DTU Compute

Lyngby 25-09-2024
Michael R. Hansen

Luggage and Flights
an exercise based on an old exam question

Flight travelers may check-in the pieces of luggage, that should follow them on their journey, also when it contains multiple stops. A piece of luggage is marked with an *identification* (type `Lid`) by the start of the journey and that identification is associated with the *route* (type `Route`) of the journey. A route is a list of pairs identifying the *flights* (type `Flight`) and *airports* (type `Airport`) the luggage is passing on the journey.

Furthermore, a *luggage catalogue* (type `LuggageCatalogue`) is maintained, that uniquely identifies the routes of all pieces of luggage leaving some airport.

This is captured by the type declarations:

```
type Lid     = string
type Flight  = string
type Airport = string

type Route            = (Flight * Airport) list
type LuggageCatalogue = (Lid * Route) list
```

An example of a luggage catalogue is

```
[("DL 016-914", [("DL 189","ATL"); ("DL 124","BRU"); ("SN 733","CPH")]);
 ("SK 222-142", [("SK 208","ATL"); ("DL 124","BRU"); ("SK 122","JFK")])]
```

where first element in the list describes that the piece of luggage with identification "DL 016-914" is following a route, where it is first flown to Atlanta ("ATL") with flight "DL 189", then flown to Bruxelles "BRU" with flight "DL 124", and so on.

1. Declare a function `findRoute: Lid*LuggageCatalogue -> Route`, that finds the route for a given luggage identification in a luggage catalogue. A suitable exception should be raise if a route is not found.
2. Declare a function `inRoute: Flight -> Route -> bool`, that decides whether a given flight occurs in a route.
3. Declare a function `withFlight` *f lc*, where *f* is a flight and *lc* is a luggage catalogue. The value of the expression `withFlight` *f lc* is a list of luggage identifiers for the pieces of luggage that should travel with *f* according to *lc*. The sequence in which the identifiers occur in the list is of no concern.

   For the above example, both "DL 016-914" and "SK 222-142" should travel with the flight "DL 124".

An *arrival catalogue* associates with every airport, identifications of all pieces of luggage that should arrive at the airport. This is captured by the type declaration:

```
type ArrivalCatalogue = (Airport * Lid list) list
```

The following arrival catalogue is derived from the luggage catalogue appearing on the previous page:

```
[("ATL", ["DL 016-914"; "SK 222-142"]);
("BRU", ["DL 016-914"; "SK 222-142"]);
("JFK", ["SK 222-142"]);
("CPH", ["DL 016-914"])]
```

4. Declare a function `extend: Lid*Route*ArrivalCatalogue -> ArrivalCalalogue` so that `extend`(lid, r, ac) is the arrival catalogue obtained by extending ac with the information that lid will arrive at each airport contained in route r.

5. Declare a function `toArrivalCatalogue: LuggageCatalogue -> ArrivalCatalogue`, that creates an arrival catalogue from the information of a given luggage catalogue.

After Lecture 4 you should try to solve Question 5 using `extend` from Question 4 in combination with either `List.fold` or `List.foldBack`. The types of these functions are:

- `List.fold: ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a`
- `List.foldBack: ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b`