Name ______________________________________ Score ________

CS 315 Data Structures

mid-semester test # 2

(November 13, duration: 75 minutes)

1) Consider the sequence of keys [65, 81, 117, 28, 81, 78] inserted into a closed hash table (of size 13) using double hashing in which the primary hash function is h(x) = x % 13, and the secondary hash function is s(x) = 5 - (x % 5). Exhibit the table (including the status value of each bucket). What is the load factor? *13 points*

2) Solve the above problem for an open hash table. (Of course, in this case, the secondary hash function is not relevant.) Assume that insertion is at the end of the list.

8 points

3) Exhibit the binary (min)heap that results from applying build-heap procedure on the following array A using both algorithms presented in class. Exhibit the result of performing deleteMin from one of the minheaps. Show the heap as a binary tree.

Array A[1] = 23 A[2] = 19 A[3] = 20 A[4] = 36 A[5] = 7 A[6] = 11 A[7] = 9

5 + 5 + 3 points

4) Let A be an array of integers in which some keys are stored in A[1] to A[k]. Write a procedure that takes as input A and k, and determines if A[1 : k] forms a min-heap. What is the time complexity of your solution? (Include all the instructions such as assignment, comparison etc. and express your answer exactly as a function of k.) *7 + 3 points*

```
bool heapCheck( int[] A, int k)
```

5) What is the smallest (largest) number of nodes in a heap of height 7? What is the height of a heap with 280 nodes? *6 points*

6) (a) Let n be the number of elements in a min-heap. Write a Boolean expression involving j and n such that the Boolean expression is true if and only if j is a leaf node. *3 points*

(b) Design an <u>efficient</u> algorithm that takes as input a heap H (with n integers) and an index j that represents a leaf node and deletes the key in index j. First describe the procedure informally, then write the function using a syntax close to that of c++. Estimate <u>precisely</u> the time complexity of your algorithm. 2 + 8 + 3 *points*

Informal description:

```
int deleteKey(heap H, int j) {
// deletes the key H[j] and returns it
// assuming j is a leaf of the heap
```

7) (a) Show step by step the computations performed by the algorithm to convert a post-fix expression into an expression tree applied to the expression shown below: *6 points*

(A + B)* C + D * F

(b) Show the result of performing a pre-order traversal on the tree created in (a) *3 points*