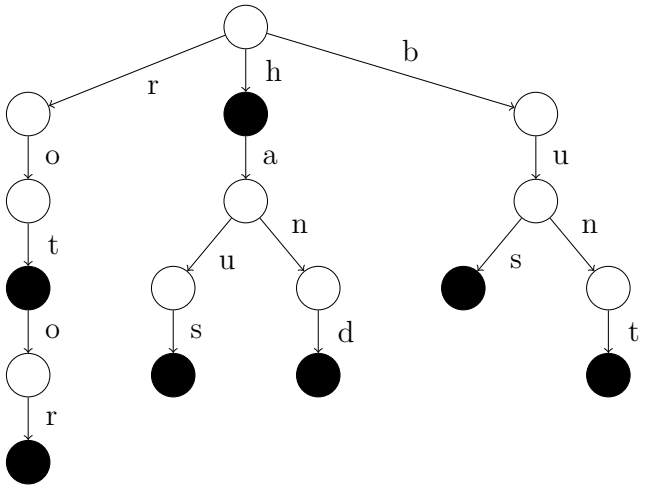
Aufgabe 5 Eins, Zwei, Trie [23 Punkte]

Ein Trie (auch Präfixbaum) ist ein Suchbaum zur effizienten Speicherung von Wörtern. Jeder Knoten im Baum repräsentiert dazu ein bestimmtes Wort. Außerdem sind die Kanten, die von einem Knoten zu dessen Kindern führen, mit verschiedenen Buchstaben versehen.



Möchte man nun durch den Baum navigieren, um zum Beispiel das Wort "rot" zu finden, beginnt man bei der Wurzel, welche das leere Wort repräsentiert. Von dort aus verfolgt man die Kanten, die mit den Buchstaben 'r', 'o' und 't' (in dieser Reihenfolge) versehen sind, um zum Knoten für das gesuchte Wort zu gelangen. Ein Trie enthält damit für jedes gespeicherte Wort auch Knoten für alle Präfixe dieses Wortes (z.B. "", "r" und "ro"). Deshalb erfordert die Speicherung des Wortes "rotor" lediglich zwei zusätzliche Knoten. Um explizit gespeicherte Wörter von denjenigen Wörtern zu unterscheiden, die lediglich Präfix eines gespeicherten Wortes sind, enthält jeder Knoten ein Boolsches Flag, welches für erstere auf $true$ (schwarze Knoten) und letztere auf $false$ (weiße Knoten) gesetzt wird.

Zur Repräsentation eines Trie sei der folgende OCaml-Datentyp gegeben:

```
type trie = Node of bool * (char * trie) list
```

1. Implementieren Sie die Funktion

   ```
   val insert : string -> trie -> trie,
   ```

   welche ein Wort in einen Trie einfügt. Verwenden Sie die Funktion `val explode : string -> char list`, um das Wort in eine Liste der Buchstaben zu zerlegen.

2. Implementieren Sie die Funktion

   ```
   val merge : trie -> trie -> trie,
   ```

   die zwei Tries t_1 und t_2 so verschmilzt, dass ein neuer Trie t entsteht, der ein Wort w genau dann enthält, wenn w entweder in t_1, t_2 oder in beiden Tries enthalten ist.

Hinweis: Da Knoten ihre Kinder in einer (assoziativen) Liste speichern, können einige Funktionen aus dem OCaml `List` Modul von großem Nutzen sein (siehe Anhang).
Hinweis: Sie dürfen entscheiden, ob die Kindlisten der Knoten sortiert sind; Ihre Implementierung muss aber über alle Teilaufgaben konsistent sein!