# Programming Paradigms 2025
# Session 8: Catching up

## Problems for solving and discussing

Hans Hüttel

28 October 2025

### About this session

This time we will be re-visiting problems and topics from previous session that either caused headaches or were numbered problems that we never got to solve because we ran out of time.

The overall goal this time is to help you better master topics from the first half of the course, and in particular those of recursion and datatypes.

### Problems that we will definitely talk about

1. ***(10 minutes)***

   Find Haskell expressions that have the types

   1. (Ord a, Num a)=>a -> a -> [[Bool]] -> Bool
   2. Num a =>(t -> a, t)-> a -> a

2. ***(25 minutes)***

   A former minister of science and education is now trying to get a university degree and is busy learning Haskell. The minister is trying to construct a function triples that takes a list of tuples (each tuple has exactly 3 elements) and converts that list of tuples into a tuple of lists.

   triples [(1,2,3), (4, 5, 6), (7, 8, 9)] should produce ( [1,4,7], [2, 5, 8], [3, 6, 9] ).

   The minister wrote the following piece of code and a type specification but ran into problems.

```
triples :: Num a => [(a,a,a)] -> ([a],[a],[a])

triples [] = ()
triples [(a,b,c)]= ([a],[b],[c])
triples (x:xs,y:ys,z:zs) = [x,y,z] : Triples [(xs,ys,zs)]
```

   Fix the issues that the triples function has. Write a recursive definition of triples that actually works. *Please use recursion and local declarations in your solution and use the approach described in Section 6.6.*

3. ***(25 minutes)***

   A new computer game uses nested boxes. A box is one of the following two kinds:

   - A box can be red and contain a pair of boxes, or
   - A box can be blue and contain a value.

   All values found in a nested box must be values of the same type.

   See Figure 1 for an example.

   a) Define an algebraic datatype Box that describes the valid forms of nested boxes. Give a term of type Box that describes the box in Figure 1.

   b) A nested box is called *samey* if every time we see a pair of blue boxes somewhere inside it (immediately or more than one level down), the values in those two blue boxes are equal. The nested box in Figure 1 is *not* samey.

   Define a function samey that can tell us if a box is samey. What should the type of samey be?

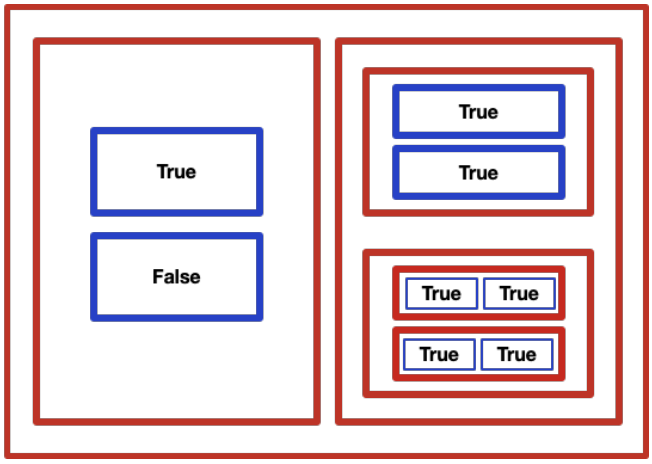


Figure 1: A box from the computer game that contains values of type Bool.

4. ***(20 minutes)***

   From linear algebra we know that a vector space with inner product is one for which the operations of vector sum and dot product are defined. Given two vectors $v_1$ and $v_2$, the sum $v_1 + v_2$ is again a vector, and the inner product $v_1 \cdot v_2$ is a number. *Please note:* In linear algebra there is much more to the definition of inner product spaces than these two requirements, but in this problem, please ignore that. Also assume that the inner product is a number of type Int.

   Define a typeclass InVector whose instances are types that have the required operations and where vector sum is called &&& and inner product is called ***. *Hint:* Which section in the textbook do you need here?

   Find out how to declare Bool as an instance of InVector. *Hint:* You have to find a definition of vector sum and inner product for truth values.

## More problems to solve at your own pace

a) Find a function or an expression that has the type

```
Fractional t1 => (t2 -> t1) -> (t2 -> t1) -> (t1 -> t3) -> t2 -> t3
```

b) The goal of this problem is to define a function frequencies that, given a string $s$, creates a list of pairs [(x1,f1) ,....( xk,fk)] such that if the character xi occurs a total number of fi times throughout the list $s$, then the list of pairs will contain the pair (xi, fi).

   As an example of this,

```
frequencies "regninger"
```

   should return the list

```
[('r',2),('e',2),('g',2),('n',2),('i',1)]
```

   a) What should the type of the function be?

   b) Use *recursion* to give a definition of frequencies.

c) A theorem in number theory states that every non-zero real number $x$ can be written as a *continued fraction*. This is a potentially infinite expression of the form

$$x = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{a_4 + \cfrac{1}{\cdots \cfrac{1}{a_n}}}}}} \tag{1}$$

For rational numbers, the $a_i$'s will eventually all be 0, so the continued fraction is finite; for irrational numbers, the continued fraction will be infinite. See e.g. [1] for more.

The goal of this problem is to write a Haskell function cfrac that will, given a real number $r$ and a natural number $n$, finds the list of the first $n$ numbers in the continued fraction expansion of $r$. What should the type of cfrac be?

## Bibliography

[1] Wikipedia. Continued fractions. `https://en.wikipedia.org/wiki/Continued_fraction`.