Data Structures

Midterm Exam, Fall 2005

01. **(30%)** Explain the following terms and terms comparisons:

- **(a)** Complete binary trees
- **(b)** Binary search trees
- **(c)** FIFO lists vs. LIFO lists
- **(d)** Doubly linked lists
- **(e)** Min heaps vs. Min trees
- **(f)** AVL trees
- **(g)** Performance analysis
- **(h)** Tree traversal
- **(i)** Activation records
- **(j)** Indirect recursion

02. **(8%)** Prove or disprove the following statements:

(a) $\sum_{i=0}^{n} i^3 = \Theta(n^4)$

(b) $n^{1.001} + n\log n = \Theta(n^{1.001})$

(c) $n^2 / \log n = \Theta(n^2)$

(d) $n! = \mathrm{O}(n^n)$

03. **(9%)** Assume that it takes two units of memory location to store an integer and row major order is adopted. Consider the following array declaration:

```
int A[5][6][10];
```

(a) If `A[0][0][0]` is stored at address 2000, calculate the memory address of `A[2][3][7]`.

(b) If `A[0][0][0]` is stored at address 2000, indicate which array element is at the location 2300.

(c) If `A[3][0][0]` is stored at address 2000, calculate the memory address of `A[1][5][9]`.

04. **(5%)** For any nonempty binary tree, T, if n_0 is the number of leaf nodes and n_2 is the number of nodes of degree 2, prove that $n_0 = n_2 + 1$.

05. **(a)** **(8%)** How to represent sparse matrices as linked lists? Your answer should include the node structure and a pseudo code to read in a matrix and set up its linked representation.

(b) **(3%)** Assume that the given argument of the sparse matrix read-in is an $m \times n$ matrix with k non-zero entries. Determine the time complexity of your pseudo code answered in **(a)**.

06. **(4%)** Given an in-order sequence BAECDJHFGI and a post-order sequence ABCDEFGHIJ, can you derive a unique binary tree? If yes, draw the binary tree; or you have to give two distinct binary trees which can generate above sequences.

07. **(6%)** Write the postfix form of the following expressions:

(a) $A - B * D + E / F + A * D + C$

(b) $(A - B) * D + E / (F + A * D) + C$

08. **(3%)** Derive the worst case time complexity of the binary search function `binsearch` as follows. (Assume that there are *n* elements in the array `list`.)

```
int binsearch(int list[], int searchnum, int left, int right) {
  int middle;
  while (left <= right) {
      middle = (left + right) / 2;
      if (list[middle] < searchnum)
          left = middle + 1;
      else if (list[middle] == searchnum)
          return middle;
      else
          right = middle – 1;
   }
   return -1;
}
```

09. **(a)** **(8%)** Describe how to delete an element from a binary search tree. Calculate the time complexity of the deletion operation.

(b) **(8%)** Describe how to insert an element into a min heap. Calculate the time complexity of the insertion operation.

10. **(a)** **(8%)** Explain how to implement a circular queue by using an array.

(b) **(8%)** Explain how to implement a doubly linked circular list.

(c) **(2%)** Give two applications of stacks.