1. **Trie Recursion**

   A **trie** is a type of tree where the values of each node are *letters* representing part of a larger *word*. A valid word is a string containing the letters along any path from root to leaf. For simplicity, assume that our trie is represented with the tree abstract data type and where the value of each node contains just a single letter.

   Implement `collect_words`, which takes in a trie `t` and returns a Python list containing all the words contained in the trie.

```
>>> greetings = tree('h', [tree('i'),
...                        tree('e', [tree('l', [tree('l', [tree('o')])]),
...                                   tree('y')])])
>>> print_tree(greetings)
h
  i
  e
    l
      l
        o
    y

def collect_words(t):
    """Return a list of all the words contained in the tree where the value of each node in
    the tree is an individual letter. Words terminate at the leaf of a tree.

    >>> collect_words(greetings)
    ['hi', 'hello', 'hey']
    """

    if _________________________________________________________:

        return _________________________________________________

    words = []

    ________________________________________________________________________________________:

        words += ______________________________________________________________________________

    return words
```