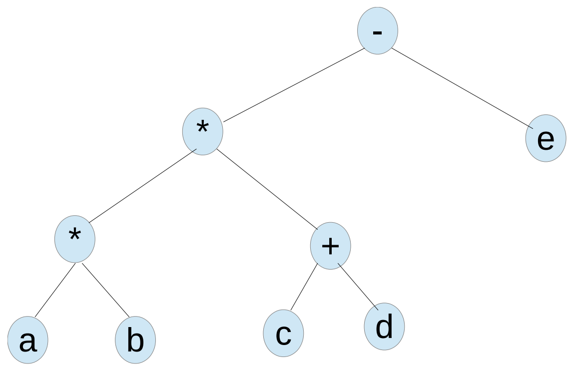
Review Problems for CSCI 3333 Test 1

1. Write a program that, for any given input integer N >= 0, returns the number of 1's in the binary representation of N.

2. This problem has two parts:
   a. Show the result (draw the tree) of inserting 3, 1, 4, 6, 9, 2, 5, 7 into an initially empty binary search.
   b. Show the result (draw the tree) of deleting the root of the binary search obtained in part a.

3. This problem has two parts:
   a. Show the result (draw the tree) of inserting 2, 1, 4, 5, 9, 3, 6, 7 into an initially empty AVL Tree.
   b. Show the result (draw the tree) of deleting 1 and 3 from the tree obtained in part a.

4. Give the prefix, postfix and infix expressions corresponding to the binary express tree in the following:



5. Describe how to use a binary search tree to sort a list of N integers. Estimate the number of comparisons needed on average, and the number of comparisons needed in the worst case.

6. Describe how to use an AVL tree to sort a list of N integers. Estimate the number of comparisons needed on average, and the number of comparisons needed in the worst case.

7. Write efficient algorithms that takes only a pointer to the root of a binary tree T (this is a general binary tree, may not necessarily a binary search tree) and compute:

   a. The number of nodes in T.
   b. The number of leaves in T.

8. Draw the binary expression tree for (a + b * c) + ((d * e + f) * h).

9. Suppose that we want to add the function ***findKth*** to the binary search tree class. This function returns the k-th smallest elements. When the binary search tree has N nodes (or elements), ***findKth(1)*** returns the smallest element, and ***findKth(N)*** returns the largest element. Assume that all elements are distinct. Explain how to design a solution for ***findKth*** that works in $O(k \log N)$ average time. Note that your solution shall not comprise the performance of other binary search tree operations.

10. Given input 4371, 1323, 6173, 4199, 4344, 9679, 1989 and a hashing function $h(x) = (x \bmod 10)$, do the following:
    a. Draw the separate chaining hash table
    b. Draw the hash table using linear probing.
    c. Draw the hash table using quadratic probing.
    d. Draw the hash table using double hashing with the second hash function $h_2(x) = 7 - (x \bmod 7)$.

11. Discuss the advantages and disadvantages of the following hash collision resolution strategies: separated chaining, linear probing, quadratic probing, and double hashing.
12. Suppose we want to find the first occurrence of a string $P_1P_2\cdots P_k$ in a long input string $A_1A_2\cdots A_N$. We can solve this problem by hashing the pattern string, obtaining a hash value H_P, and comparing this value with the hash value formed from $A_1A_2\cdots A_k, A_2A_3\cdots A_{k+1}, A_3A_4\cdots A_{k+2}$, and so on until $A_{N-k+1}A_{N-k+2}\cdots A_N$. If we have a match of has values, we compare the strings character by character to verify the match. We return the position in A if the strings actually do match, and we continue in the unlikely event that match is false.
    a. Show that if the hash value of $A_iA_{i+1}\cdots A_{i+k-1}$ is known, then the hash value of $A_{i+1}A_{i+2}\cdots A_{i+k}$ can be computed in constant time.
    b. Show that the running time is $O(k+N)$ plus the time spent refuting false matches.