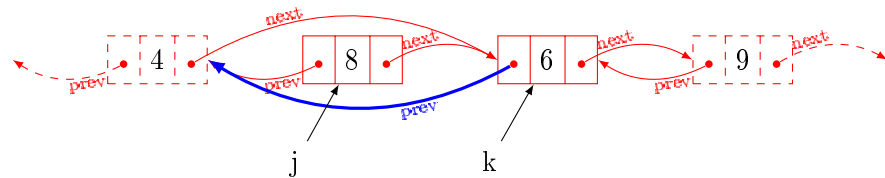
After rad 1  
`k <- j.next`



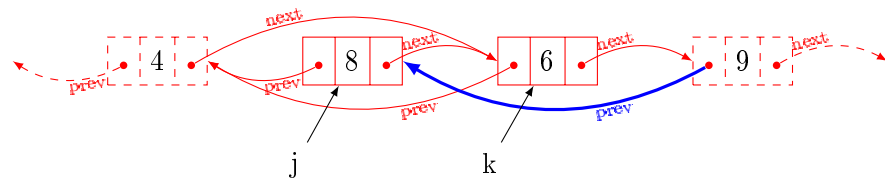
After rad 2  
`j.prev.next <- k`



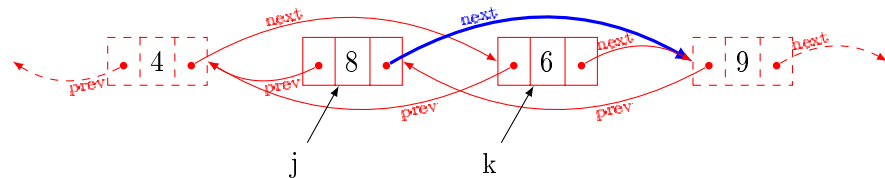
After rad 3  
`k.prev <- j.prev`



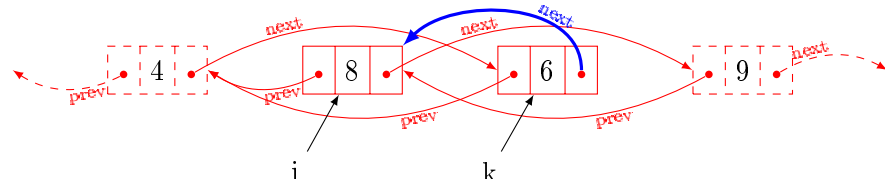
After rad 4  
`k.next.prev <- j`



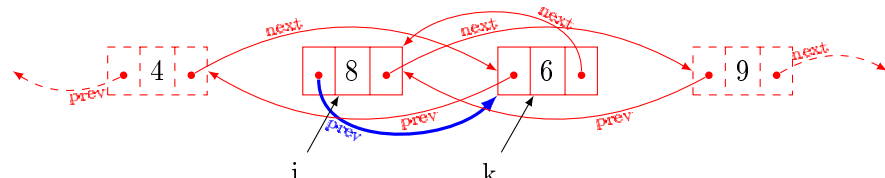
After rad 5  
`j.next <- k.next`



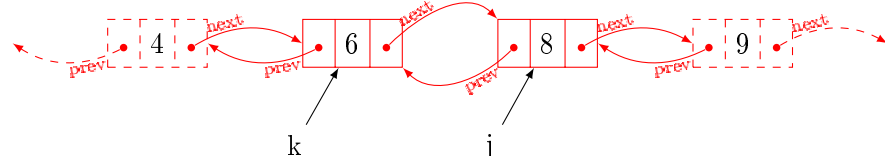
After rad 6  
`k.next <- j`



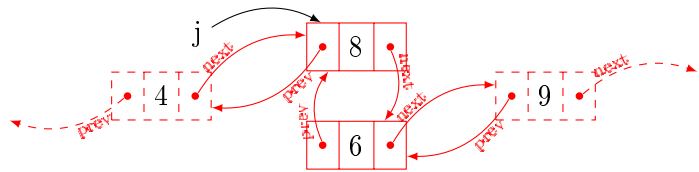
After rad 7  
`j.prev <- k`



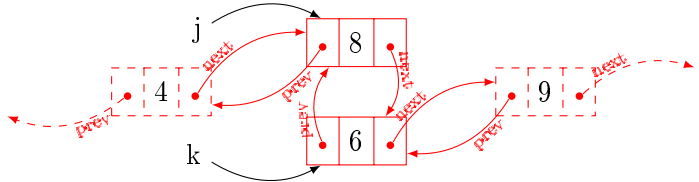
After rad 7, omritat



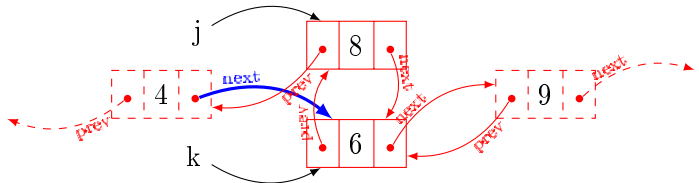
Lösningsförslag 1: Vid start  
Cellerna ritade annorlunda



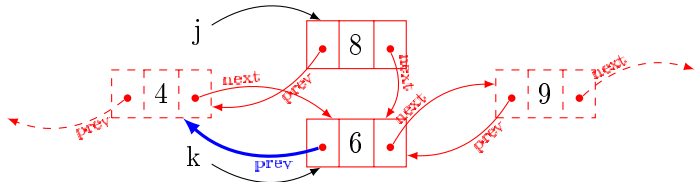
Efter rad 1  
`k <- j.next`



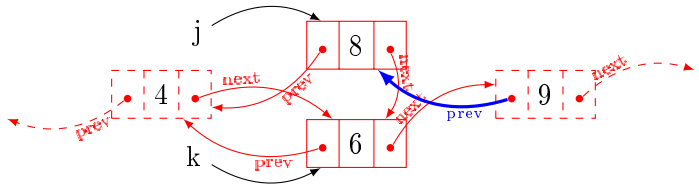
Efter rad 2  
`j.prev.next <- k`



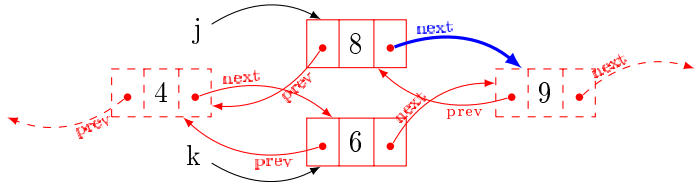
Efter rad 3  
`k.prev <- j.prev`



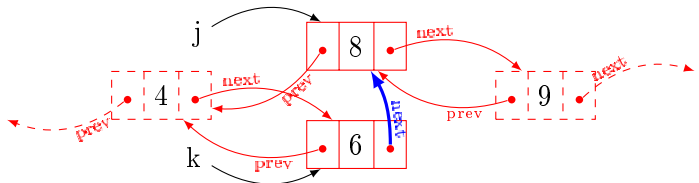
Efter rad 4  
`k.next.prev <- j`



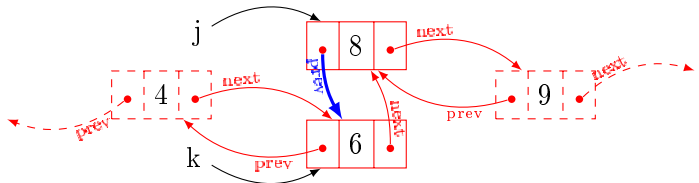
Efter rad 5  
`j.next <- k.next`



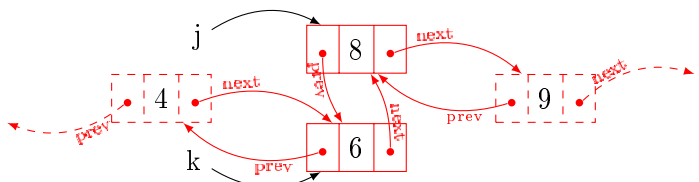
Efter rad 6  
`k.next <- j`



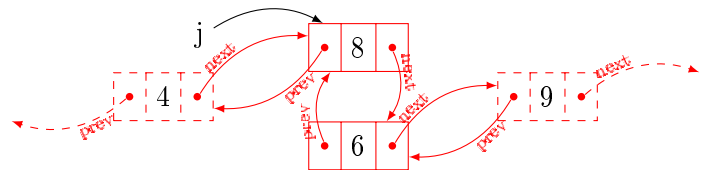
Efter rad 7  
`j.prev <- k`



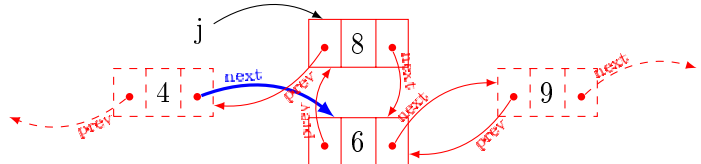
Klar



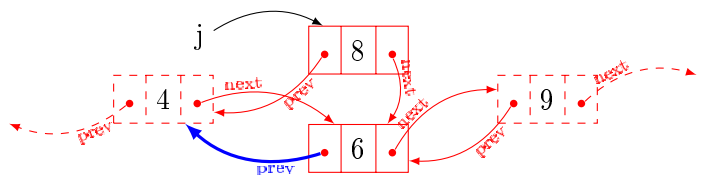
Lösningförslag 2: Vid start



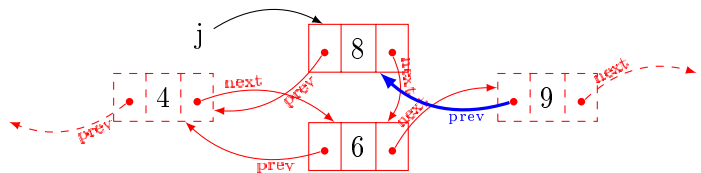
Efter rad 1  
`j.prev.next <- j.next`



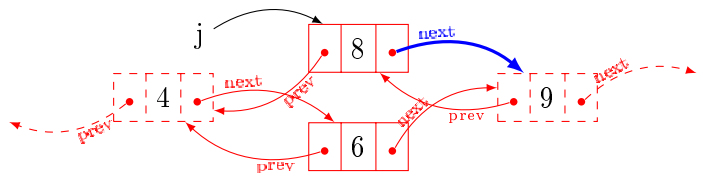
Efter rad 2  
`j.next.prev <- j.prev`



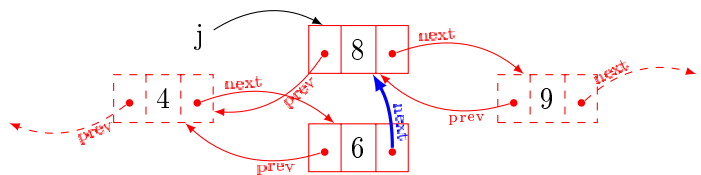
Efter rad 3  
`j.next.next.prev <- j`



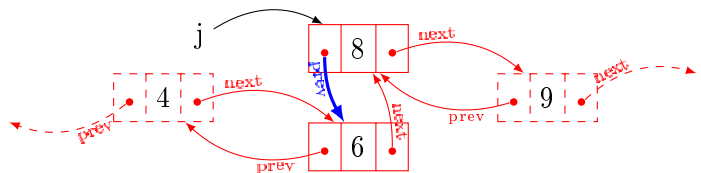
Efter rad 4  
`j.next <- j.next.next`



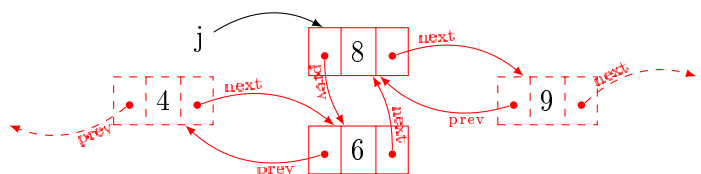
Efter rad 5  
`j.prev.next.next <- j`



Efter rad 6  
`j.prev <- j.prev.next`



Klar



Totalpoäng: 3

## 7.1 Insättning, borttagning

Vi har en hashtabell med 8 platser ( $m=8$ ) som ska lagra positiva tal och hashfunktionen  $h(x) = x \bmod 8$ . Vi använder sluten hashning och kvadratisk teknik för att undvika kollisioner. Tomma positioner i tabellen innehåller värdet -1. Vi har redan stoppat in talen 12, 17, 5 och 3 i tabellen, i den ordningen.

Om vi nu stoppar in talet 13 i tabellen, vilken plats hamnar värdet på?

Välj alternativ (Position 0, Position 1, Position 2, Position 3, Position 4, Position 5, **Position 6**, Position 7)

Om vi sedan stoppar in talet 19 i tabellen, vilken plats hamnar värdet på?

Välj alternativ (Position 0, Position 1, Position 2, Position 3, Position 4, Position 5, Position 6, **Position 7**)

Om vi sedan stoppar in värdet 9 i tabellen, vilken plats hamnar värdet på?

Välj alternativ (Position 0, Position 1, **Position 2**, Position 3, Position 4, Position 5, Position 6, Position 7)

Om vi till sist tar bort värdet 12 ur tabellen, vilket värde kommer att finnas i position 4 efter borttagningen?

Välj alternativ (-1, 12, 17, 5, 3, 13, 19, 9, **Något annat värde**)

Totalpoäng: 4

## 7.2 Storlek

Antag att vi vill använda en hashtabell för att lagra c:a 15 element. Vi tänker använda sluten hashing med kvadratisk teknik för kollisionshantering.

Vilket av följande värden är det bästa alternativet på storleken  $m$  för vår hashtabell?

☐ 31 element



☐ 32 element

☐ 33 element

☐ 34 element

---

Totalpoäng: 2