I have pulled questions from a variety of exams to fit our current coverage. So your exam may differ from this one in length.

Exam 2

True-False (2 points each) (Circle the correct answer)

T  F  1. Hashing can be a useful way of sorting.
T  F  2. If an operation takes \( O(1) \) worst case time, then it takes \( O(1) \) amortized time.
T  F  3. Lazy deletion is not needed when using chaining.
T  F  4. If two algorithms for the same problem both run in \( O(n^2) \) time then they will take nearly the same amount of time to run on any input.
T  F  5. An \( O(n^2) \) algorithm is faster than an \( O(n^3) \) algorithm if \( n \) is large enough.

Fill in the Blank (1 point each)

1. In deleting an entry in a hash table, we often mark the cell as “empty, but previously held something”. This strategy of dealing with deletions is termed ____________________________.
2. A ____________________________ is a search tree which moves the most recently accessed item to the root.
3. The ____________________________ is the distance from a node in the tree to a NULL.
4. ____________________________ refers to finding the average running time per operation over a sequence of operations.
5. A ____________________________ is a self-adjusting version of a leftist heap.
6. A ____________________________ tree is one in which all leaf nodes are at depth \( n \) or \( n-1 \) and all the leaves at depth \( n \) are towards the left.
7. I want a search tree in which the average time for a find is \( \log n \). A tree that would work is ____________________________.
8. A data structure which adds in any order but deletes based on importance is termed ____________________________.

Multiple Choice (3 points each)

1. When is insertion sort a good choice for sorting an array?
   a. Each component of the array requires a large amount of memory.
   b. Each component of the array requires a small amount of memory.
   c. The array has only a few items out of place.
   d. The processor speed is fast.
2. The best case, in terms of the asymptotic running time, for QuickSort happens when:
   a. The array is already sorted
   b. The array is sorted in reverse
   c. All elements of the array are equal
   d. The number of elements to the right of the pivot is approximately equivalent to the
   number of elements to the left of the pivot
   e. All elements of the array are negative

3. Recall all sorting algorithms you have studied. Suppose that you have an array of 128 elements. Which number is closest to the minimum number of binary element comparisons that would be necessary to sort the elements if they are originally in random order?
   a. 10
   b. 4075
   c. 389
   d. 891
   e. 7887

4. For a d-heap (where d is 4 and addressing begins at 0), a parent of i is located at
   a. \( i/4 \)
   b. \((i-1)>>2\)
   c. \((i-1)>>4\)
   d. \( i*4+1 \)
   e. none of the above

5. You write a hash table implementation that uses separate chaining. Which technique is most efficient for handling collisions?
   a. Linear probing
   b. Quadratic probing
   c. Double hashing
   d. None of the above

6. Every node in a (min) binary heap
   e. Has two children
   f. Is no larger than its children
   g. Is no smaller than its children
   h. Has a smaller left child than right child
   i. Two or more of the above
7. Which tree could **NOT** result after deleting 10 from following simple binary search tree:

![Binary Search Tree Diagram]

a. ![Tree Diagram a]

b. ![Tree Diagram b]

c. ![Tree Diagram c]

d. They could all result from the deletion.

e. None of them could result from the deletion.
8. What item is at the root after the following sequence of insertions into an empty splay tree: 1, 11, 3, 10, 8, 4, 6, 5, 7, 9, 2?
   j. 1
   k. 2
   l. 4
   m. 8
   n. None of the above

9. Which of the statements about splay trees is false?
   a. A single access operation could examine every node in the tree
   b. Any n consecutive operations from an initially empty splay tree must take at most O(n log n) time
   c. Inserting the items 1, 2, \ldots, n into an initially empty splay tree takes O(n) total time
   d. The most recently accessed item is at the root
   e. None of a to d is false

10. Finding a node (in an AVL tree) has what complexity?
    a. O(n log n)
    b. O(n)
    c. O(log n)
    d. O(1)
    e. none of the above

11. Which is true of a mergesort and a quicksort?
    a. both have the same average case complexity
    b. both are recursive
    c. both take O(n) to divide the problem or combine the pieces
    d. all of the above

12. Here is an array which has just been partitioned by the first step of quicksort: 3, 0, 2, 4, 5, 8, 7, 6, 9
Which of these elements could be the pivot?
    a. 3
    b. 4
    c. 5
    d. 6
    e. (b) or (c)

13. What is the worst-case time for mergesort to sort an array of n elements?
    a. O(log n)
    b. O(n)
    c. O(n log n)
    d. O(n²)

14. What is the worst-case time for quicksort to sort an array of n elements?
    a. O(log n)
    b. O(n)
    c. O(n log n)
    d. O(n²)
15. Insertion sort fits in a category of sorting algorithms. What is this category?
   a. Stable, non-oblivious
   b. Non-Stable, Oblivious
   c. O(n log n)
   d. Sorts whose best case time is the same as the worst case time
   e. More than one of the above

16. If well-implemented, a hash table is O(1) for a lookup, and a binary search tree is O(log n). What is the most significant reason why you might want to use a binary search tree anyway?
   a. Binary search trees use less memory.
   b. Your hash table might not be big enough.
   c. You want to retrieve your data in sorted order.
   d. Your data consists only of integers.

17. Suppose we are sorting an array of eight integers using a quadratic sorting algorithm. After four iterations of the algorithm's main loop, the array elements are ordered as shown here:
   1 3 6 7 14 11 9 12
   Which statement is correct?
   A. The algorithm is a heap sort.
   B. The algorithm is a mergesort.
   C. The algorithm is an insertion sort.
   D. The algorithm is a quicksort.

18. Consider this binary search tree:
   14
     / \
    2   16
      / \
     1   25
       / 
      24
Suppose we remove the root, replacing it with its in-order successor. What will be the new root?
   A. 16   B. 24   C. 25   D. 2   E. 2

19. In a hash table using linear probing, there is a difference between those spots which have never been used and those spots which have previously been used but no longer contain an item. Which method has a better implementation because of this difference?
   a. insert   b. find   c. rehash   d. size
1. (14 points) Answer the following questions about the code below.

```c
void swap(int a[], int i, int j)
{
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;
}

void sort(int a[], int size)
{
    int big = size;
    for (int i = 0; i < size; i++)
        if (a[big] < a[i]) big = i;
    swap(a, size, big);
    sort(a, size-1);
}
```

1. Briefly describe how the sort works.
2. Correct any errors.
3. What does size represent?
5. Is the sort oblivious? Explain.

2. (10 points) A tree stored as left-most-child/next-right-sibling is supposed to represent a min heap. Write the code to return true if it is a min heap and false otherwise.
3. (5 points) Two different algorithms are written to solve the same problem. Algorithm A is $O(n)$. Algorithm B is $O(n^2)$. You expect algorithm A to work better. However, when you run a specific example on the same machine, algorithm B runs quicker. Give two different reasons to explain how such a thing could happen.
4. (8 points) Consider the leftist min heaps below in which the key is stored in the top half of a node and the null path length is stored in the bottom half of the node. Show the results after merging the heaps.

5. Show the results of merging the following skew heaps:
6. Show the results of merging a single node tree (containing 2) with the Skew Heap below.

```
   9 <- merge ->  2
    / \       /
   8   7     6   5
    / \     / \
   4   3   4   3
```

7. Show the result of merging the two binomial queues below:

```
Two binomial queues
```

```
   16   12
   \   /    \
    18  21  24
     \   /  \
      23  51  65
   13   14
   \   /  \
    26  24  65
```