Opgave 14

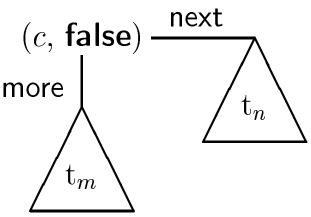
En *trie* er en datastruktur til opbevaring af *ord*, hvor man søger at genbruge fælles præfikser. I Java kan vi definere

```
public class trie {
  private char letter;
  private boolean word;
  private trie next, more;
  ⋮
}
```

En knude indeholder et bogstav (letter) samt en angivelse af, hvorvidt det er sidste bogstav i et ord (word). I undertræet more findes fortsættelsen af de ord, der begynder med letter, og i undertræet next er de øvrige ord.

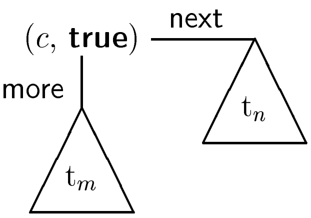
Formelt kan vi definere *words*(t), der er mængden af ord i en trie t:

- hvis t er tom, så er $\mathit{words}(\mathrm{t}) = \emptyset$
- hvis t er af formen



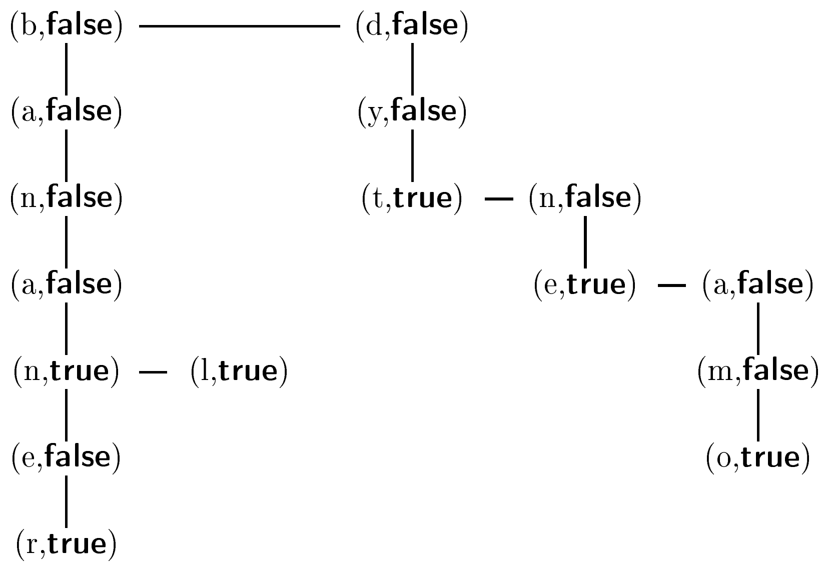
  så er $\mathit{words}(\mathrm{t}) = \{cw \mid w \in \mathit{words}(\mathrm{t}_m)\} \cup \mathit{words}(\mathrm{t}_n)$

- hvis t er af formen



  så er $\mathit{words}(\mathrm{t}) = \{c\} \cup \{cw \mid w \in \mathit{words}(\mathrm{t}_m)\} \cup \mathit{words}(\mathrm{t}_n)$

Hvis t fx er følgende trie



så er

$$\mathit{words}(\mathrm{t}) = \{\mathrm{banan,banal,bananer,dyt,dyne,dynamo}\}$$

a) Indsæt ordene *bamse* og *dyd* i ovenstående trie.

b) Lav en trie-baseret implementation af følgende abstrakte datastruktur:

```
public interface Vocabulary {
  // Query methods
  public boolean member(String s);
  public void print();
  // Update methods
  public void insert(String s);
}
```

hvor metoderne har følgende effekter:

new trie() skaber en ny trie, t, med $\mathit{words}(\mathrm{t}) = \emptyset$.

t.member(w**)** returnerer **true** hvis ordet w er i trien t og ellers **false**.

t.print() Udskriver $\mathit{words}(\mathrm{t})$ tilsvarende ovenfor.

t.insert(w**)** indsætter ordet w i trien t, dvs. hvis t er trien før operationen, og t′ er trien efter operationen, så er $\mathit{words}(\mathrm{t}') = \mathit{words}(\mathrm{t}) \cup \{\mathrm{w}\}$.