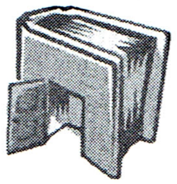
Project 10.1 More About BSTs—Concordance

Objective: In this project you will add more functions to the `BST` class template and then use it in a program that builds a text concordance.



Part 1: More BST Operations

1. Add a function member `level()` to the `BST` class template that returns the level in the BST at which a specified item is located. The root of the BST is at level 0, its children are at level 1, their children are at level 2, and so on.

   Test your function by adding appropriate statements to `treetester.cpp` from Lab 10.1.

2. Add a function member `levelByLevel()` to the `BST` class template to traverse a tree level by level. First, visit the root, then all the nodes on level 1, then all the nodes on level 2, and so on. Nodes on the same level should be visited in order from left to right.

   Hint: Write a nonrecursive function, and use a queue of pointers. To begin, if the BST is not empty, add its root to the queue. Then repeatedly remove the front element from the queue, visit the node it points to, and push its left link (if it isn't null) and then its right link (if it isn't null) onto the queue.

   Test your function by adding appropriate statements to `treetester.cpp`.

Part 2: Application—A Text Concordance

A text concordance is an alphabetical listing of all the distinct words in a piece of text. You are to write a program to construct a concordance for a document stored in a file. The words are to be stored in an array or `vector` of 26 binary search trees.

<u>Version 1:</u>

Write a program that:

1. Declares `concord` to be an array or `vector` of 26 BSTs, one for each letter of the alphabet. Words that begin with 'a' or 'A'; will be stored in the BST at `concord[0]`, those that begin with 'b' or 'B' in the BST at `concord[1]`, ... ; in general, words that begin with character `ch` will be stored in the BST at `concord[upch - 'A']`, where `upch` is the uppercase equivalent of `ch`.

2. Reads strings from a file and for each string:

   a. Removes all characters from it that are not letters and converts all letters to uppercase.

   b. If the string isn't empty:

      i. Searches the appropriate BST for the string.

      ii. If it's not found, inserts the string into the BST.

3. Displays the concordance with the words in alphabetical order.

Version 2:

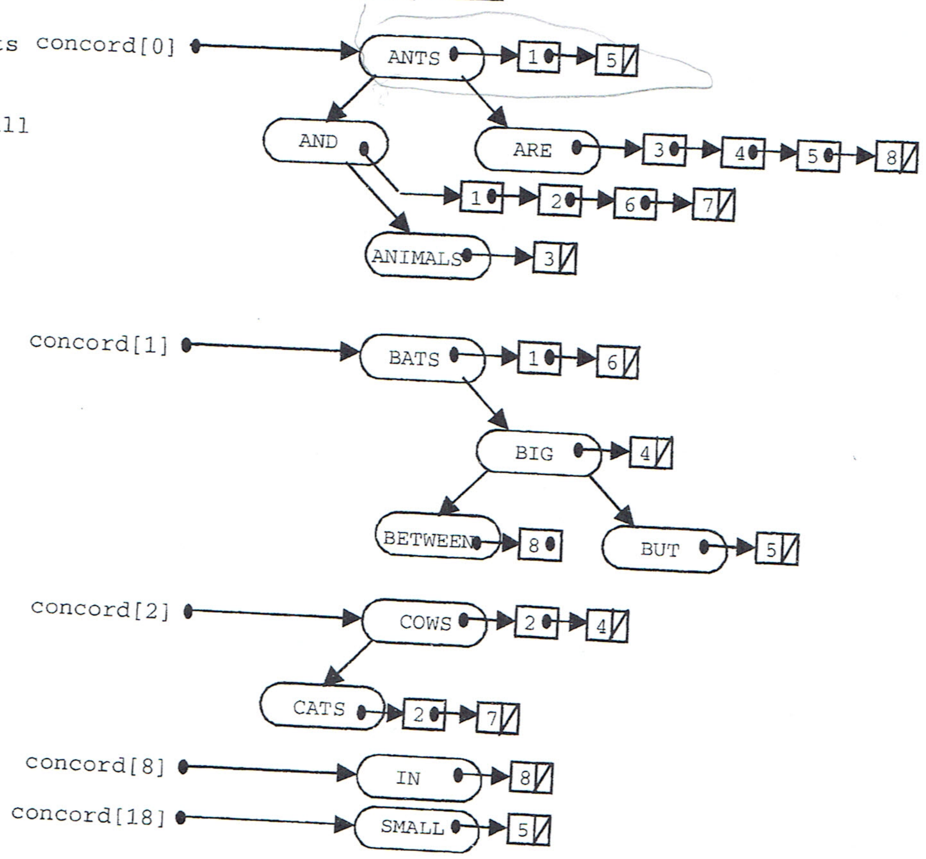
In addition to words, concordances usually store a list of all numbers of pages on which a word appears. Modify the concordance so that the BSTs store words, and for each word, a linked list of all numbers of lines on which that word appeared in the input text. When the concordance is output, the words should be in alphabetical order, and for each word, the line numbers in ascending order.

Example:

For the text file:

```
Ants and bats
and cows and cats
are animals
Cows are big
but ants are small
and cats are in
between
```

Concordance produced:



All other `concord[i]` are empty.