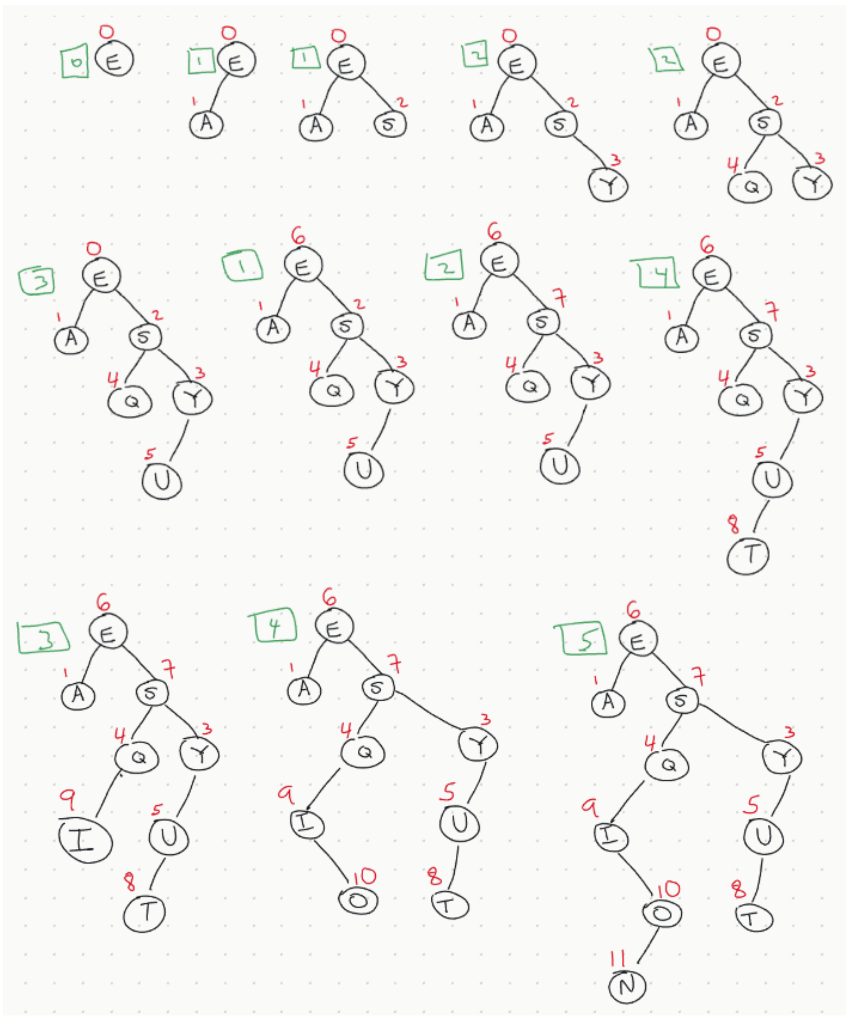

Figure 1: Exercise 1: Sequence of trees formed when inserting the keys `E A S Y Q U E S T I O N` into a binary search tree.

Sergio E. Garcia Tapia
*Algorithms* by Sedgewick and Wayne (4th edition) [SW11]
December 1st, 2024

## 3.2: Binary Search Trees

**Exercise 1.** Draw the BST that results when you insert the keys `E A S Y Q U E S T I O N`, in that order (associating the value `i` with the `i`th key, as per the convention in the text) into an initially empty tree. How many compares are needed to build the tree?

**Solution.** The sequence of binary search trees is shown in Figure 1. The boxes next two the root show the number of compares necessary to complete the insertion at that step. The sequence of compare counts is: 0, 1, 1, 2, 2, 3, 1, 2, 4, 3, 4, 5. Adding gives a total of 28 compares.

**Exercise 2.** Inserting the keys in the order `A X C S E R H` into an initially empty BST gives a worst-case tree where every node has one null link, except one at the bottom, which has two null links. Give five other orderings of these keys that produce worst-case trees.

**Solution.** We can insert them in the following ways:

(i) `A X C S E R H` (given).

(ii) `X S R H E C A`.

(iii) `A C E H R S X`.

(iv) `X A S C R E H`.

(v) `X A S R H E C`.

(vi) `A X C E H R S`.

See Figure 2.

**Exercise 3.** Give five orderings of the keys `A X C S E R H` that, when inserted into an initially empty BST, produce the *best-case* tree.

**Solution.**

(i) `H C S A E R X`.

(ii) `H S C A E R X`.

(iii) `H S C E A X R`.

(iv) `H S R X C A E`.

(v) `H S X R C A E`.

These all result in the same tree, depicted in Figure 3.

**Exercise 4.** Suppose that a certain BST has keys that are integers between `1` and `10`, and we search for `5`. Which sequence below *cannot* be the sequence of keys examined?

(a) `10, 9, 8, 7, 6, 5`

(b) `4, 10, 8, 6, 5`

(c) `1, 10, 2, 9, 3, 8, 4, 7, 6, 5`

(d) `2, 7, 3, 8, 4, 5`

(e) `1, 2, 10, 4, 8, 5`

**Solution.**

(a) This is a valid sequence. For example, this could be a tree where the keys were inserted in descending order.

(b) This is a valid sequence.

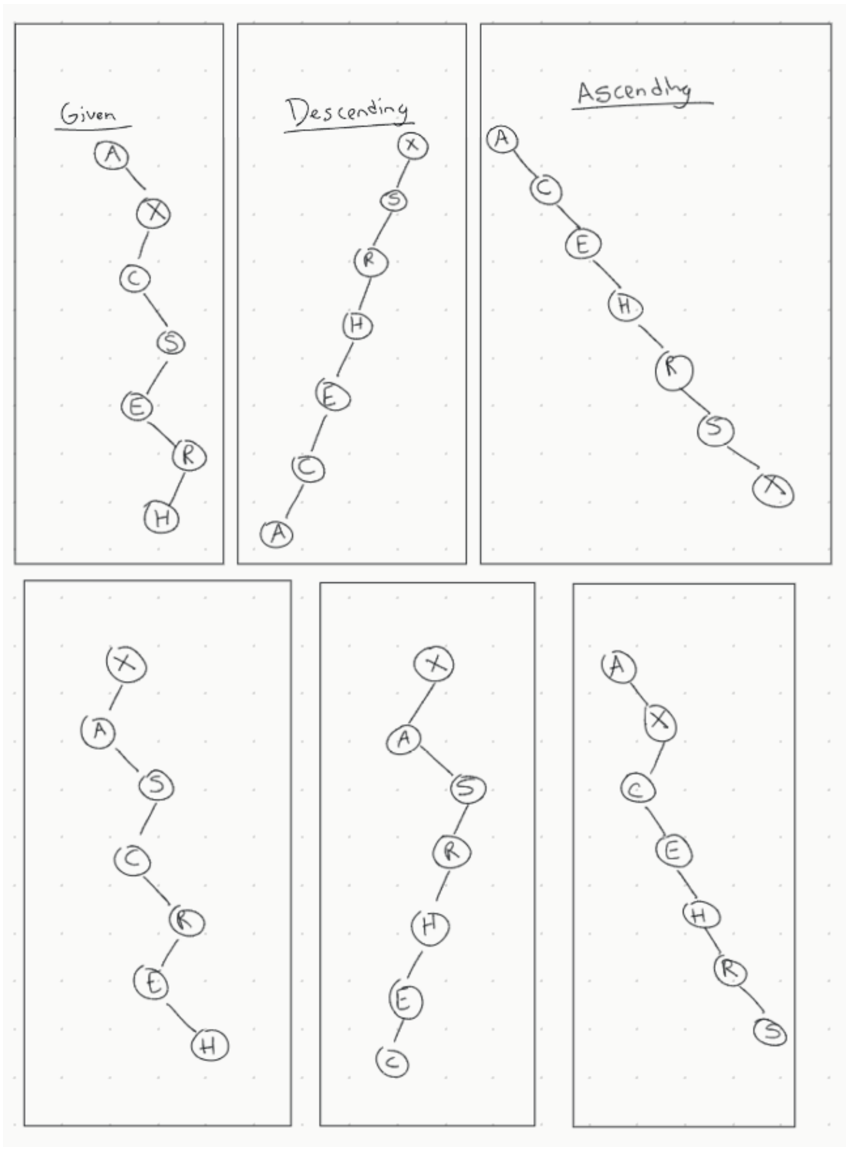

Figure 2: Exercise 2: Worst-case binary search trees created by inserting the keys `A X C S E R H` into an initially empty tree in a particular order

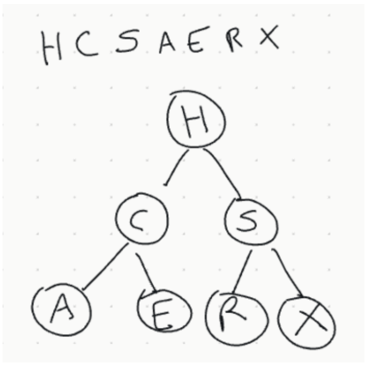

Figure 3: Exercise 3: Best-case binary search trees created by inserting the keys `A X C S E R H` into an initially empty tree in a particular order

(c) This is a valid sequence.

(d) This sequence cannot happen. The sequence of compares suggests the following:

   (i) 2 is the root, and 5 is larger, so we go right.
   (ii) 5 is smaller than 7, so we go left.
   (iii) 5 is smaller than 3, so we go right.
   (iv) 5 is compared against 8.

   This last step is impossible because since 8 is larger than 7, but it appears on the subtree rooted at its left child, consisting of smaller keys, a contradiction.

(e) This is a valid sequence.

**Exercise 5.** Suppose that we have an estimate ahead of time of how often search keys are to be accessed in a BST, and the freedom to insert them in any order that we desire. Should the keys be inserted into the tree in increasing order, decreasing order of likely frequency of access, or some other order? Explain your answer.

**Solution.** Increasing order is not desirable because it leads to a worst-case tree where every internal node hade has 1 null link and the leaf node has 2 null links (in essence, a linked list).

## References

[SW11] Robert Sedgewick and Kevin Wayne. *Algorithms.* 4th ed. Addison-Wesley, 2011. ISBN: 9780321573513.