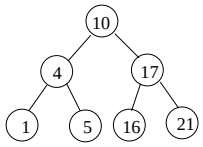
Homework 6 (80 points), Spring 2003

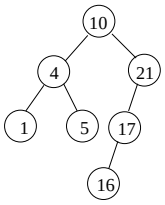
Q1: Exercise 12.1-1 (10 points)

For the set of keys {1, 4, 5, 10, 16, 17, 21}, draw binary search trees of height 2, 3, 4, 5, and 6.

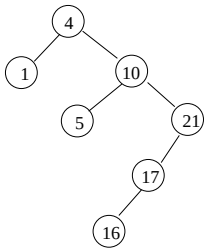
Height 2



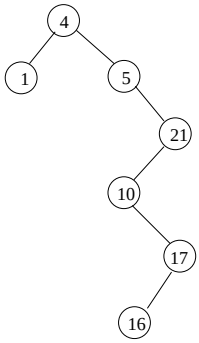
Height 3



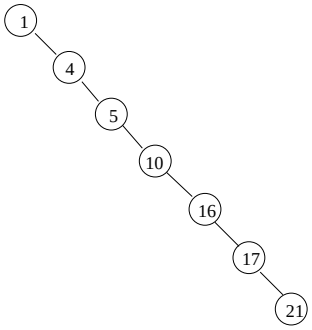
Height 4



height 5



Height 6



Q2: Exercise 12.2-5 (10 points)

Show that if a node in a binary search tree has two children, then its successor has no left child and its predecessor has no right child.

- If a node has left child, then its predecessor should be the rightmost node in its left subtree. This implies that its predecessor has no right child.
  If a node has right child, then its successor should be the leftmost node in its right subtree. This implies that its successor has no left child.

Q3: Exercise 12.3-3 (10 points)

We can sort a given set of n numbers by first building a binary search tree containing these numbers (using `Tree-Insert` repeatedly to insert the numbers one by one) and then printing the numbers by an inorder tree walk. What are the worst case and best case running times for this sorting algorithm?

- The running time for this algorithm
  = the running time of building the tree + the running time of tree walk
  = best case $O(n \lg n)$ (worst case $O(n^2)$) + $O(n)$
  = best case $O(n \lg n)$ (worst case $O(n^2)$)

Q4: Exercise 12.3-4 (10 points)

Suppose that another data structure contains a pointer to a node y in a binary search tree, and suppose that y's predecessor z is deleted from the tree by the procedure `Tree-Delete`. What problem can arise? How can `Tree-Delete` be rewritten to solve this problem?

- Since y is the successor of z, then the node y could be physically removed from the tree when deleting z. The 'another data structure' has a pointer to y originally should point to z to maintain the correct reference.
- To solve this problem, we can rewrite the procedure `Tree-Delete` by physiaclly removing y to z's position and removing z from the tree.

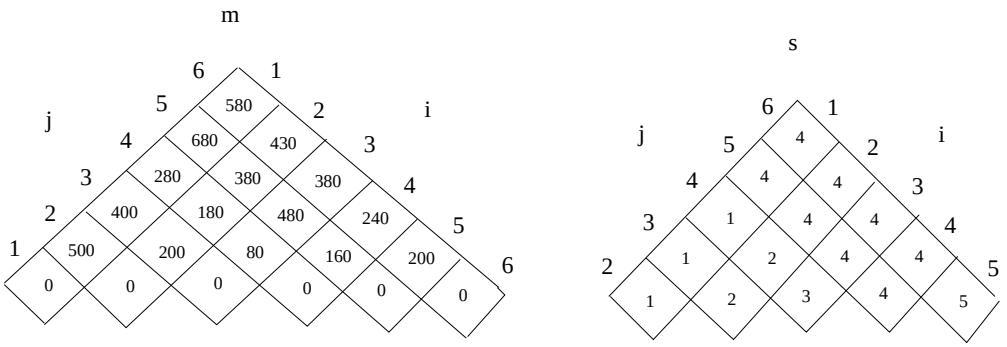
Q5: Exercise 15.1-5 (10 points)

Professor Canty conjectures that there might exist some e_i, $a_{i,j}$, and $t_{i,j}$ values for which `Fastest-Way` produces $l_1[j] = 2$ and $l_2[j] = 1$ for some station number j. Assuming that all transfer costs $t_{i,j}$ are nonnegative, show that the professor is wrong.

- Suppose that $l_1[j] = 2$ and $l_2[j] = 1$ are true for some station j.
  From $l_1[j] = 2$, we know that $f_1[j-1] > f_2[j-1] + t_{2,j-1}$
  From $l_2[j] = 1$, we know that $f_2[j-1] > f_1[j-1] + t_{1,j-1}$
  This implies $f_1[j-1] > f_2[j-1] + t_{2,j-1} > f_1[j-1] + t_{1,j-1} + t_{2,j-1}$. This is incorrect because all transfer costs are nonnegative. Contradiction.
  Thus, the professor is wrong.

Q6: (10 points)

Find an optimal parenthesization of a matrix-chain product whose sequence of dimensions is $< 10, 5, 10, 4, 2, 20, 5 >$. You need to draw the tables as Figure 15.3 in the textbook.



- Based on the table s in the figure, the optimal parenthesization is $((A_1(A_2(A_3A_4)))(A_5A_6))$.

Q7: (10 points)

Determine an LCS of $X =< 1, 0, 1, 0, 0, 1, 0, 1 >$ and $Y =< 1, 1, 0, 0, 1, 0, 1, 0, 1 >$

-

	j	0	1	2	3	4	5	6	7	8	9
i		y_j	1	1	0	0	1	0	1	0	1
0	x_i	0	0	0	0	0	0	0	0	0	0
1	1	0	$\nwarrow 1$	$\nwarrow 1$	$\leftarrow 1$	$\leftarrow 1$	$\nwarrow 1$	$\leftarrow 1$	$\nwarrow 1$	$\leftarrow 1$	$\nwarrow 1$
2	0	0	$\uparrow 1$	$\uparrow 1$	$\nwarrow 2$	$\nwarrow 2$	$\leftarrow 2$	$\nwarrow 2$	$\leftarrow 2$	$\nwarrow 2$	$\leftarrow 2$
3	1	0	$\nwarrow 1$	$\nwarrow 2$	$\uparrow 2$	$\uparrow 2$	$\nwarrow 3$	$\leftarrow 3$	$\nwarrow 3$	$\leftarrow 3$	$\nwarrow 3$
4	0	0	$\uparrow 1$	$\uparrow 2$	$\nwarrow 3$	$\nwarrow 3$	$\uparrow 3$	$\nwarrow 4$	$\leftarrow 4$	$\nwarrow 4$	$\leftarrow 4$
5	0	0	$\uparrow 1$	$\uparrow 2$	$\nwarrow 3$	$\nwarrow 4$	$\leftarrow 4$	$\nwarrow 4$	$\uparrow 4$	$\nwarrow 5$	$\leftarrow 5$
6	1	0	$\nwarrow 1$	$\nwarrow 2$	$\uparrow 3$	$\uparrow 4$	$\nwarrow 5$	$\leftarrow 5$	$\nwarrow 5$	$\uparrow 5$	$\nwarrow 6$
7	0	0	$\uparrow 1$	$\uparrow 2$	$\nwarrow 3$	$\nwarrow 4$	$\uparrow 5$	$\nwarrow 6$	$\leftarrow 6$	$\nwarrow 6$	$\uparrow 6$
8	1	0	$\nwarrow 1$	$\nwarrow 2$	$\uparrow 3$	$\uparrow 4$	$\nwarrow 5$	$\uparrow 6$	$\nwarrow 7$	$\leftarrow 7$	$\nwarrow 7$

- From the above table, the LCS of X and Y is $< 1, 0, 1, 0, 1, 0, 1 >$ with length 7.

Q8: Exercise 15.4-5 (10 points)

Given an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers.

- Let the sequence X is the sequence of n numbers.
  Let the sequence Y is the sorted (increasing order) sequence of that n numbers.
  We can transform the problem "finding the longest monotonically increasing subsequence of X" to an equivalent problem "finding the LCS of X and Y".
  Sorting n numbers takes $O(n \lg n)$. Finding the LCS of X and Y takes $O(n^2)$. Thus, the total running time is $O(n^2)$.