Information

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
i	Information		Information eller resurser

Grundbegrepp

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
1.1	Positioner och Index	6	Matchning
1.2	Egenskaper hos datatyper	5	Sammansatt

Träd

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
2.1	Traversering av binärt träd (3+3+3+3 poäng)	12	Sammansatt
2.2	Identifiera trädalgoritmer (2+2+2+2 poäng)	8	Sammansatt

Grafer, grafalgoritmer

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
3.1	Dijkstras algoritm (12 poäng)	12	Textfält

Sortering

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
4.1	Sortering efter kombinationer av nycklar, sekvens (4 poäng)	4	Sammansatt
4.2	Sortering efter kombinationer av nycklar, egenskaper (3 poäng)	3	Sammansatt

Komplexitet

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
5.1	Tillväxthastigheter (9 poäng)	9	Dra och släpp i text
5.2	Ordo c, n0 (4 poäng)	4	Textfält
5.3	Ordo g(n) (2 poäng)	2	Flervalsfråga

Listor, pseudokod

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
6.1	Pseudokod (5+4+6 poäng)	15	Programmering
6.2	Komplexitet (2+2+2 poäng)	6	Flervalsfråga

LZ78

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
7.1	Hjälpdatatyper (2+2 poäng)	4	Textfält
7.2	LZ78 avkodning	10	Textfält

Förtydliganden

Uppgift	Uppgiftstitel	Totalpoäng	Uppgiftstyp
8.1	Förtydliganden	0	Essä

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!
- Du har tillgång till en miniräknarfunktion i Inspera om du behöver.
- Niclas kommer att besöka tentamenslokalen i början av skrivtiden, troligen runt 09.30. Ett tips är att kolla igenom samtliga frågor för att se om du hittar några oklarheter att fråga honom om.
- I övrigt, ser du en oklarhet i någon fråga, *beskriv oklarheten och hur du har tolkat den* på sista "frågan" i denna tentamen. Där finns också plats att skriva kommentarer om du vill förtydliga något kring dina svar på någon fråga som inte har textsvar.
- Endast de som är inte är godkända 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.
- De flesta frågorna ger noll poäng vid fel svar. En del flersvarsfrågor ger avdrag vid fel svar för att inte uppmuntra till gissningar. Det står i så fall angivet i frågan. Avdraget påverkar då bara den aktuella frågan eller grupp av frågor (sida i Inspera).

Betyg:
Maximal poäng är 100.

- För betyg **3** (godkänt) krävs normalt **50** poäng.
- För betyg **4** krävs normalt **65** poäng.
- För betyg **5** krävs normalt **80** poäng.

Lycka till!

/Niclas

1.1 Positioner och Index

Ange för vart och ett av påståendena nedan om det är **Sant** eller **Falskt**. Denna fråga ger avdrag för fel svar. Om du vill nollställa ett svar kan du behöva ladda om sidan.

	Sant	Falskt
a) En Position kan användas tillsammans med flera listor.	○	○ ✔
b) Ett Index kan användas tillsammans med flera fält.	○ ✔	○
c) En Position i en lista blir odefinierad när listans struktur förändras.	○ ✔	○
d) Ett Index i ett Fält blir odefinierat när fältets struktur förändras.	○	○ ✔
e) En Lista kan innehålla flera positioner än element.	○ ✔	○
f) En Lista kan innehålla flera elementvärden än element.	○	○ ✔

Totalpoäng: 6

1.2 Egenskaper hos datatyper

Denna fråga ger avdrag för fel svar. Om du vill nollställa ett svar kan du behöva ladda om sidan.

a) Vilken/vilka av datatyperna nedan är **enkla**?
Välj ett eller flera alternativ

- [ ] Heltal ✔
- [ ] Fält
- [ ] Post (Record)
- [ ] Lista
- [ ] Tabell

b) Vilken/vilka av datatyperna nedan är **sammansatta**?

Välj ett eller flera alternativ

- [ ] Heltal
- [ ] Fält ✔
- [ ] Post (Record) ✔
- [ ] Lista ✔
- [ ] Tabell ✔

c) Vilken/vilka av datatyperna nedan har är **homogena**?
Välj ett eller flera alternativ

- [ ] Heltal
- [ ] Fält ✔
- [ ] Post (Record)
- [ ] Lista ✔
- [ ] Tabell ✔

d) Vilken/vilka av datatyperna nedan är **ordnade**?

Välj ett eller flera alternativ

- [ ] Heltal
- [ ] Fält ✔
- [ ] Post (Record)
- [ ] Lista ✔
- [ ] Tabell

Totalpoäng: 5

2.1 Traversering av binärt träd (3+3+3+3 poäng)

Illustrationen till vänster visar ett binärt träd. Nedanstående frågor handlar om traversering av trädet. I svaren, skriv **nodernas etiketter, separerade med kommatecken**, t.ex: **A,B,C,D,E,F,G**

a) I vilken ordning kommer noderna att besökas om trädet traverseras enligt **bredden-först**?

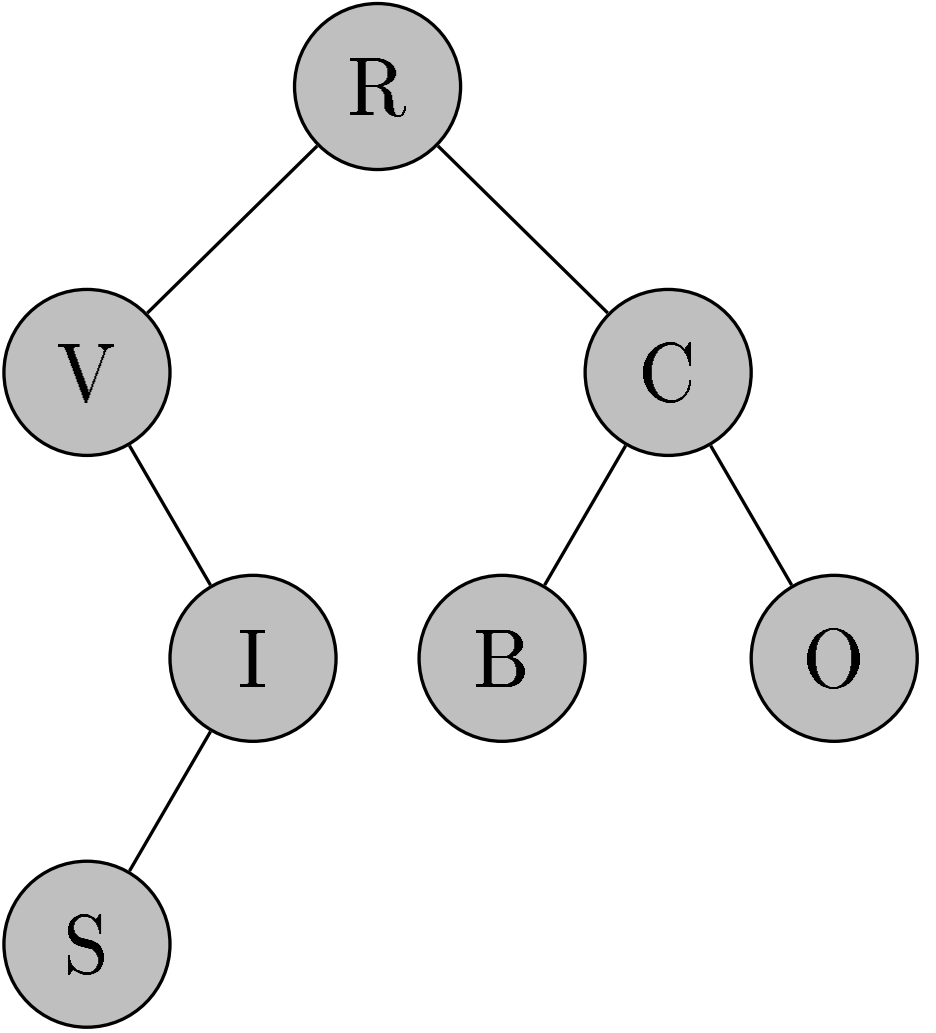
(R,V,C,I,B,O,S, RVCIBOS)

b) I vilken ordning kommer noderna att besökas om trädet traverseras enligt **djupet-först, preorder**? (R,V,I,S,C,B,O, RVISCBO)

c) I vilken ordning kommer noderna att besökas om trädet traverseras enligt **djupet-först, inorder**? (V,S,I,R,B,C,O, VSIRBCO)

d) I vilken ordning kommer noderna att besökas om trädet traverseras enligt **djupet-först, postorder**? (S,I,V,B,O,C,R, SIVBOCR)

Totalpoäng: 12


R
V
C
I
B
O
S

2.2 Identifiera trädalgoritmer (2+2+2+2 poäng)

Betrakta algoritmerna **foo** och **bar** i panelen till vänster. I frågorna nedan är **T** ett godtyckligt binärt träd och **T1** är det binära träd som återfinns längst ner i panelen (och är samma träd som i förra uppgiften).

e) Beskriv i text vilken trädegenskap anropet **foo(Root(T), T)** beräknar:

(höjden, trädets höjd)

f) Vad skulle anropet **foo(Root(T1), T1)** returnera? (3)

g) Beskriv i text var **bar(Root(T), T)** skriver ut, och i vilken ordning:

(lövens etiketter från vänster till höger, etiketterna på lövnoderna från vänster till höger)

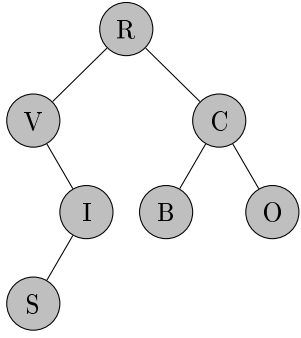
h) Vad skulle anropet **bar(Root(T1), T1)** generera för utskrift?

(SBO, S,B,O)

Totalpoäng: 8

```
Algorithm foo(n: Node, T: BinTree)

if Has-left-child(n, T) then
   p ← 1 + foo(Left-child(n, T))
else
   p ← 0

if Has-right-child(n, T) then
   q ← 1 + foo(Right-child(n, T))
else
   q ← 0

return max(p, q)
```

```
Algorithm bar(n: Node, T: BinTree)

if Has-left-child(n, T) then
   bar(Left-child(n, T))

if Has-right-child(n, T) then
   bar(Right-child(n, T))

if not (Has-left-child(n, T) or Has-right-child(n, T)) then
   print(Label(n, T))
```



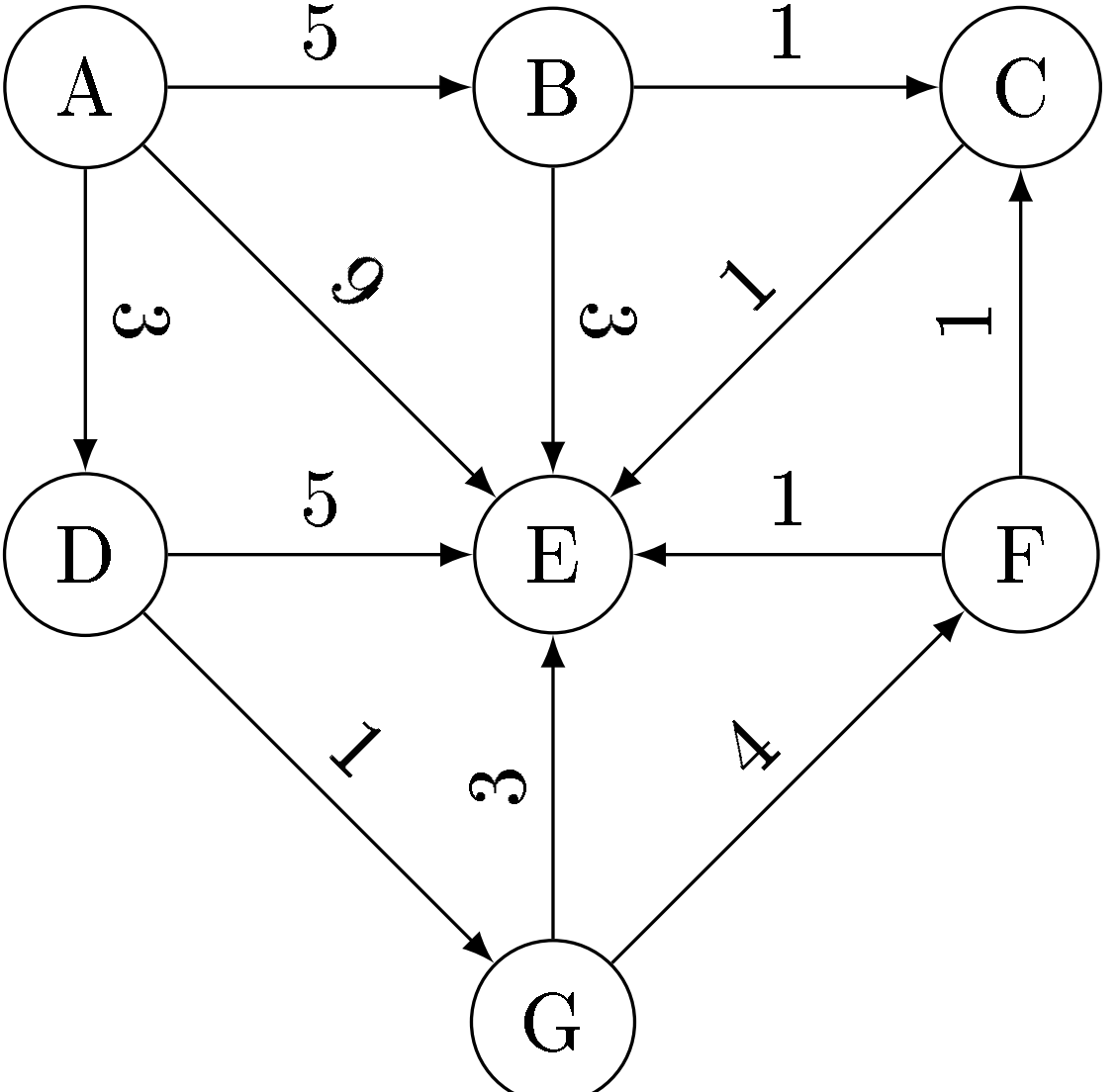
3 Dijkstras algoritm (12 poäng)

Applicera Dijkstras algoritm på grafen till vänster. Starta i nod **A**. När algoritmen ska iterera över ett antal bågar, iterera över bågarna i stigande viktordning.

I tabellen nedan, ange i vilken ordning noder får ett **förändrat** avstånd. Tänk på att en nod kan få förändrat avstånd flera gånger. Om ingen nod förändras, ange ett streck (minustecken -) på både nodnamnet och avståndet.

Steg	Nod	Avstånd
0	**A**	0
1	**D**	3
2	**B**	5
3	**E**	9
4	(G)	(4)
5	(E)	(8)
6	(E)	(7)
7	(F)	(8)
8	(C)	(6)
9	(-)	(-)

Totalpoäng: 12


A
B
C
D
E
F
G
5
1
3
6
3
1
1
5
1
1
3
4

4.1 Sortering efter kombinationer av nycklar, sekvens (4 poäng)

Vid sortering är det vanligt att värdena som sorteras består av Poster (*Records*), där något eller några fält används som nycklar vid sorteringen. En vanlig önskan är att posterna i första hand ska sorteras efter en *primär* nyckel, t.ex. efternamn, i andra hand efter en *sekundär* nyckel, t.ex. förnamn.

Den effektivaste lösningen är att skriva en egen jämförelsefunktion som kombinerar nycklarna för att avgöra vilken Post som ska komma först. Under vissa förutsättningar (se nästa fråga) kan vi dock uppnå samma resultat genom upprepad användning av en sorteringsalgoritm som endast kan sortera efter ett fält i posten.

Antag att anropet **Sort(l, 'firstname')** tar listan **l** av Poster och returnerar en kopia av listan sorterad efter förnamn. Antag på samma sätt att anropet **Sort(l, 'lastname')** returnerar en kopia av **l** sorterad efter efternamn.

a) Låt l innehålla den osorterade listan. Någon eller några av följande anrop/sekvenser av anrop ger oss en lista s sorterad i första hand efter efternamn och i andra hand efter förnamn. Ett eller flera svar är korrekt. Vilken/vilka? Denna fråga ger avdrag för fel svar.

- [ ] t <- Sort(l, 'firstname')
s <- Sort(t, 'lastname') ✔
- [ ] s <- Sort(Sort(l, 'firstname'), 'lastname') ✔
- [ ] s <- Sort(Sort(l, 'lastname'), 'firstname')
- [ ] t <- Sort(l, 'lastname')
s <- Sort(t, 'firstname')

Totalpoäng: 4

4.2 Sortering efter kombinationer av nycklar, egenskaper (3 poäng)

b) För att lösningen i förra uppgiften ska ge önskat resultat krävs att vår sorteringsalgoritm har en viss egenskap. Vilken?

- Sorteringsalgoritmen har komplexitet högst $O(n \log n)$.
- Sorteringsalgoritmen är *in-place*.
- Sorteringsalgoritmen är rekursiv.
- Sorteringsalgoritmen använder ett pivåelement.
- Sorteringsalgoritmen är stabil. ✔

Totalpoäng: 3

5.1 Tillväxthastigheter (9 poäng)

Studerna följande tillväxthastigheter:

- O(2^n) = $O(2^n)$
- O(n^2) = $O(n^2)$
- O(n) = $O(n)$
- O(log n) = $O(\log n)$
- O(1) = $O(1)$
- O(n log n) = $O(n \log n)$

Ordna tillväxthastigheterna i ordning efter tillväxt för stora n. Notera att en långsam tillväxt ger en snabb algoritm och vice versa.

Hjälp

Växer långsammast (snabbast algoritm): O(1) ✔ .

Växer näst långsammast: O(log n) ✔ .

Växer näst näst långsammast: O(n) ✔ .

Växer näst näst snabbast: O(n log ✔ .

Växer näst snabbast: O(n^2) ✔ .

Växer snabbast (långsammast algoritm): O(2^n) ✔ .

O(n)

O(n^2)

O(1)

O(log n)

O(2^n)

O(n log n)

Totalpoäng: 9

5.2 Ordo c, n0 (4 poäng)

Studera funktionen $T(n) = 3n^2 - 3n + 10$. Givet $T(n)$, vad blir c och lägsta n_0 enligt ordo-definitionen? Om något värde behöver avrundas, avrunda uppåt till närmast högre heltal.

Skriv in värdena på c respektive n_0 separerade med ett kommatecken: (3,4, 4,2, 4,3) .

Totalpoäng: 4

5.3 Ordo g(n) (2 poäng)

Studera funktionen $T(n) = 3n\log(n^2)$. Vilket av följande alternativ är det bästa $g(n)$ som uppfyller Ordo-definitionen?

Välj ett alternativ:

- $g(n) = \log(n)$
- $g(n) = n\log(n)$ ✔
- $g(n) = n^2\log(n)$
- $g(n) = \log(n^2)$
- $g(n) = n^2\log(n^2)$

Totalpoäng: 2

6.1 Pseudokod (5+4+6 poäng)

a)

Skriv en algoritm (funktion) **Multiplicity(l: DList, v: Value)** som tar en riktad (och osorterad) lista **l** och ett värde **v** och returnerar *multipliciteten* för **v**, dvs. antalet gånger som **v** förekommer som värde i **l**.

Algoritmen ska ha följande huvud:

Algorithm Multiplicity(v: Value, l: DList)

b)

Skriv en algoritm **Max-multiplicity(l: DList)** som använder **Multiplicity()** och returnerar den högsta multipliciteten i listan **l**.

Algoritmen ska ha följande huvud:

Algorithm Max-multiplicity(l: DList)

c)

Antag att du har en riktad lista **l** som kan innehålla upprepade värden. Skriv en funktion **Most-common(l: DList)** som returnerar en ny lista som innehåller alla elementvärden som har högst multiplicitet i **l**. (Kallas för *typvärde* (engelska: *mode*) på statistikspråk.

Algoritmen ska ha följande huvud:

Algorithm Most-common(l: DList)

Du kan välja två svårighetsgrader på denna uppgift.

- För 4 poäng får din lösning returnera en lista med *godtyckligt antal upprepningar* av elementvärdena. **Most-common** på listan **[3, 5, 9, 3, 7, 9]** skulle då kunna returnera t.ex. **[3, 9, 3, 9]** eller **[9, 3, 9]**.

- För full poäng (6p) får utlistan inte innehålla upprepade värden, dvs. **Most-common** skulle returnera **[3, 9]** eller **[9, 3]** på listan ovan.

I samtliga fall spelar ordningen på värdena ingen roll.

I din kod får du anta att likhetsoperatorn är definierad för enkla datatyper, t.ex. Heltal, men inte för datatypen Value. Däremot finns funktionen **Value-isequal(a, b: Value)** som returnerar True om **a** och **b** anses lika.

Var noggrann med att visa hur du tar hand om returvärden från funktionerna. Förutom de definierade funktionerna ovan får du bara använda funktioner i gränsytorna till **DList**.

Det som kommer att bedömas är

- att pseudokoden löser uppgiften under de givna förutsättningarna,
- att pseudokoden är fri från språkspecifika konstruktioner (t.ex. inga i++ eller måsvingar),
- att koden är korrekt indenterad,
- att koden är rimligt kommenterad, och
- om koden har optimal komplexitet ($g(n)$ är viktigast).

Skriv in pseudokoden för alla funktioner här

```
1
```

Totalpoäng: 15

```
abstract datatype DList(val)
auxiliary pos
  Empty() → DList(val)
  Isempty(l: DList(val)) → Bool
  First(l: DList(val)) → pos
  Next(p: pos, l: DList(val)) → pos
  Isend(p: pos, l: DList(val)) → Bool
  Inspect(p: pos, l: DList(val)) → val
  Insert(v: val, p: pos, l: DList(val))
                                → (DList(val), pos)
  Remove(p: pos, l: DList(val)) → (DList(val), pos)
```

6.1 Lösningsförslag

Notera att:

- `Multiplicity()` ska returnera 0 om värdet inte finns i listan.
- `Max-multiplicity()` för en tom lista ska returnera 0.
- `Most-common ()` för en tom lista ska returnera en tom lista.

```
Algorithm Multiplicity(v: Value, l: DList)
// Count the number of occurences of the value v in the list l

// Initialize counter
m ← 0

// Iterate over the input list...
p ← First(l)
while not Isend(p, l) do
    if Value-isequal(Inspect-value(p, l), v) then
        // Increment the counter for each matching value
        m ← m + 1
    // Advance in the list
    p ← Next(p, l)

// Return the count
return m
```

```
Algorithm Max-multiplicity(l: DList)
// Compute the largest multiplicity of any value in the list l

// Highest multiplicity seen so far
mx ← 0

// Iterate over the input list...
p ← First(l)
while not Isend(p, l) do
    // Compute multiplicity for this value
    mult ← Multiplicity(l, Inspect-value(p, l))
    // If this multiplicity is higher than any seen so far...
    if mult > mx then
        // ...remember this value
        mx ← mult

    // Advance in the input list
    p ← Next(p, l)

// Return the highest seen multiplicity
return mx
```

```
Algorithm Most-common-with-duplicates(l: DList)
// Returns a list with the most common values in the list l, i.e.,
// those values that have the largest multiplicity. The values will
// have the same multiplicity in the output list as in the input list.

// First compute the largest multiplicity in the list
m ← Max-multiplicity(l)

// Output list with values that have max multiplicity
t ← Empty()

// Iterate over the list...
p ← First(l)
while not Isend(p, l) do
    v ← Inspect-value(p, l)
    // If the multiplicity of this value equal the maximum...
    if Multiplicity(v, l) = m then
        // ...insert value into the output list. Since the order does not matter,
        // insert it first (simple).
        t ← Insert(v, First(t), t)

    // Advance in the input list
    p ← Next(p, l)

// Return the output list
return t
```

```
Algorithm Most-common-no-duplicates(l: DList)
// Returns a list with the most common values in the list l, i.e.,
// those values that have the largest multiplicity. The output list
// contains each value exactly once.

// First compute the largest multiplicity in the list
m ← Max-multiplicity(l)

// Output list with values that have max multiplicity
t ← Empty()

// Iterate over the list...
p ← First(l)
while not Isend(p, l) do
    v ← Inspect-value(p, l)
    // If the multiplicity of this value equal the maximum...
    if Multiplicity(v, l) = m then
        // ...and the values does not appear in the output list
        if Multiplicity(v, t) > 0 then
            // ...insert value into the output list. Since the order does not matter,
            // insert it first (simple).
            t ← Insert(v, First(t), t)

    // Advance in the input list
    p ← Next(p, l)

// Return the output list
return t
```

6.2 Komplexitet (2+2+2 poäng)

Låt n vara antalet element i den riktade listan.
Vad är komplexiteten för **Multiplicity()**?

- $O(1)$
- $O(n)$ ✔
- $O(n^2)$
- $O(n^3)$

Vad är komplexiteten för **Max-multiplicity()**?

- $O(1)$
- $O(n)$
- $O(n^2)$ ✔
- $O(n^3)$

Vad är komplexiteten för **Most-common()**?

- $O(1)$
- $O(n)$
- $O(n^2)$ ✔
- $O(n^3)$

Totalpoäng: 6

7.1 Hjälpdatatyper (2+2 poäng)

Kompressionsalgoritmerna Huffman och LZ78 använder olika datatyper vid kodning (kompression) och dekodning (dekompression).

a) Vilken datatyp använder LZ78-algoritmen för *kodning*? _____ (trie, traj, ett trie, ett traj)

b) Vilken datatyp använder LZ78-algoritmen för *dekodning*? _____ (tabell, table, en tabell)

Totalpoäng: 4

7.2 LZ78 avkodning

Du har fått ett hemligt meddelande kodat med LZ78 algoritmen som lyder:
(0, B), **(0, A)**, **(0, N)**, **(2, N)**, **(0, K)**, **(2, K)**, **(4, S)**.

Vilket är det avkodade meddelandet? (BANANKAKANS)

Totalpoäng: 10

8 Förtydliganden

Om du anser att du behöver förtydliga ditt svar på någon/några frågor som inte har textsvar kan du skriva det här. Annars lämnar du fältet tomt.

Teckenf...

Ord: 0

Totalpoäng: 0