CPS506 Lab 5 – Functions and Type Signatures in Haskell

Preamble

Haskell is a statically-typed language, yet we are not required to explicitly assign type when naming variables and functions. Haskell instead uses an inference engine to decide, at compile time, what type (or type class) a value or function will be. We can ask Haskell to report this type in **GHCi** using **`:t`**.

It is common in Haskell to describe functions using this type signature, so it's important to understand them. In this lab, type signatures are provided for several functions that you will write. When your function is loaded and we ask **GHCi** for its type using **`:t`**, it should match the description.

Lab Description

Create a Haskell file called **`lab5.hs`**, and define a single module called **`Lab5`** that contains the following simple functions:

i) **`thirdLast :: [a] -> a`**

Returns the third last item in an input list. You may assume the list has at least three elements. You may NOT use the **!!** operator.

ii) **`everyOther :: [a] -> [a]`**

Returns a list containing every other element, starting with the first element. Your function should work regardless of the size of the list. It could have an even or odd length. You may assume the input list is not empty. You may NOT use the **!!** operator.

iii) **`sumPosList :: (Num p, Ord p) => [p] -> p`**

Returns the sum of all positive values in a list. The sum of the elements in the empty list is zero.

Submission

This is a practice exercise only. There is nothing to submit.