1 Erlang Basics

1.1 Explain the Erlang Type System

Erlang is dynamically typed.

1.2 References cannot be defined by words starting with lowercase letters.

Because someone thought it would be cool to have bare words, but only for single words, starting with lowercase letters?

1.3 Difference between `list` and `tuple`?

The size of a tuple is fixed, while lists can grow.

1.4 What are `foreach`, `filter`, `map`, `fold`?

foreach – Applies the function `f` once to each element in list `l`.

filter – Returns an list of all values from list `l` for which the function `f` returned true.

map – Same as `foreach`, but it returns a list of the result of `f` for each element in `l`.

foldl – Iterates through the list `l`, calling function `f` once for each element. `f` is called with two arguments, the element, and the return value of the previous call to `f` or the start value `a`.

2 Research Questions

2.1 Erlang is known for its mantra "Let it crash." What does this mean and how is Erlang designed to prefer this method of maintenance?

Instead of checking for all possible error states the program should be architected to fail and crash when an invalid state is entered. Another monitoring process should then restart or fix the failed process. Erlang is designed for this mantra by focusing on multithreading, and client server designs.

2.2 List three examples of the types of systems that are built using Erlang. Do you believe that Erlang is a good choice for these systems?

– Realtime chat application.

– SMS systems.

– AWS database backend.

Facebook's and T-Mobile's applications are obviously well fitted, Erlang was designed for telecommunications. Amazons choice of Erlang for their database system makes sense as well, Erlang is designed for realtime long running processes.