Functional Programming in Erlang

Now we'll deploy the template from the previous videos to make recursive definitions. The aim of these exercises is to familiarise you with defining functions over lists in Erlang, in particular the different way that functions can combine the elements of a list.

Combining list elements: the product of a list

Using the template from the last session, define an Erlang function to give the `product` of a list of numbers. The product of an empty list is usually taken to be `1`: why?

Combining list elements: the maximum of a list

Define an Erlang function to give the `maximum` of a list of numbers.

- You might find it helpful to use the function `max/2` that gives the maximum of two values.
- It's not obvious what should be the value for the maximum of an empty list of numbers. You could therefore choose to define maximum on lists with at least one element only: to do this you will need to change the base case of the template.

Direct and tail recursion

In each case you could give either a direct recursion or a tail recursive definition. Now give the other kind of definition for `product` and `maximum`.

Which of the two styles – direct recursion and tail recursion – do you find most natural? Why? You can use the comments on this step to discuss your views with other participants on the course.