Recursive Functions and Induction: Zip

Read the chapter about total recursive functions in the “Tutorial on Isabelle/HOL” (fun, Chapter 3.5).

In this exercise you will define a function Zip that merges two lists by interleaving. Examples: Zip \([a_1, a_2, a_3] \ [b_1, b_2, b_3]\) = \([a_1, b_1, a_2, b_2, a_3, b_3]\) and Zip \([a_1] \ [b_1, b_2, b_3]\) = \([a_1, b_1, b_2, b_3]\).

Use three different approaches to define Zip:

1. by primitive recursion on the first list,
2. by primitive recursion on the second list,
3. by total recursion (using fun).

consts \(\text{zip1} : \ ''\ a \ \text{list} \ \Rightarrow \ ''\ a \ \text{list} \ \Rightarrow \ ''\ a \ \text{list}''\)
consts \(\text{zip2} : \ ''\ a \ \text{list} \ \Rightarrow \ ''\ a \ \text{list} \ \Rightarrow \ ''\ a \ \text{list}''\)
consts \(\text{zipr} : \ ''\ a \ \text{list} \ \Rightarrow \ ''\ a \ \text{list} \ \Rightarrow \ ''\ a \ \text{list}''\)

Show that all three versions of Zip are equivalent.

Show that \(\text{zipr}\) distributes over append.

lemma "\([\text{length p} = \text{length u}; \ \text{length q} = \text{length v}] \ \Rightarrow \ \text{zipr} \ (\text{p} @ \text{q}) \ (\text{u} @ \text{v}) = \text{zipr} \ \text{p} @ \text{zipr} \ \text{q} \ \text{v}\)"

Note: For fun, the order of your equations is relevant. If equations overlap, they will be disambiguated before they are added to the logic. You can have a look at these equations using thm \(\text{zipr}.\text{simps}\).