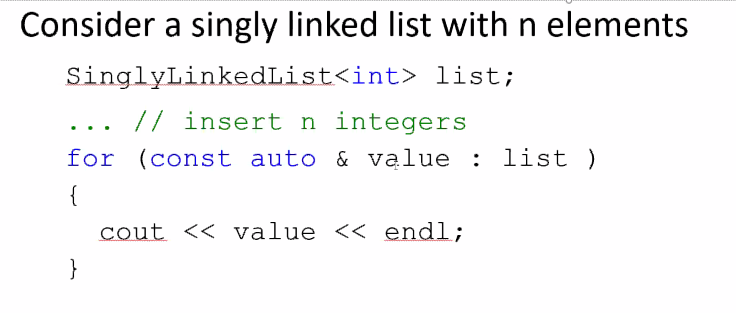
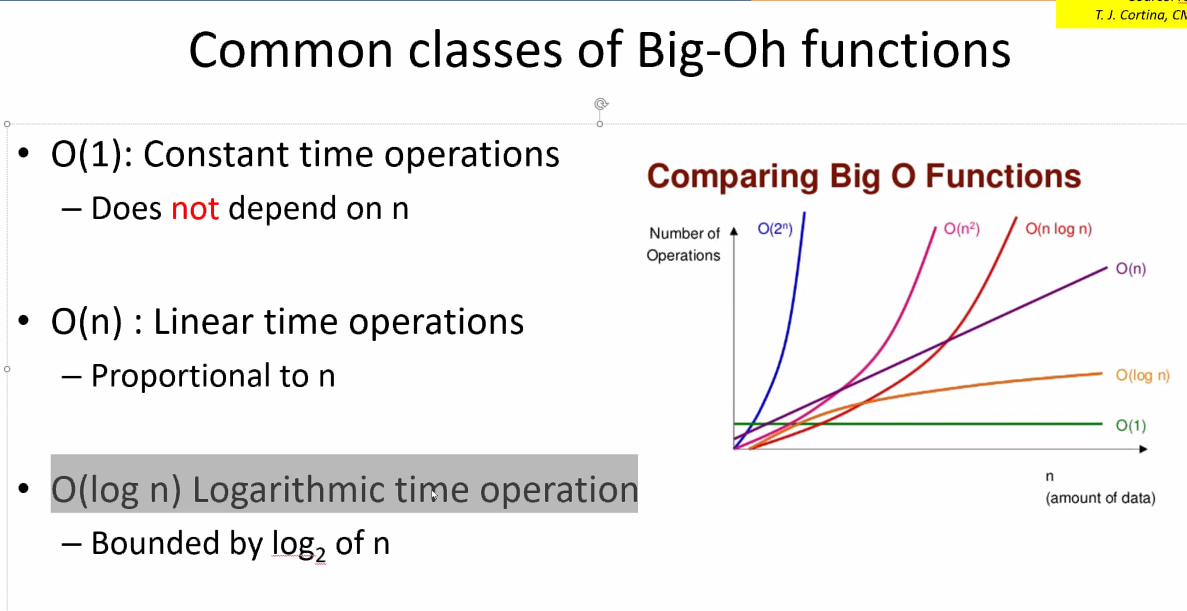
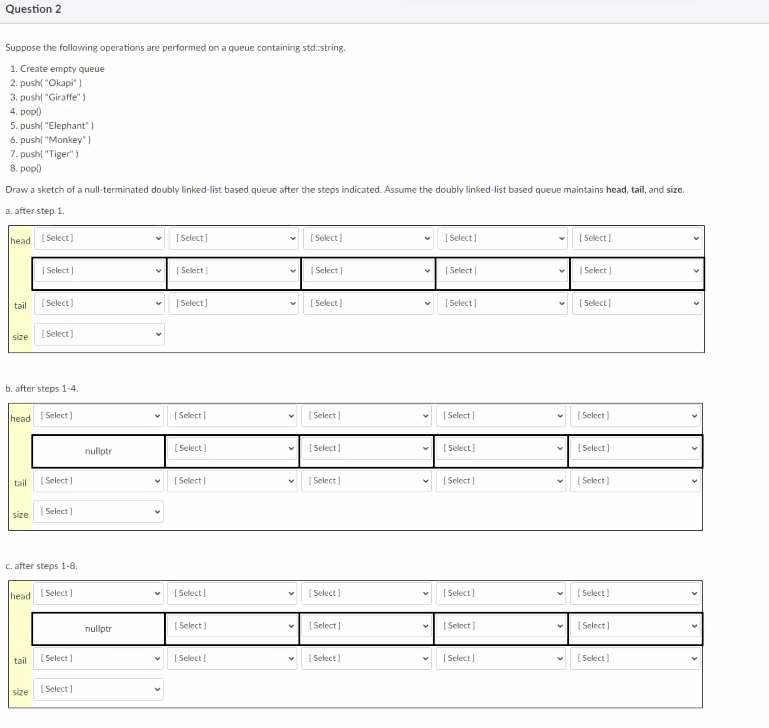
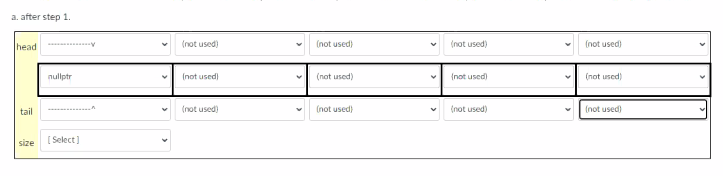
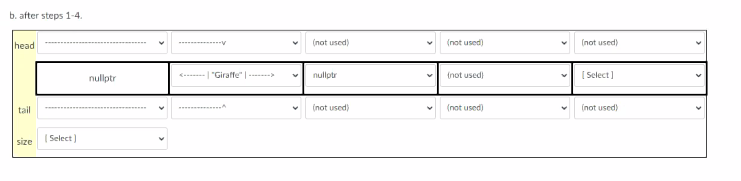
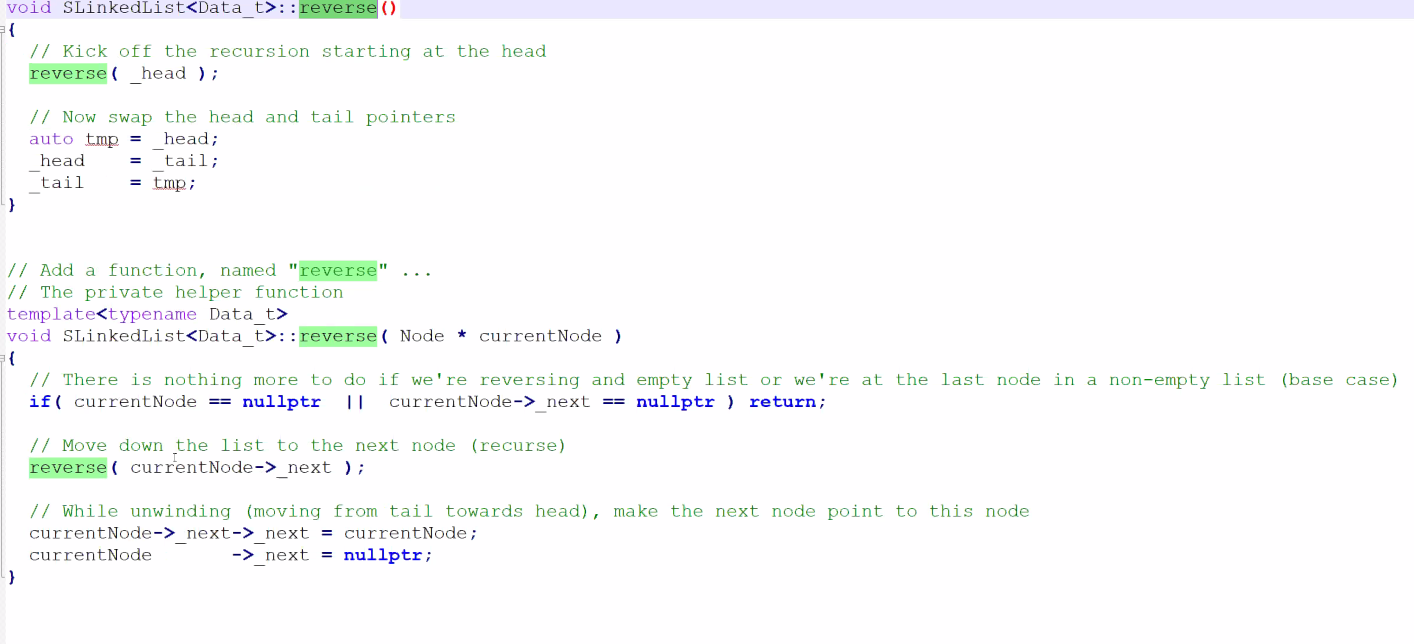
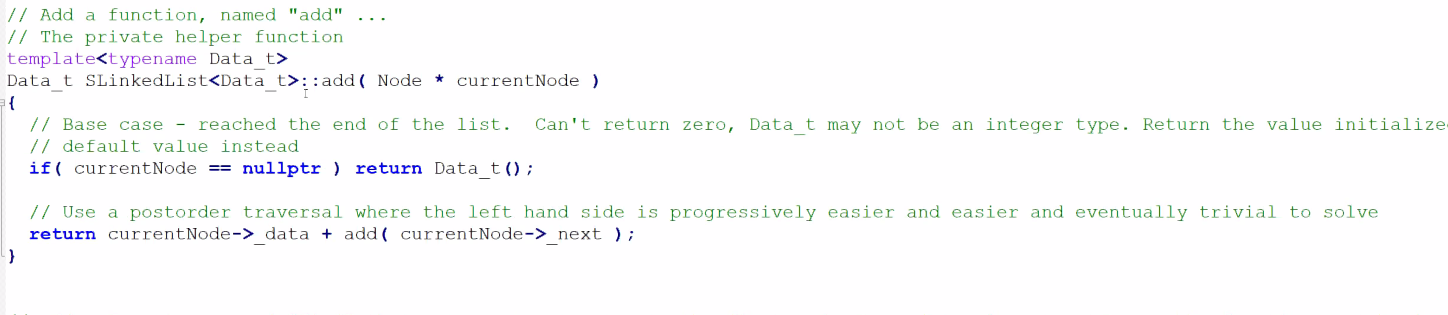
Lecture 14

CPSC 131  
10/19/2020

1. Leniencies
   1. You can open a text editor to write your code.
   2. You can also use a photo editor/paint to draw your sketches
   3. BUT YOU NEED TO BE ON ZOOM
   4. Hold ID up on the camera
2. Structures
   1. There are four parts to this midterm so missing one will cost you 25% of the entire dman thing.
   2. There are 4 sketching questions and each one will want you to load an image.
   3. Due at 6:45 PM
3. Content
   1. BIG O
      1. When we look at Big O, look for the looks.
         1. We look for the complexity of executing the set of steps.
         2. If the set of steps we are looking at have nothing to do with the container = O(1)
         3. Example  
              
            This is O(n)
      2. Common classes  
         
      3. O(log n) = lots of partitioning
      4. Amortization
         1. Example would be O(1) amortized which means that a large majority of time is constant despite a few hiccups now and then
         2. Incur amortization only to save memory and time
         3. Double size and double capacity
   2. Vector
      1. The elements, in emmoeyr, are right next to each other. The difference between arrays and vectors, the vector give you a concept of size. For example, an vector could be empty but array never can be (constant capacity and size)
      2. Difference between extendable and fix vector
   3. Stack vs Heap
      1. Stack = holds your local variables, parameters, return address
      2. Heap = dynamic memory
      3. Allocate memory on the heap use new
   4. Queues  
        
        
        
      
   5. Coding
      1. Recursion will be covered
         1. Remember to include:  
            public functions  
            private recursive functions  
            Data\_t add(Node \* currentNode);
      2. Three parts
         1. Base Case
         2. Recursive step (a series of problem solving)
         3. Conclusion
      3. Reverse  
         
      4. Add function  
         
      5. Find Function  
         