Visualization of Data Structures in Haskell

It is often difficult to understand a data structure based solely on the Haskell code. This is especially true when applying algorithms to these data structures and trying to understand how they change as a result. A visualization can help provide a better understanding of the data structure.

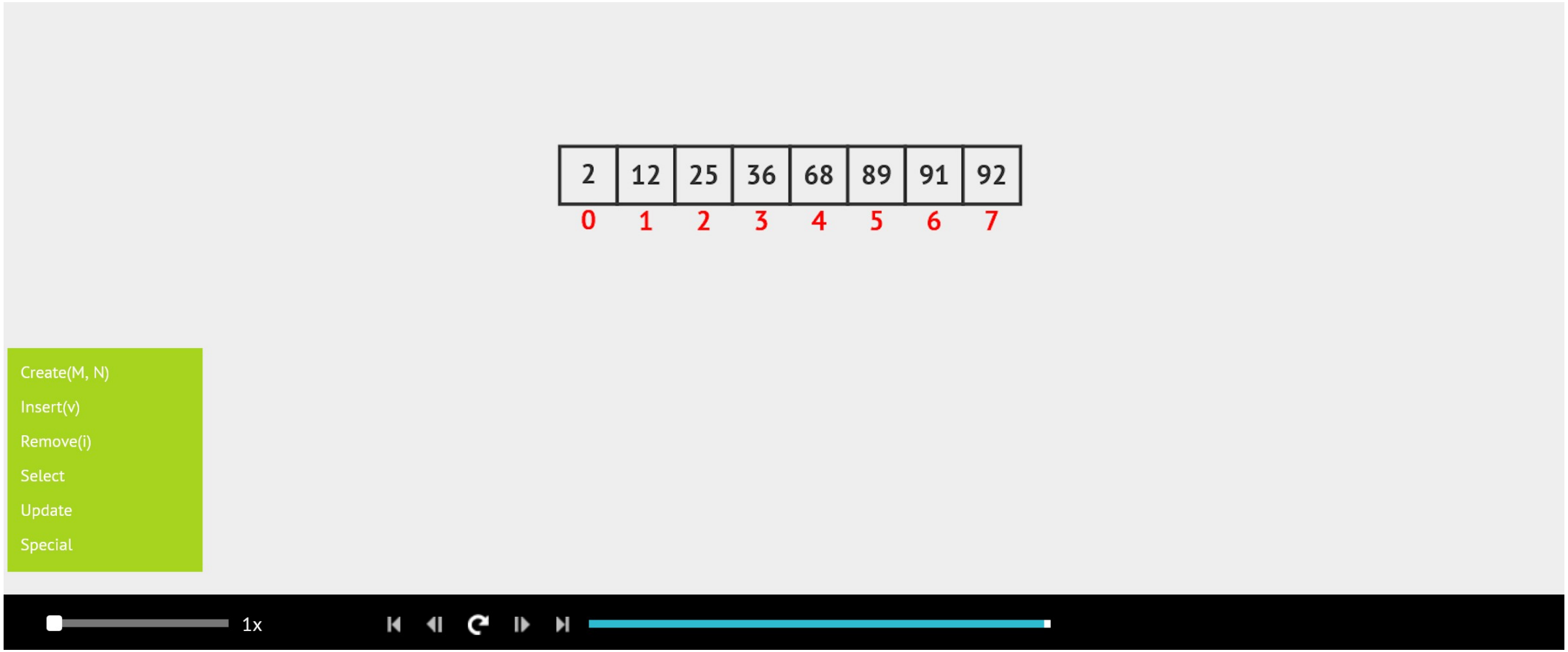
Visualization of Data Structures in Haskell
Your Task

- Visualization of lists and binary trees
- Visualization of algorithms applied to data structures (f.e. insertion, deletion, searching, sorting)
  - 1. Application of algorithms, i.e., visualization of the data structure before and after execution
  - 2. Step-by-step application of algorithms, i.e., visualization of intermediate steps
- Automated generation of sample data structures (with the option to export them)
- Visualization of arbitrary tree structures

Visualization of Data Structures in Haskell

Example: List

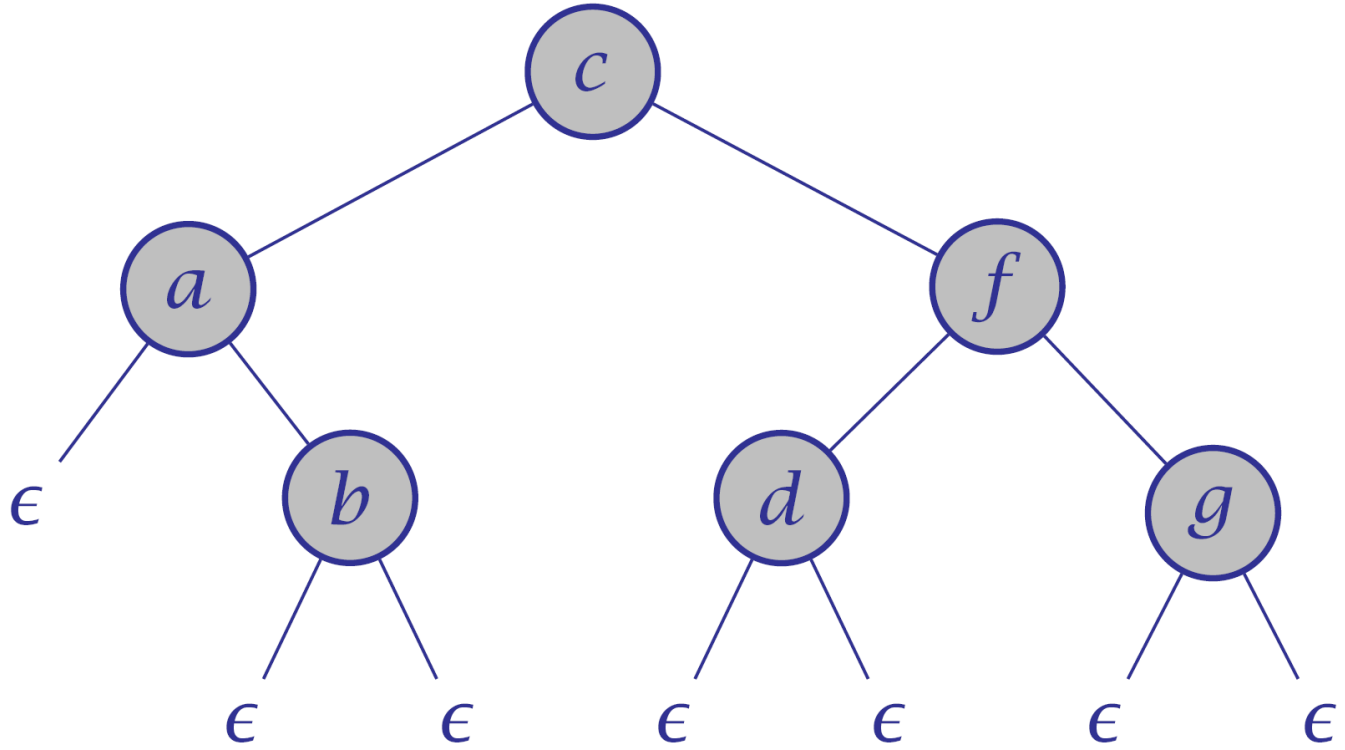
```
data List a = Nil | Cons a (List a)
```



Visualization of Data Structures in Haskell

Example: Binary Tree

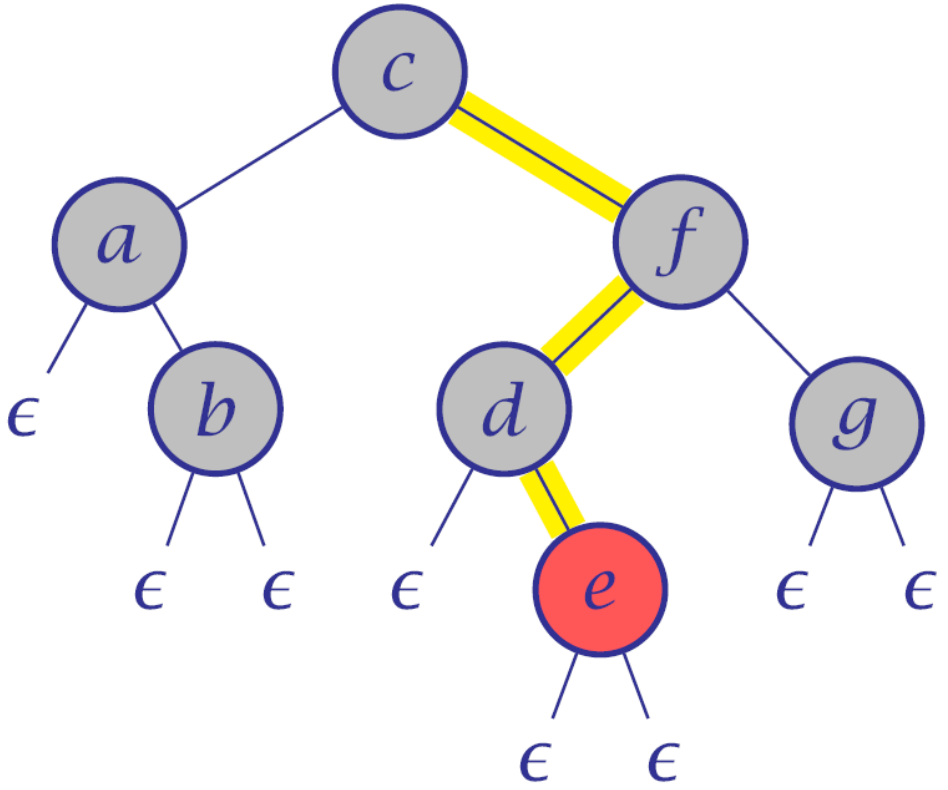
```
data Tree elem = Leaf | Node (Tree elem) elem (Tree elem)
```



Visualization of Data Structures in Haskell

Example: Binary Tree (insert)

```
data Tree elem = Leaf | Node (Tree elem) elem (Tree elem)
```



Visualization of Data Structures in Haskell

Example: more sophisticated

```
data PSQ   k p = Void  | Winner (k,p) (LTree k p) k
data LTree k p = Start | Loser  (k,p) (LTree k p) k (LTree k p)
```