Opgave 1 (20%)

Et TRINE-udtryk, der angiver en liste af heltal, kan involvere konstanter, ++-operatoren, udtagelse af dellister og kopiering af elementer. Sådanne udtryk kan beskrives med den rekursive type:

```
Type IntList = Sum(const: List(Int),
                   concat: Prod(left, right: IntList),
                   sub: Prod(argument: IntList, i, j: Int),
                   xerox: Prod(element: Int, n: Int)
                  )
```

Fx vil TRINE-udtrykket **List**(1,2,3) blive repræsenteret som:

IntList(const: **List**(1,2,3)),

List(1,2,3)++**List**(4,5,6) som:

IntList(concat: Prod(IntList(const:**List**(1,2,3)),IntList(const: **List**(4,5,6)))),

List(1,2,3)(0..2) som:

IntList(sub(IntList(const: **List**(1,2,3)),0,2))

og **List**(5 | 17) som:

IntList(xerox: **Prod**(5,17)).

> Skriv en værdiprocedure:
>
> **Proc** Eval[x: IntList] → (**List**(Int))
>
> der beregner værdien af den angivne liste. Der lægges vægt på, at besvarelsen er letlæselig, detaljeret og korrekt.

Opgave 2 (20%)

Betragt følgende algoritme, der finder de to største tal i en liste:

Algoritme: Multiplikation
Stimulans: x: **List**(Int), $|x| \geq 2$
Respons: m,n: $(x.(m) \geq x(0..|x|)) \wedge (x.(n) \geq x(0..m){+}{+}x(m{+}1..|x|))$
Metode:

```
if x.(0) ≥ x.(1) → m, n := 0, 1
 | x.(1) ≥ x.(0) → m, n := 1, 0
fi
i := 2
do { I }
    i < |x| →
      if x.(i) ≥ x.(m) → m, n := i,m
      & x.(i) ≥ x.(n) → n := i
      fi
      i := i+1
od
```

Betragt følgende samling af foreslåede invarianter:

I_1: $x.(m) \geq x.(n)$

I_2: $(0 \leq m,n < i \leq |x|) \wedge (x.(m) \geq x(0..i)) \wedge (x.(n) \geq x(0..m){+}{+}x(m{+}1..i))$

I_3: $(x.(m) \geq x(0..|x|)) \wedge (x.(n) \geq x(0..m){+}{+}x(m{+}1..|x|))$

I_4: false

a) Hvilke af invarianterne er gyldige? Angiv for hver et bevis eller et modeksempel.

b) Hvilke af de gyldige invarianter medfører sammen med terminering korrekthed? Begrund dine svar.

c) Bevis terminering og dermed korrekthed af algoritmen.

Opgave 3 (20%)

Der skal konstrueres en box SparseMatrix med følgende udseende

> **Box** SparseMatrix
>   **Type** M = ≪uendelig matrix af heltal≫
>   **Proc** Init [x: M]
>   **Proc** Lookup [x: M] (i, j: Int) → (Int)
>   **Proc** Update [x: M] (i, j, v: Int)
>   **Proc** Transpose [x: M]
>   **Proc** Add [x: M] → (Int)
> **end** SparseMatrix

som realiserer en datastruktur, hvis værdier er uendelige matricer af heltal, dvs. strukturer der til hvert par af ikke-negative heltal (i, j) knytter et heltal.

Proceduren Init giver den uendelige matrix med nuller på alle pladser. Proceduren Lookup returnerer værdien med index (i, j). Proceduren Update ændrer værdien med index (i, j) til at være v. Proceduren Transpose transponerer matricen, dvs. spejler dens elementer i hoveddiagonalen. Proceduren Add giver summen af matricens elementer.

I det følgende angiver ||x|| antallet af elementer i x, der har værdi forskellig fra nul.

a) Giv en formel specifikation af proceduren Transpose.

b) Beskriv en realisation af typen M, så Init får tidskompleksitet $O(1)$, Lookup og Update får tidskompleksitet $O(\log \|\|x\|\|)$, og Transpose og Add får tidskompleksitet $O(1)$.

Opgave 4 (20%)

I denne opgave indfører vi en ny binær operator i RASMUS. Syntaksen er:

```
R [a] S
```

Det kræves, at a er navnet på en attribut, der er fælles for relationerne R og S, samt at de ikke har andre fælles attributter. Resultatet af operationen er de tupler | r | s |, for hvilke der findes præcis en a-værdi | a | således at tuplet | r | a | er i R og tuplet | a | s | er i S. Lad i det følgende R og S være relationerne:

Sælger:Text	Vare:Text
Fido A/S	hund
Katte Discount	kat
Vov & Co.	hund
Turtles 'R Us	skildpadde
Furry Friends	hamster
Jysk Kanin Import	kanin
Fibonacci Int'l	kanin

og

Vare:Text	Køber:Text
hund	Anders Husfeldt
kat	Thore Hougaard
skildpadde	Jesper Arge
hamster	Lars Pedersen
hamster	Ole Sandholm
kanin	Jan Lauridsen
guldfisk	Ole Gulmann

a) Angiv for eksempelrelationerne resultatet af: `R [Vare] S`.

b) Angiv, hvilke af følgende regneregler, der er gyldige (under antagelse af at begge sider er lovlige relationsudtryk). Begrund dine svar.

1) `R [a] R = R`

2) `R [a] S = S [a] R`

3) `(R [a] S) [b] T = R [a] (S [b] T)`

4) `(R [a] (S + T)) = (R [a] S) + (R [a] T)`

5) `(R [a] (S * T)) = (R [a] S) * (R [a] T)`

c) Vis, hvorledes `R [a] S` generelt kan udtrykkes ved hjælp af de sædvanlige operatorer i RASMUS.

Opgave 5 (20%)

Følgende algoritme fra [Grafalgoritmer og Algoritmisk Problemløsningsteknik] beregner en topologisk sortering af en orienteret acyklisk graf.

Algoritme: Topologisk Sortering
Stimulans : $G = (V, E)$ orienteret acyklisk graf
Respons : TopSort: Vector, indeholder topologisk sortering af G
Metode :

```
Indegree:=Vector(0|n)
for (v,w) in E do
        Indegree.(w):=Indegree.(w)+1
od
NS'Init[R](n)
for v in V do
    if Indegree.(v) = 0 → NS'Insert[R](v) fi
od
TopSort,N:=Vector(0|n),1
do ¬ NS'Empty[R] →
    NS'DeleteSome[R,v]
    TopSort.(v),N:=N,N+1
    for (v,w) in E do
        Indegree.(w):=Indegree.(w)-1
        if Indegree.(w) = 0 → NS'Insert[R](w) fi
    od
od
```

a) Argumentér for, at der findes en længste vej i en orienteret acyklisk graf.

b) Modificér metoden, så den i stedet realiserer følgende algoritme.

Algoritme: Længste vej
Stimulans : $G = (V, E)$, orienteret acyklisk graf
Respons : L: Int, angiver længden af den længste vej i G
Metode :