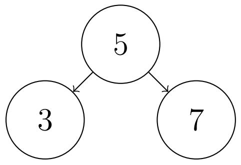
COSC 202, Spring 2024 Exam 2 Practice

The solutions are not intended to be detailed. On the exam you need to provide answers in sufficient detail to get full credit. If you are uncertain what counts as sufficient detail for any particular question, please ask your instructor.

1. Consider the following recursive function called mystery and the Binary Search Tree:

```
mystery(root, low, high)
if root is None then
    return 0
end if
count = 0
if root.key $\geq$ low then
    count += mystery(root.left, low, high)
end if
if root.key $\leq$ high then
    count += mystery(root.right, low, high)
end if
if low $\leq$ root.key $\leq$ high then
    count += 1
end if
return count
```



What will be the return value of mystery(root, 0, 3)?

2. Given a sorted binary array, we want to efficiently count the total number of 1's in it. For instance

   - Input: nums[ ] = [0, 0, 0, 0, 1, 1, 1]
   - Output: The total number of 1's present is 3

   Below is some pseudocode for a recursive algorithm that solves this task.

```
 class Main
{
    // Function to find the total number of 1's
    public static int count(int[] nums, int left, int right)
    {
        if (nums == null || nums.length == 0) {
            return 0;
        }

        if (nums[right] == 0) {
            return 0;
        }

        if (nums[left] == 1) {
            return (right - left + 1);
        }

        int mid = (left + right) / 2;
        return count(nums, left, mid) + count(nums, mid + 1, right);
    }

    public static void main(String[] args)
    {
        int[] nums = { 0, 0, 0, 0, 1, 1, 1 };

        System.out.println("The total number of 1's present is "
                                    + count(nums, 0, nums.length - 1));
    }
}
```

   (a) Justify the correctness of this algorithm. In your response make sure to justify each of the base cases, as well as the recursive step.

   (b) Write a recurrence relation for this algorithm and use this to derive the algorithm's time complexity.

   (c) Let's assume there was only one base case instead of three, where statements **if (nums[right] == 0)** and **if (nums[left] == 1)** did not exist. The code would not return the correct value, of course. It would only return zero. Can you write down the recurrence relation for this code and its subsequent time complexity?

3. Given a BST and a key, design a recursive algorithm that returns the rank of the key — i.e., the number of nodes in the tree less than the key.

4. Below is some pseudocode for LSD sorting with one semantic error.

```
sorted_alphabet = {'a': 1, 'b': 2, 'c':3, 'd':4, 'e':5, 'f':6 ... 'z':26}
R = sorted_alphabet.size() # size of alphabet: 26
aux = new array[lst.length]
for j in [0, 1 ... W-1]: #W is max length of words
    count = new int[R+1]
    for i in [0, 1 ... lst.length-1]:
        ind = sorted_alphabet[lst[j][i]]
        count[ind] +=1
    for i in [0, 1 ... R-1]:
        count[r+1] += count[r]
    for i in [0, 1 ... lst.length-1]:
        aux[count[lst[j][i]]] = lst[j]
        count[lst[i]] +=1
    for i in [0, 1 ... lst.length-1]:
        lst[i] = aux[i]
```

(a) (4 points) How would the erroneous code above sort the following list of strings [“cap”, “bee”, “hen”, “cat”, “ant”]? *Hint: it might be helpful to identify the error before trying to trace the entire code.*

(b) (3 points) How would you fix the error?