Submit screenshots of the results, and code files in a zip file.

**A screenshot of a computer program

Description automatically generated**

**Figure 1.** Searching the dictionary and adding two words.

A screenshot of a computer

Description automatically generated

**Figure 2.** Displaying dictionary entries via in-order traversal. “apples” becomes the new leftmost leaf node in the binary search tree, so it’s visited first in an in-order traversal.

A screen shot of a computer

Description automatically generated

**Figure 3.** Displaying dictionary entries via in-order traversal. “zebra” becomes the new rightmost leaf node in the binary search tree, so it’s visited last in an in-order traversal.

Creating the dictionary takes advantage of the sorted nature of the input file. The code jumps to the middle of the file less ½ of the desired number of words to insert to the binary search tree. These words are read into a vector, where the middle index becomes the root of the BST. From here, the left half of the tree is populated by recursively dividing the integer representing the middle index by two and using the word at this new index as a left child (less than root). Each subtree rooted in a left child node of the root is then built up by choosing indices less than or greater than the root of its subtree as necessary. Similarly, the right half of the BST is built by moving through the right half of the words vector. In this way, a tree conforming to the BST properties is built without needing any special rotations.