```haskell
-- Copyright [2001..2002] Keller & Chakravarty
--

module Lab13
where



data Dictionary = Node Bool [(Char, Dictionary)]
                  deriving (Eq, Show)

dumpDictonary :: Dictionary -> [String]
dumpDictonary (Node isEnd tries) =
  if isEnd
  then
    [""] ++ (dumpDictonaries tries)
  else
    dumpDictonaries tries

dumpDictonaries :: [(Char, Dictionary)] -> [String]
dumpDictonaries [] = []
dumpDictonaries ((c, trie):rest) =
  let
    dic = dumpDictonary trie
    newWords = map (c:) dic
  in newWords ++ (dumpDictonaries rest)
```