# Data Structures and Programming

Midterm Exam (April 26, 2010)

1. (15 pts) Order the following functions by asymptotic growth rate, in nondecreasing order. Indicate by " $=$ " those functions that are big-Theta (i.e., $\Theta()$) of one another. Your solution should look like: $\cdots < \cdots < \cdots = \cdots = \cdots < \cdots < \cdots = \cdots = \cdots < \cdots < \cdots$.
   $n + 0.001n^5$; $n^{0.001}$; $12n^2 + 2^n$; $n^4\sqrt{1000n}$; $4n^3 + 4n^2$ $4^n$; $10nlog_2n$; $n(log_2n^2)$; $2^{2010}$; $n^5log_2n$; $9n^{9/2}\sqrt{n} + 9n^2$; $\frac{n^5}{log_2n}$; $2^{(log_2n)(log_2log_2n)}$; $0.1^{\sqrt{n}}$; $n^5 + 3n$;
   **Solution:** $0.1^{\sqrt{n}} < 2^{2010} < n^{0.001} < 10nlog_2n = n(log_2n^2) < 4n^3 + 4n^2 < n^4\sqrt{1000n} < \frac{n^5}{log_2n} < n + 0.001n^5 = n^5 + 3n = 9n^{9/2}\sqrt{n} + 9n^2 < n^5log_2n < 2^{(log_2n)(log_2log_2n)} < 12n^2 + 2^n < 4^n$

2. (5 pts) Explain the relationship and differences between *abstract data type* and *data structure.*
   **Solution:** See textbook and class notes.

```
foo(int N)
    do stuff in A steps
    for i = 1..N
        do stuff in B steps
        for j = 1.. N
            do stuff in C steps

boo(int N)
    do stuff in A steps
    for i = 1..N
        do stuff in B step
        for j = 1.. i
            do stuff in C steps

goo(int N)
    do stuff in A steps
    for i = 1..N
        do stuff in B steps
        for j = 1.. 100
            do stuff in C step
```

Figure 1: Three programs

3. (10 pts)
   (a) (4 pts) For each of the three methods (foo(), boo(), and goo()) shown in Figure 1, what is the exact runtime of each method (in steps) in terms of A,B,C, and N? You should not count loop related operations (e.g., declaring, incrementing and comparing i or j).
   **Solution:**
   $foo() : A + BN + CN^2$;

$boo() : A + BN + C(N(N + 1)/2);$
$goo() : A + BN + 100CN$

(b) (2 pts) For N = 10, which is the fastest algorithm?
**Solution:**
$foo() : 100C + 10B + A$
$boo() : 55C + 10B + A$ (Fastest Algorithm)
$goo() : 100C + 10B + A$

(c) (2 pts) Assuming that A, B, and C are constant, what is the asymptotic runtime of each method in terms of N?
**Solution:** $foo() : O(N^2)$ ; $boo() : O(N^2)$; $goo() : O(N)$

(d) (2 pts) As N grows infinitely large, which is the fastest algorithm?
**Solution:** $goo()$

4. (10 pts) Consider the use of the exclusive-or encoding method to represent doubly linked lists as discussed in class. Given a node $X$, let $Link(X)$ be the exclusive-or encoding kept in node $X$, i.e., $Link(X)$ is the outcome of taking the exclusive-or of the addresses of $X$'s predecessor and successor along the list. Consider a doubly linked list consisting of the following: $X_1 \;\; X_2 \;\; X_3 \;\; X_4 \;\; X_5 \;\; X_6$. Suppose pointers $P$ and $Q$ point to nodes $X_3$ and $X_4$, respectively. Explain the updates needed to "SWAP" $X_3$ and $X_4$, making the new list $X_1 \;\; X_2 \;\; X_4 \;\; X_3 \;\; X_5 \;\; X_6$. Note that after the swapping, $P$ and $Q$ point to $X_4$ and $X_3$, respectively. Use additional temporary pointer(s), if needed. You may use the notation $\begin{pmatrix} x \\ y \\ \dots \end{pmatrix} \leftarrow \begin{pmatrix} exp_1 \\ exp_2 \\ \dots \end{pmatrix}$ defined in class to describe your algorithm.
**Solution** $A_2$ and $A_5$ are addresses of $X_2$ and $X_5$, respectively.

$$\begin{pmatrix} A_2 \\ A_5 \end{pmatrix} \leftarrow \begin{pmatrix} LINK(P) \oplus Q \\ LINK(Q) \oplus P \end{pmatrix}$$

$$\begin{pmatrix} LINK(A_2) \\ LINK(A_5) \\ LINK(P) \\ LINK(Q) \end{pmatrix} \leftarrow \begin{pmatrix} (LINK(A_2) \oplus P) \oplus Q \\ (LINK(A_5) \oplus Q) \oplus P \\ Q \oplus A_5 \\ A_2 \oplus QP \end{pmatrix}$$

$$\begin{pmatrix} P \\ Q \end{pmatrix} \leftarrow \begin{pmatrix} Q \\ P \end{pmatrix}$$

5. (5 pts) Draw a binary tree that contain the letters "b", "n", "o", "s", "u" such that the inorder traversal spells "bonus" and the preorder traversal spells "obuns".
**Solution:** "o" is the root node, "b" is the left child of the root, "u" is the right child of the root. "n" is the left child of "u". "s" is the right child of "u".

6. (10 points) Consider the AVL tree shown in Figure 2. Show the modified tree under each of the following operations. (Note: The two operations are independent. Each of them starts from the original tree.) Show your work in sufficient detail.
(a) Deletion of the key 4 (b) Insertion of the key 16.

7. (10 pts) Let $A$ and $B$ be two unsorted arrays of $m$ and $n$ elements, respectively. Let $h = max\{m, n\}$. Design an algorithm in $O(h \; log \; h)$ time to find the symmetric difference of $A$ and $B$, i.e., $(A \cup B) -$

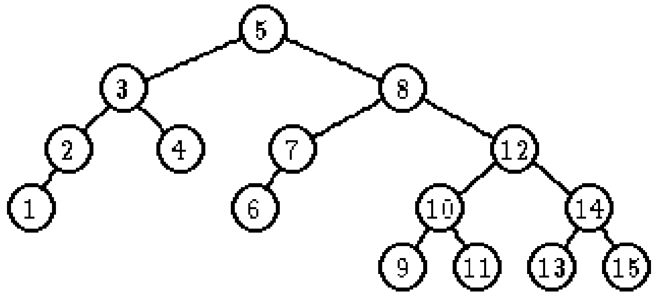



Figure 2: An AVL Tree

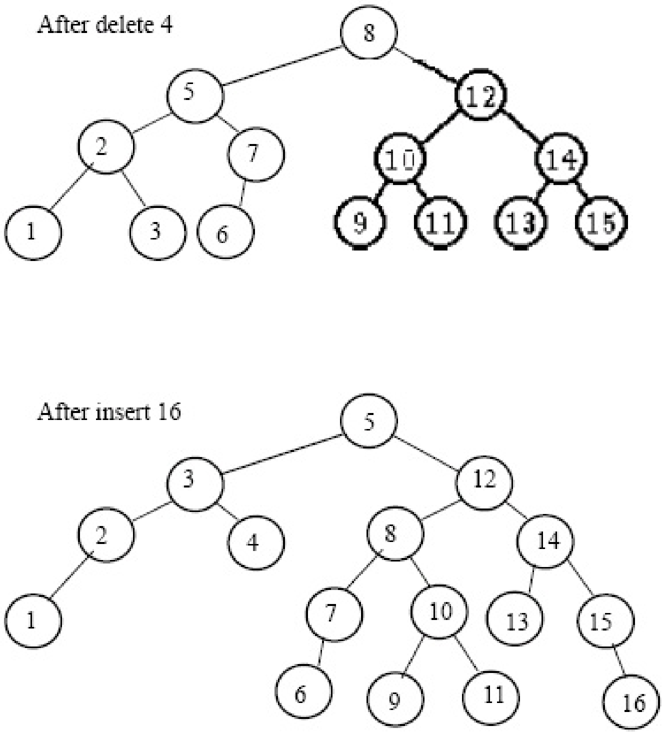



Figure 3: Solutions of Problem 6

$(A \cap B)$, which is the set of elements in either $A$ or $B$ but not both.
**Solution:**

(a) Build a balanced tree (such as AVL tree) $T$ from $A$ – $O(mlogm)$ time

(b) For each element $x \in B$, if $x$ is in $T$, delete $x$; else insert $x$ – $O(nlogh)$ time

8. (10 pts) Let T be the smallest AVL tree of height $h$. How many nodes does it have, if the smallest AVL tree of height $h-2$ has $m$ nodes and the smallest AVL tree of height $h-3$ has $k$ nodes? Show your derivation in sufficient detail. (Hint: Let $n(h)$ be the smallest number of nodes in AVL tree of height $h$. Express $n(h)$ in terms of $m$ and $k$.)
**Solution:** Let $n(h)$ be the smallest number of nodes in AVL tree of height $h$. Then $n(h-1) = 1+n(h-2)+n(h-3) = 1+m+k$ and $h(n) = 1+n(h-1)+n(h-2) = 1+1+m+k+m = 2+2m+k$.

9. (10 pts) Draw the tree that results when you first insert the keys
J E F A D K G
in that order into an initially empty binary search tree using the standard algorithm (i.e., without balancing the tree), then insert H using (bottom-up) **splay** insertion. Show your derivation in sufficient detail.
**Solution**

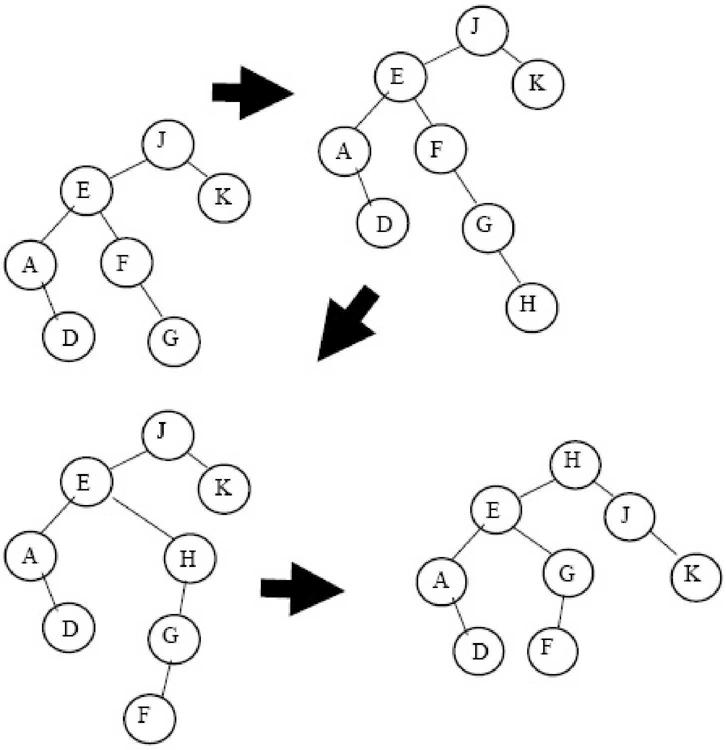
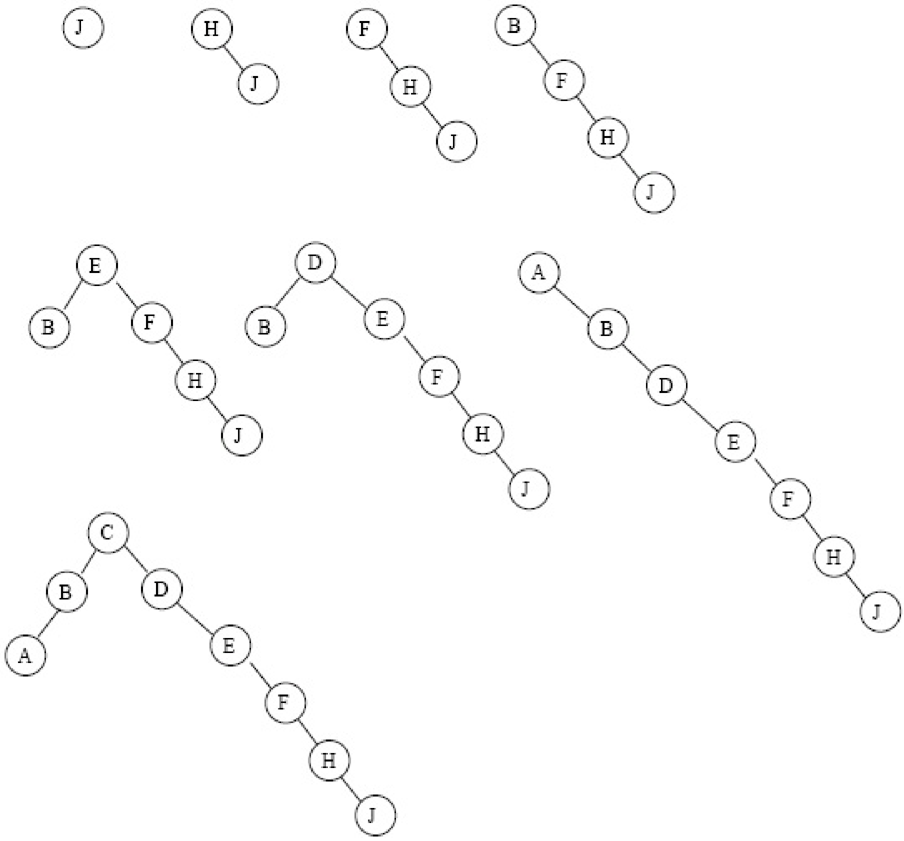
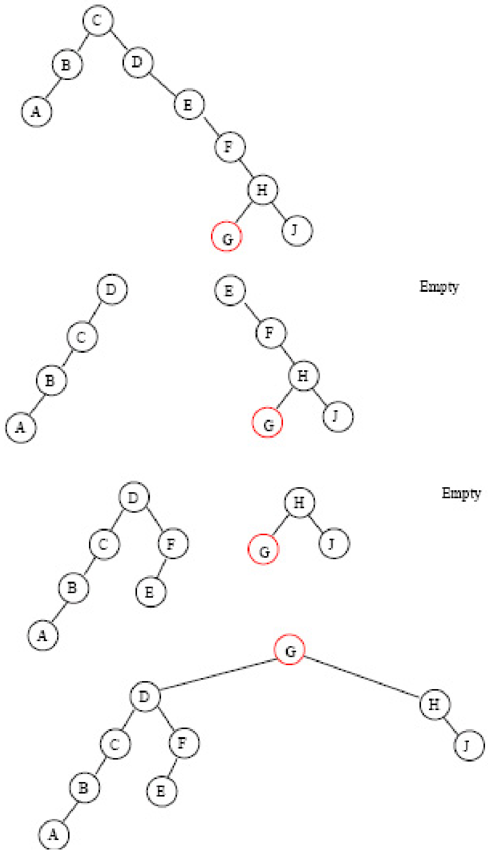



Figure 4: Solution of Problem 9

10. (15 pts) Draw the **splay tree** that results when you perform **top-down** insertion of the keys
J H F B E D A C G I
in that order into an initially empty tree. Show your derivation in sufficient detail.
**Solution**




Empty
Empty

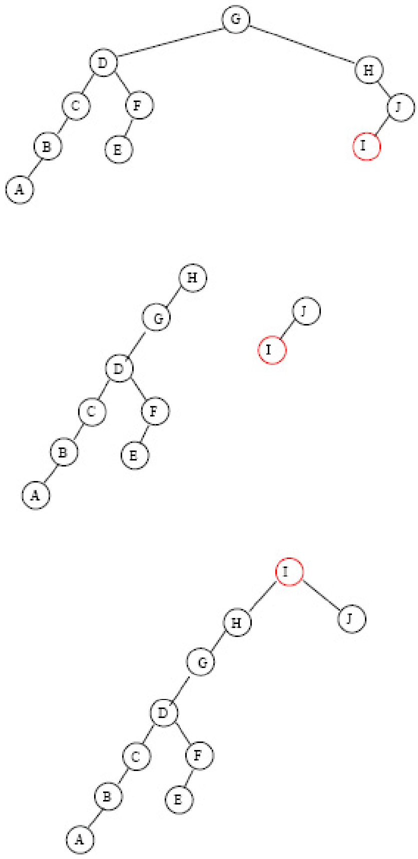



Figure 5: Solution of Problem 10