Solutions to assignment 2 - CSC 236

1. If the tree has no nodes, it must be the empty tree and there is only one such tree, so $N(0) = 1$. Otherwise, the tree must have been constructed by a node that is connected to three trees. Notice that the tree is described precisely by the left, middle and right tree the root is connected to. If we knew that the left tree has l nodes, the middle tree m nodes and the right tree r nodes, then there would be $N(l)$ possible trees for the left tree, $N(m)$ possible trees for the middle tree, $N(r)$ possible trees for the right tree. Since any choice of tree is independent for these three types, there are as many as $N(l) \cdot N(m) \cdot N(r)$ possible trees when *we assume* that there are l, m, r nodes inthe left, middle and right subtrees. So we only need to enumerate over all possible valid partition of the nodes to these three numbers. A triplet is valid if all numbers are nonnegative (can be zero) and sum to $n - 1$ (do you see why?). To conclude, we get:

$$N(n) = \sum_{l=0}^{n-1} \sum_{m=0}^{n-1-l} N(l) \cdot N(m) \cdot N(n - 1 - (l + m)).$$

   Notice that the recursion above is valid, since always l, m and $n - 1 - (l + m)$ are smaller than n hence the reference to N is always done with an argument smaller than the value to be computed currently. To formally show this recurence relation is correct, we use complete induction and assume that N is correct for all values less than n where n is an aribtrary natural number. If $n = 0$ we directly have show that $N(0) = 1$. Otherwise, the argument above coupled with the induction hypothesis show that indeed $N(n) = \sum_{l=0}^{n-1} \sum_{m=0}^{n-1-l} N(l) \cdot N(m) \cdot N(n - 1 - (l + m))$.

2. If $n = 7^k$ then we will get that $L(n) = L(7^k) = 1 + L(7^{k-1}) = 1 + 1 + L(7^{k-2}) = \ldots = k + L(1) = k = \log_7 n$.Rather than proving it now, we will prove for the general case that $L(n) = \lfloor \log_7 n \rfloor$ for $n > 0$ and $L(0) = 0$. We show this by complete induction. Since $L(n) = 0 = \lfloor \log_7 n \rfloor$ for $0 < n < 7$ and since $L(0) = 0$ we need only to check $L(n)$ for $n > 6$.

$$L(n) = 1 + L(\lfloor n/7 \rfloor) + 1 = \lfloor \log_7 \lfloor n/7 \rfloor \rfloor + 1 = (\lfloor \log_7 n \rfloor - 1) + 1 = \lfloor \log_7 n \rfloor.$$

   For T, we have the recursion,

   $T(n) = 3$ if $n < 7$ and otherwise $T(n) = 4 + T(\lfloor n/7 \rfloor)$. Unwinding will give $T(n) = 4 * \log_7 n + T(1)$ for n which is a power of 7. We guess $T(n) = 4 * \lfloor \log_7 n \rfloor + 3$ for a general n. Proving this by induction is essentially the same as with L.

3. The possible sequences of hops the grasshopper can perform cn be split to those which end with a hop of length 1, and those of hops of length 3. Summing these two types we get that $G(n) = G(n - 1) + G(n - 3)$ for $n \geq 3$ and $G(n) = 1$ if $n < 3$ (since then

only one type of hops cn be used, hence there is only one choice). We first prove that $G(n)$ is a nondecreasing sequence, namely that for all n $G(n) \leq G(n+1)$. Notice that $1 = G(0) = G(1) = G(2)$. Also notice that it is very easy to show that $G(n)$ is always nonegative (the simplest is to remember that $G(n)$ *counts* a certain quantity...). Using that we et that for $n \geq 3$ we have $G(n) = G(n-1) + G(n-3) \geq G(n-1)$.

We now wish to show that for all $n > 0$, $G(n) \leq F(n)$ where $F(n)$ is the nth Fibonacci number that is defined by $F(n) = F(n-1) + F(n-2)$ for $n > 1$ and $F(n) = n$ otherwise. We prove this inductively. For $1 \leq n < 3$ we see that $F(n) = G(n) = 1$. For $n \geq 3$ assume that $G(k) \leq F(k)$ for $0 < k < n$. We have that

$$G(n) = G(n-1) + G(n-3) \leq G(n-1) + G(n-2) \leq F(n-1) + F(n-2) = F(n).$$

Let's explain the above more carefully. The first equality is simply the recurrence relation that holds for $n > 2$. The next inequality is due to the fact that G is non decreasing, hence $G(n-3) \leq G(n-2)$. The next inequality is due to the induction hypothesis (it is important to note that $n-1, n-2$ are both bigger than 0). The last equality is the definition of F. We have shown the induction step and conclude the the claim $G(n) \leq F(n)$ holds for $n > 0$.

4. Consider the predicate $P(N)$: if bs1 runs with parameters satisfying the PreCondition, and $N = e - b + 1$ then Postcondition is satisfied. We wish to show this that for all $N > 0$ P(N) holds (why don't we need to worry about $P(0)$?) Let $N \geq 1$ be and arbitrary number and ssume for all $0 < k < N$ that $P(k)$ holds and show for $P(N)$. P(1) holds since when $1 = b - e + 1$ we have that $b = e$ and so the condition in the second line holds. It is immediate that this part of the code makes sure that if x is in $A[b..e]$ which is the same as $A[b]$ then b is returned and "not found" is returned otherwise, which is consistent with the PostCndition. If $N > 1$ then the condition on line 2 is not met and lines 6-8 ar executed. The key observation is that m as defined satisfies $b \leq m < e$. The rest is the same as with correctness to the "usual" recursive binary search (can be found in the text). To verify the observation, notice that $b = \sqrt{b^2} = \lfloor \sqrt{b^2} \rfloor \leq \lfloor \sqrt{b \cdot e} \rfloor = m$. Furthermore, since $N > 1$ we know that $b < e$ and hence $b \cdot e < e^2$ and so $\sqrt{b \cdot e} < \sqrt{e^2} = e$. But that means that $m = \lfloor \sqrt{b \cdot e} \rfloor < e$.

   Bonus: todo.