Name: ______________________________

# CIS 263 Practice Test 1

Updated: October 7, 2019

The test is tentatively scheduled Monday, 14 October.

In addition to the problems below, be sure to review the homework problems as well as the couse notes.

1. State *and explain* the definition of big-O.
2. Explain why we use big-O to compare algorithms.
3. Explain why binary search runs in $O(\log n)$ time.
4. Under what conditions is it possible to sort a list in less than $O(n \log n)$ time?
5. List and explain the worst-case and average-case running times for each Vector method below:
   (a) insert(iterator here, Object item)
   (b) insertAtHead
   (c) insertAtTail (aka push_back)
   (d) get(iterator here)
   (e) get(index i)
   (f) remove(iterator here)
   (g) remove(index i)
   (h) splice(iterator place_here, iterator from_here, iterator to_here)

   (Be sure you understand when (and why) `push_back` runs in constant average time.)
6. List and explain the worst-case and average-case running times for each LinkedList method below:
   (a) insert(iterator here, Object item)
   (b) insertAtHead
   (c) insertAtTail (aka push_back)
   (d) get(iterator here)
   (e) get(index i)
   (f) remove(iterator here)
   (g) remove(index i)
   (h) splice(iterator place_here, iterator from_here, iterator to_here)
7. When should you use a Vector, and when should you use a Linked List?

Name: ______________________________

8. Assume you have a singly linked list with no tail pointer. Implement `removeTail()`. Raise an exception of the method is called on an empty list.

```
template<typename Object>
class LinkedList {

private:
   class Node {
     Object data;
     Node* next;
   };
   Node *head;

public:
   LinkedList() : head(nullptr) {}
   Object removeTail(Object data);
};
```

9. What are iterators? What purpose do they serve?

10. What does it mean to *invalidate* an iterator?

11. Explain the difference between separate chaining and open addressing in hash tables.

12. Define *load factor* and explain its relevance to hash table performance.

13. What are *collisions* with respect to hash tables?

14. Which hash tables distinguish between slots that have never been used, and slots that once contained an item but has now been deleted.

15. List and explain the worst-case and average-case running times for each HashTable method below

    (a) void insert(Key k, Value v)

    (b) bool contains(Key k)

    (c) Value get(Key k)