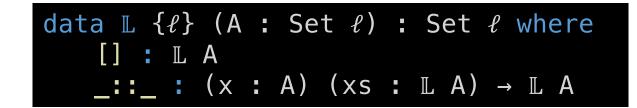
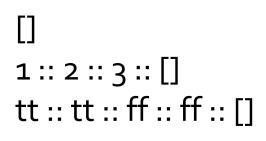
Chapter 19: Lists in Agda

Zhenjiang Hu, Wei Zhang School of Computer Science, PKU November 9, 2022



The List Datatype and Type Parameters







Basic Operations on Lists

```
[\_] : \forall \{\ell\} \{A : Set \ell\} \rightarrow A \rightarrow \mathbb{L} A
[ x ] = x :: []
is-empty : \forall \{\ell\} \{A : Set \ell\} \rightarrow \mathbb{L} A \rightarrow \mathbb{B}
is-empty [] = tt
is-empty (_ :: _) = ff
head : \forall \{\ell\} \{A : Set \ell\} \rightarrow (l : L A) \rightarrow is-empty l \equiv ff \rightarrow A
head [] ()
head (x :: xs) _ = x
head2 : \forall \{\ell\} \{A : Set \ell\} \rightarrow (l : L A) \rightarrow maybe A
head2 [] = nothing
head2 (a :: _) = just a
```



Basic Operations on Lists

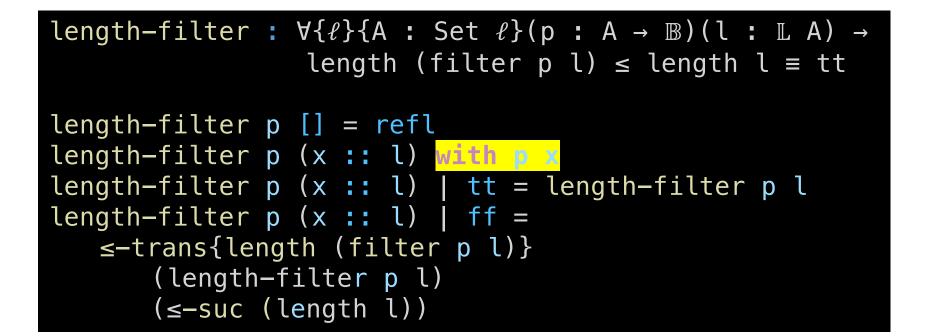
```
length : \forall \{\ell\} \{A : Set \ \ell\} \rightarrow \mathbb{L} A \rightarrow \mathbb{N}
length [] = 0
length (x :: xs) = suc (length xs)
\_++\_ : \forall \{\ell\} \{A : Set \ell\} \rightarrow \mathbb{L} A \rightarrow \mathbb{L} A \rightarrow \mathbb{L} A
[] ++ ys = ys
(x :: xs) ++ ys = x :: (xs ++ ys)
map : \forall \{\ell \ \ell'\} \{A : Set \ \ell\} \{B : Set \ \ell'\} \rightarrow (A \rightarrow B) \rightarrow \mathbb{L} A \rightarrow \mathbb{L} B
map f [] = []
map f(x :: xs) = f x :: map f xs
filter : \forall \{\ell\} \{A : Set \ell\} \rightarrow (A \rightarrow \mathbb{B}) \rightarrow \mathbb{L} A \rightarrow \mathbb{L} A
filter p [] = []
filter p(x :: xs) = let r = filter p xs in
                                     if p x then x :: r else r
foldr : \forall \{\ell \ \ell'\} \{A : Set \ \ell\} \{B : Set \ \ell'\} \rightarrow (A \rightarrow B \rightarrow B) \rightarrow B \rightarrow L A \rightarrow B
foldr f b [] = b
foldr f b (a :: as) = f a (foldr f b as)
```



Reasoning about List Operations



Length of Filtered Lists, and the with Construct





Filter Is Idempotent, and the keep Idiom





Homework

19.1. Define a polymorphic function takeWhile, which takes in a predicate on type A (i.e., a function of type A \rightarrow B), and a list of As, and returns the longest prefix of the list that satisfies the predicate.

19.2. Define a function repeat function that takes a number n and an element a, and constructs a list of length n where all elements are just a.

19.3. Prove that if value a satisfies predicate p, then takeWhile p (repeat n a) is equal to repeat n a, where takeWhile is the function you defined in the previous problem.

