Today HW01 due. Reasoning about programs; Dynamic Programming contd. §§ 15.4,5.
Next class Quiz 01. Wrap-up of: basics, program proofs, dynamic programming.
Reminders Homework. Newsgroup. Reading. Coding. Practice. Don't fall behind.

This exercise has extra questions to serve as practice for the upcoming quiz.

1. Recall the notation $[n]$ to represent the set of n integers $[1, 2, 3, \ldots, n]$, for $n \geq 0$. (Thus $[0]$ represents the empty set in this notation.) Let Σ be a finite set, called the *alphabet*.

   A finite *sequence*, of length $n \geq 0$, over Σ is essentially a function from the set $[n]$ to Σ.

   (a) Consider $\Sigma = \{A, B, C\}$, $n = 5$, and a function s that maps $1, 2, 3, 4, 5$ to A, A, C, B, C, respectively (e.g., $s(3) = C$).
   (b) It is conventional to denote $s(i)$ by s_i, so that we may say $s_3 = s_5 = C$, and $s_2 = A$.
   (c) It is also conventional to denote a sequence by simply concatenating the s_i values for $i = 1, 2, \ldots, n$, in order. Thus, we may write $s = AACBC$.

   Consider the sequence (using the third convention) $t = banana$. Represent it using the other two conventions.

2. Recall the textbook's definition of *subsequence* (p. 391). Of the two sequences below, is the second a *subsequence* of the first? Why or why not?

```
Y A B A D A B A D A A B
B A D A D A
```

3. If s is a subsequence of both t_1 and t_2 then s is called a *common subsequence* of t_1 and t_2. Find three different common subsequences of the two sequences of Question 2.

4. Determine, using an arbitrary method, the *longest common subsequence (LCS)* of the two sequences below. Briefly explain why your answer is correct.

```
Y A B A D A B A D A A B
B Y A D A D D A B A Y
```

5. How many sequences (exact number) would be checked by the exhaustive enumeration algorithm (noted near the top of page 392 of the textbook)? Justify your answer.

6. Use the result of Question 8 to generate an *edit script* that edits the first sequence of Question 4 into the second. Describe your algorithm and explain why it is correct.

7. Trace the operation of the LCS-Length algorithm (p. 394) on the sequences of Question 4. Depict the state of the b and c arrays (1) after four iterations of the outer nested loop and (2) at the end of the algorithm.

8. Trace the operation of the Print-LCS algorithm (p. 395) on the result of Question 7. Provide the arguments for each of recursive call of Print-LCS.

9. Use loop invariants and related methods to prove or disprove the correctness of the following implementation of binary search.

```java
public static int search(int[] haystack, int needle) {
    int lo = 0;
    int hi = haystack.length - 1;
    while(lo + 1 < hi) {
        int mid = (lo + hi) / 2;
        if(haystack[mid] > needle) hi = mid;
        else if (haystack[mid] < needle) lo = mid;
        else return mid;
    }
    for(int i = lo; i <= hi; i++) {
            if(haystack[i] == needle) return i;
    }
    return -1;
}
```