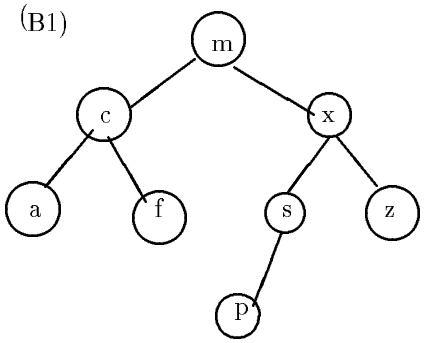
Graduate Comprehensive Exam
Data Structures and Algorithms
Spring 1998 (Section A)

Total: 100 points. Good Luck.

1. (50 pts) Answer the following questions about the given binary tree *B1*

(B1)



(a) Give the **preorder** traversal of this tree
(b) Give the **inorder** traversal of this tree
(c) Give the **postorder** traversal of this tree
(d) Give the **depth-first search** order of this tree
(e) Give the **breadth-first search** order of this tree

(f) Is *B1* a binary *search* tree? Explain.

(g) Show *B1* after inserting node n (call the resulting tree *B2*)

(h) Show *B1* after deleting node m (call the resulting tree *B3*)

(i) Which is more difficult in a search tree: insertion or deletion? Explain.

(j) Is *B1* balanced (AVL)? Explain. If your answer is "not balanced", convert it into a balanced tree.

(k) Is *B2* balanced (AVL)? Explain. If your answer is "not balanced", convert it into a balanced tree.

(l) Why is a complete balanced search tree preferable to a random search tree?

For Question 2, you may use pseudocode (with sufficient details) or a high-level programming language (like C, C++, or Ada) to *write* a function.

2. (25 pts) Given an integer array, its length, and an integer item to be found, *write* a function that performs **binary search** and returns the item's location (or -1 if not found)

   (a) iteratively (no recursion) and

   (b) recursively (with recursion).

3. (10 pts) Using big-O notation, estimate the running time of `proc(N)` in terms of `N`. Explain your answer.

```
void proc(int x)
  {
      if (x > 1)
        {
          proc(x / 4);
          // some constant-time operation here
        }
  }
```

4. (15 pts) Given these numbers: `3 8 2 6 5 1`, perform Heapsort (in ascending order) and show the heap after each element is sorted.