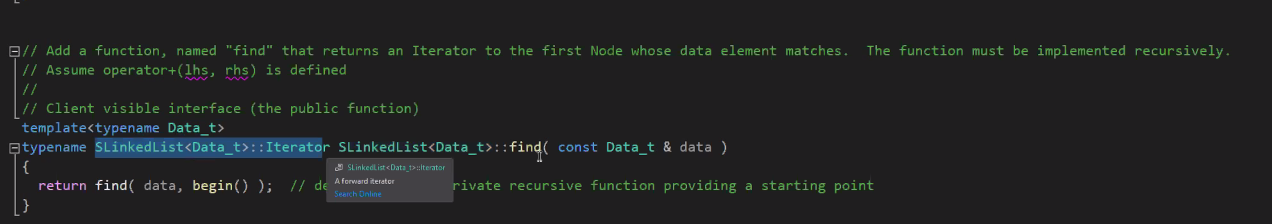
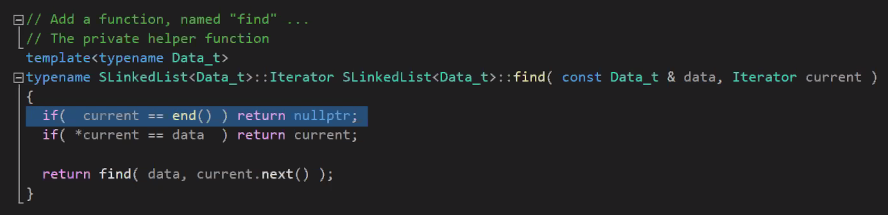
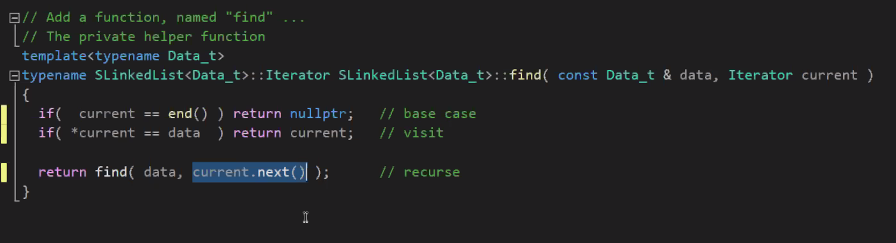
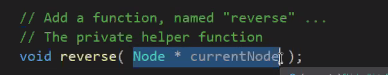
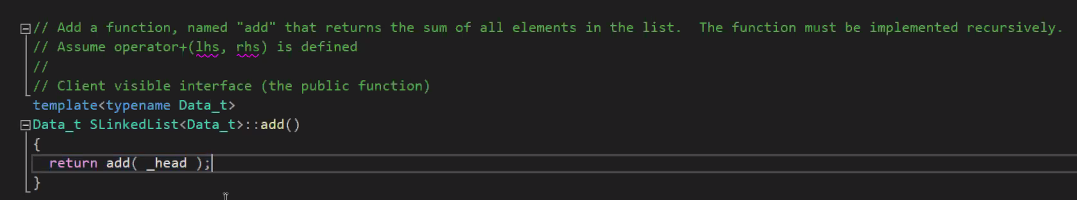
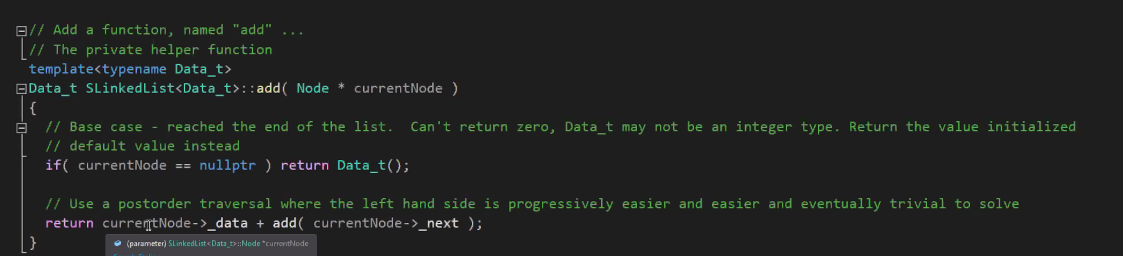
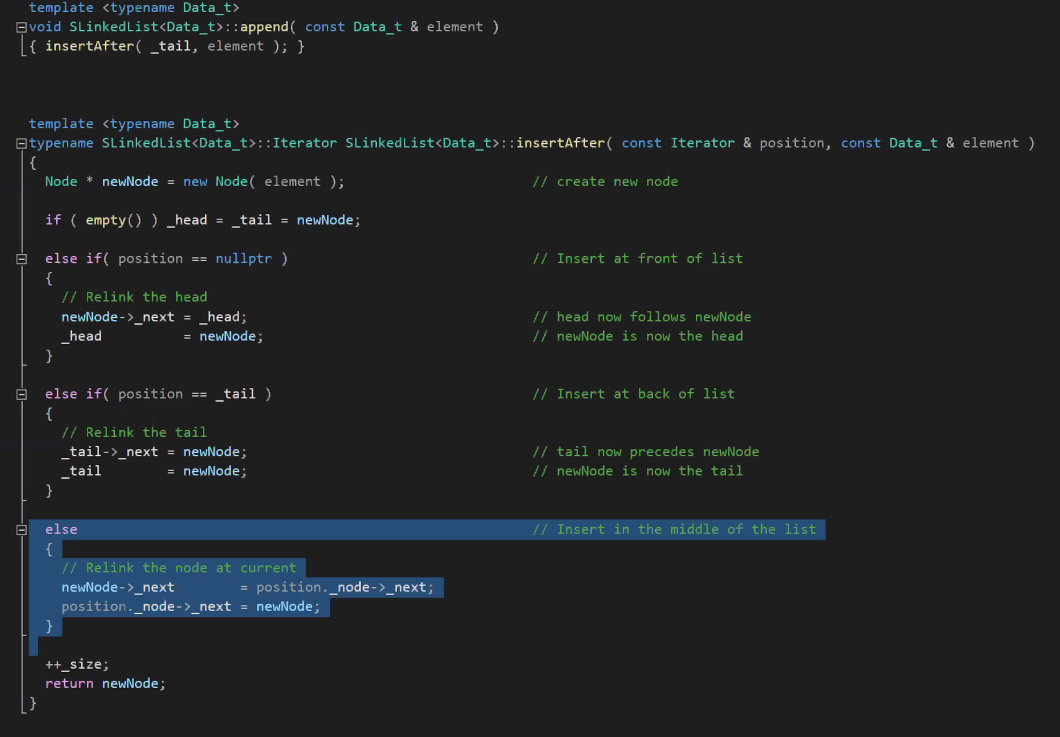
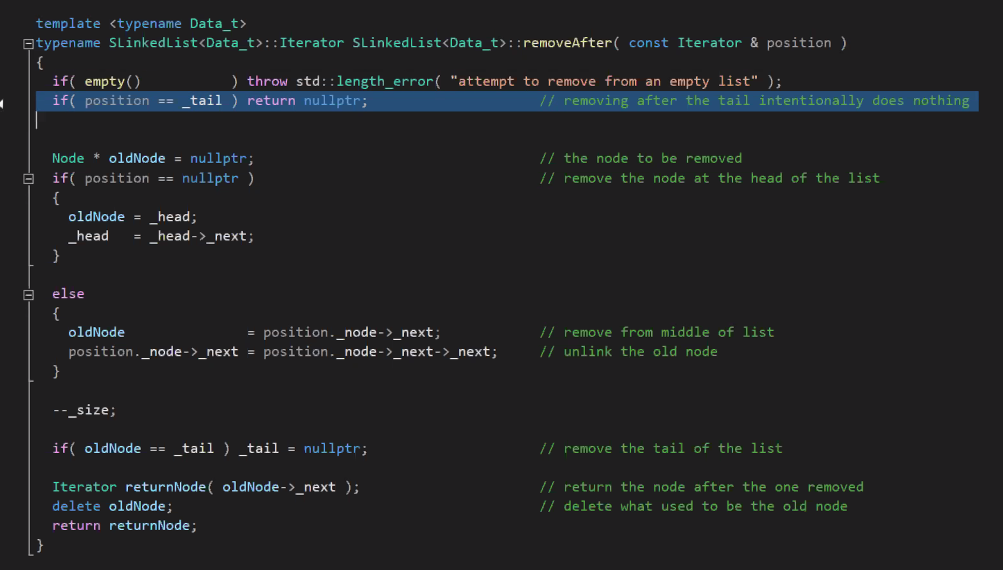
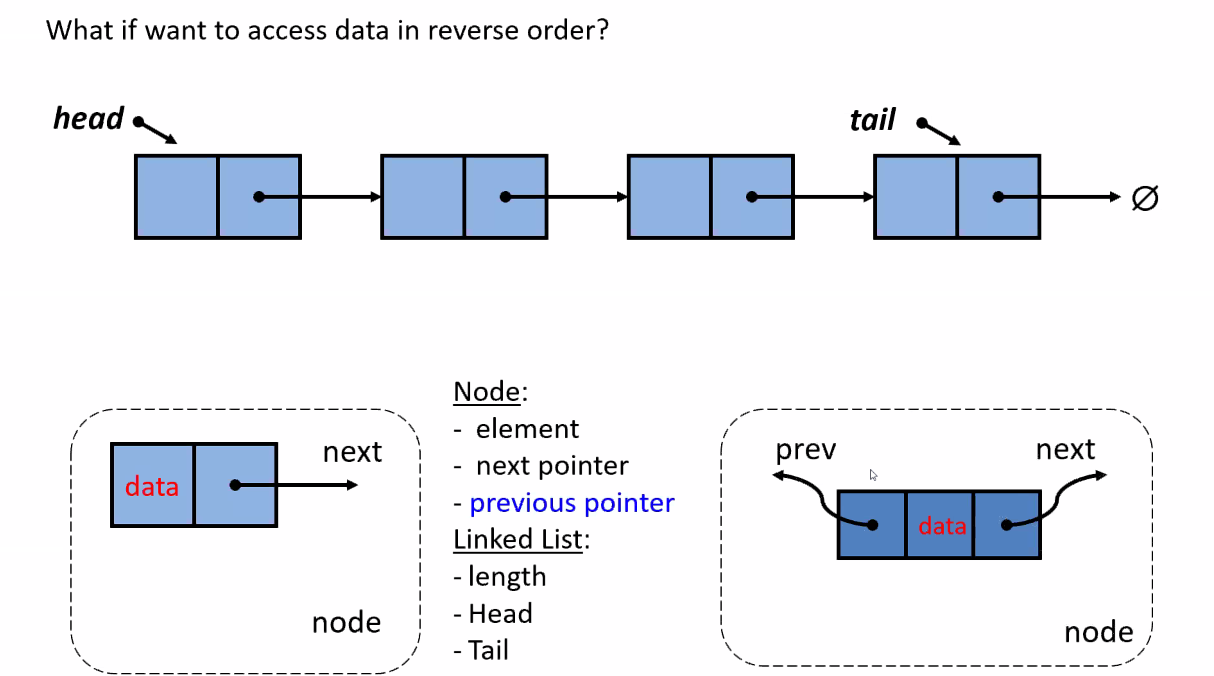
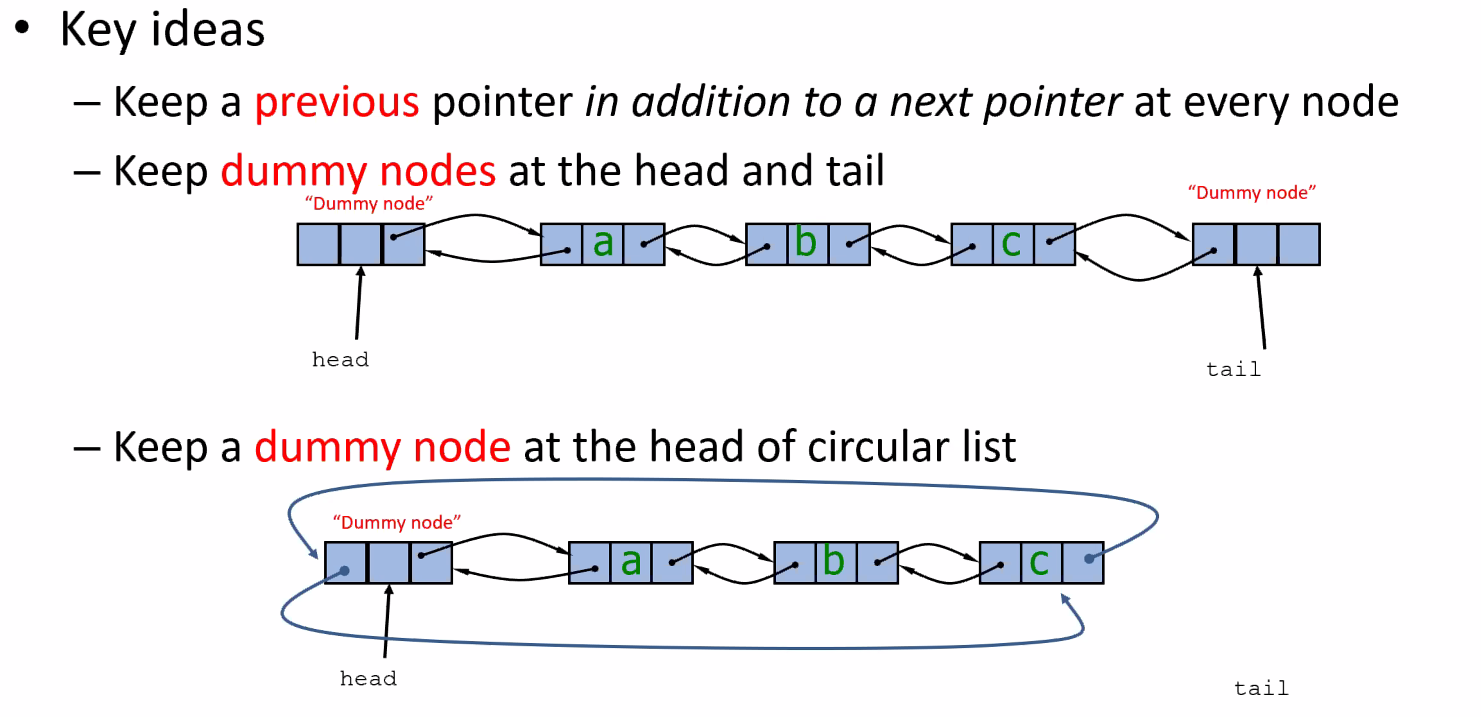
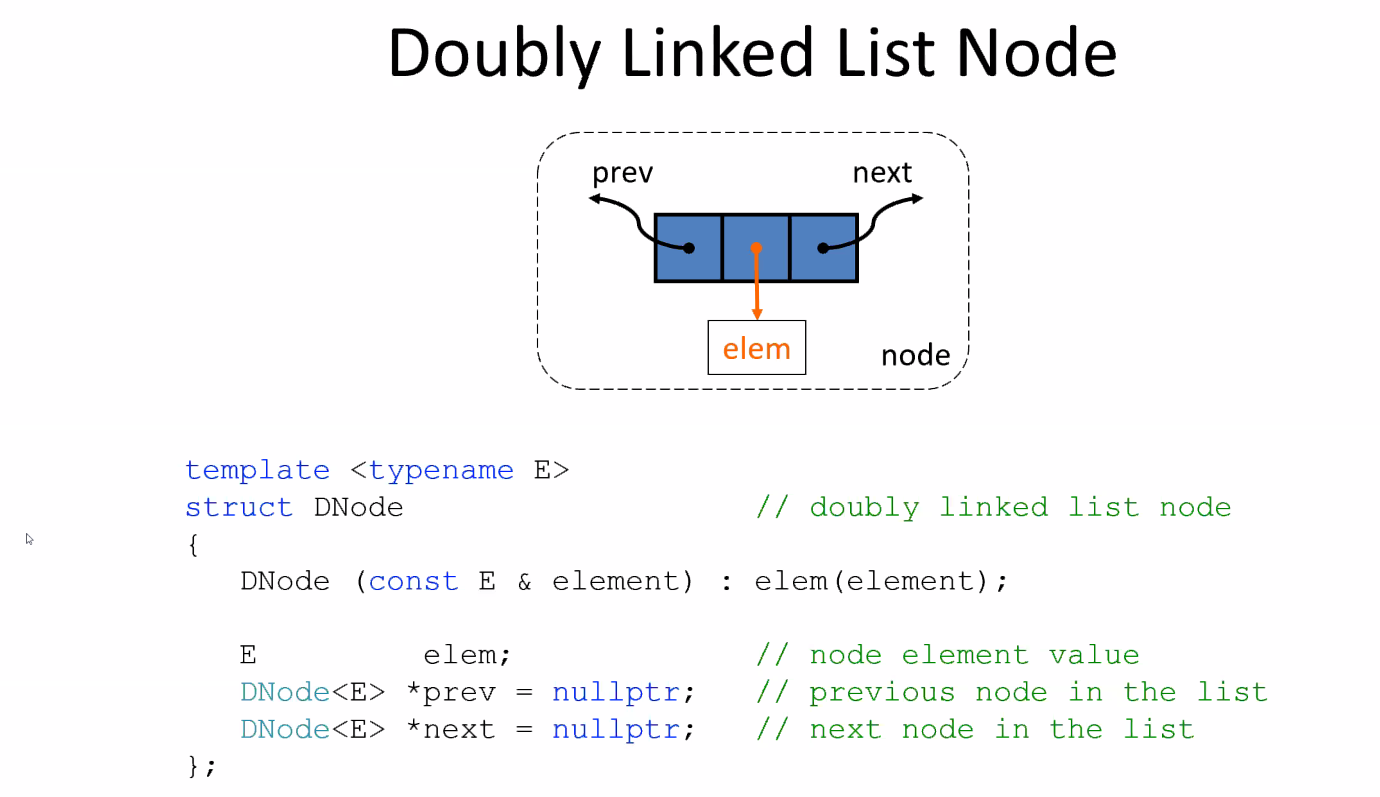
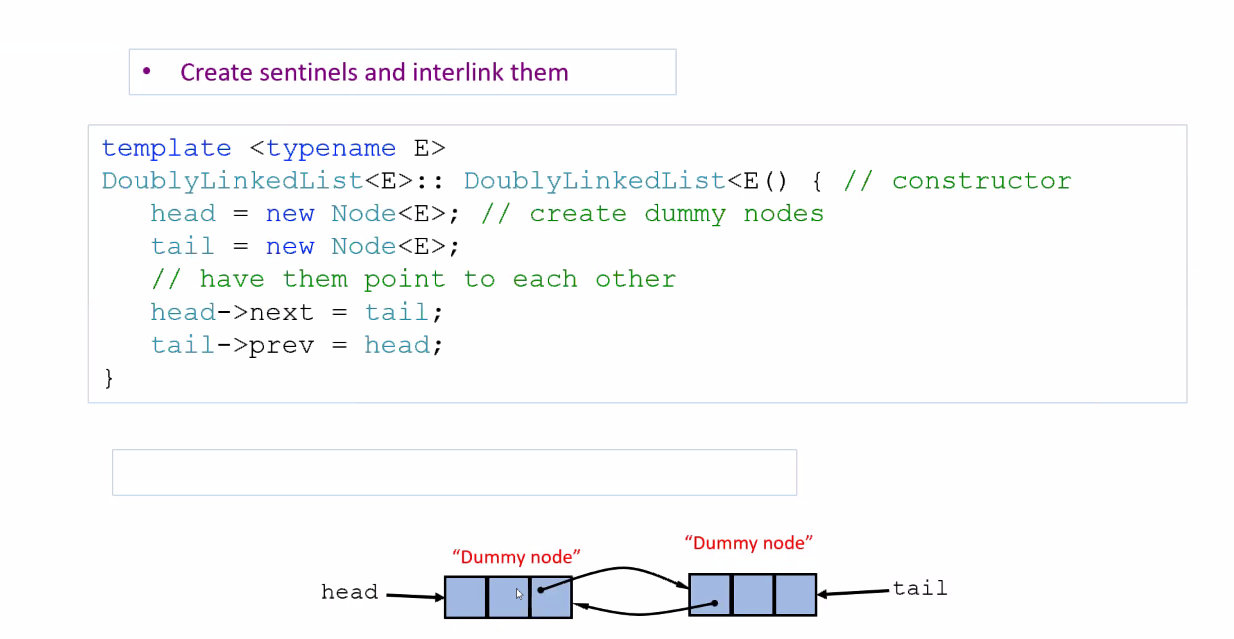
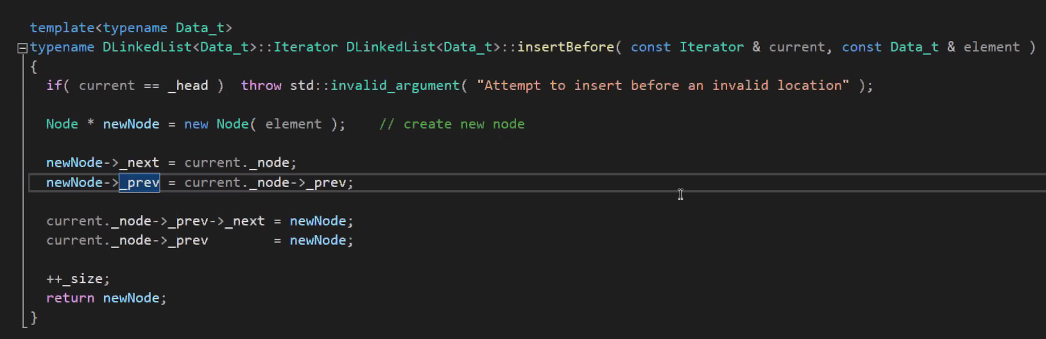
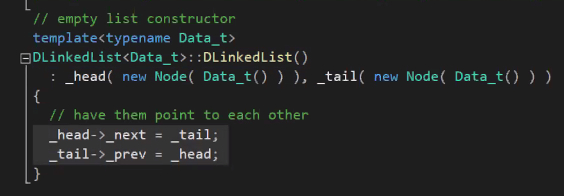
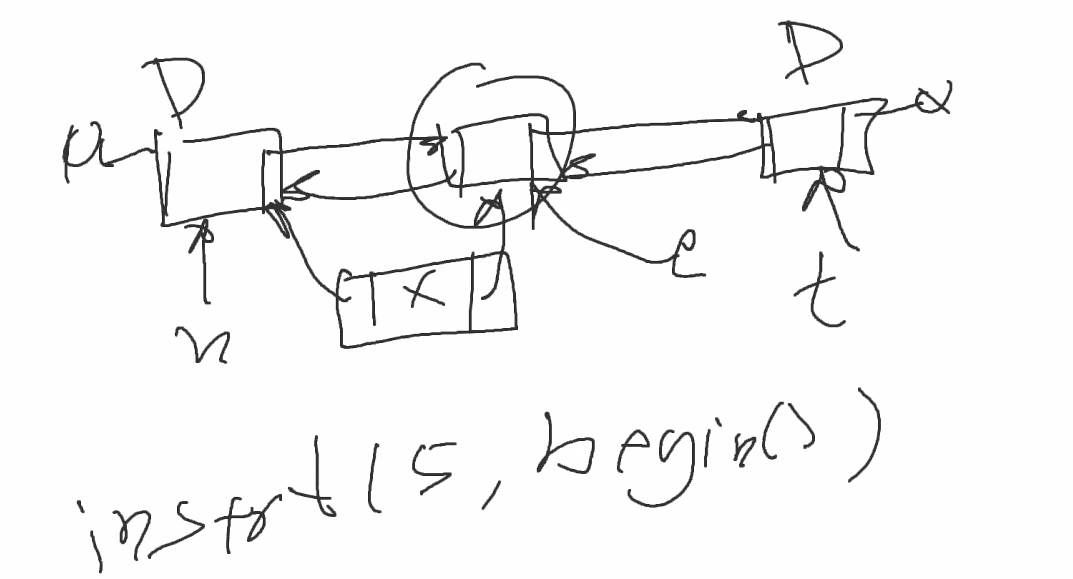
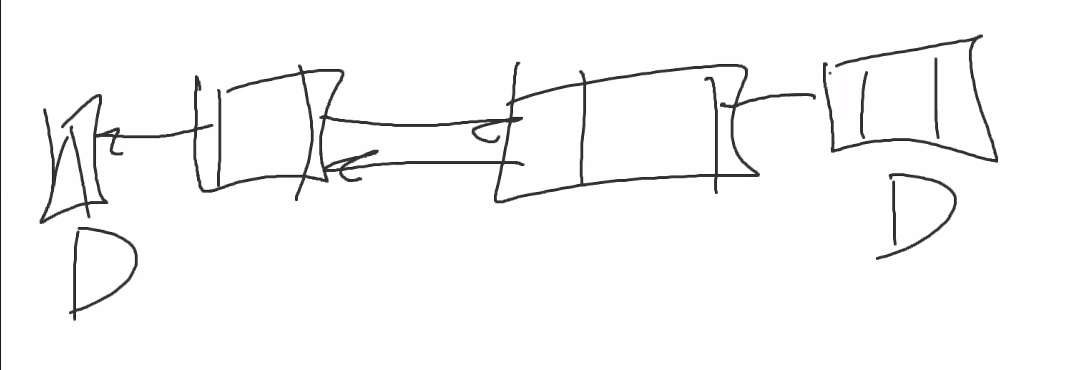
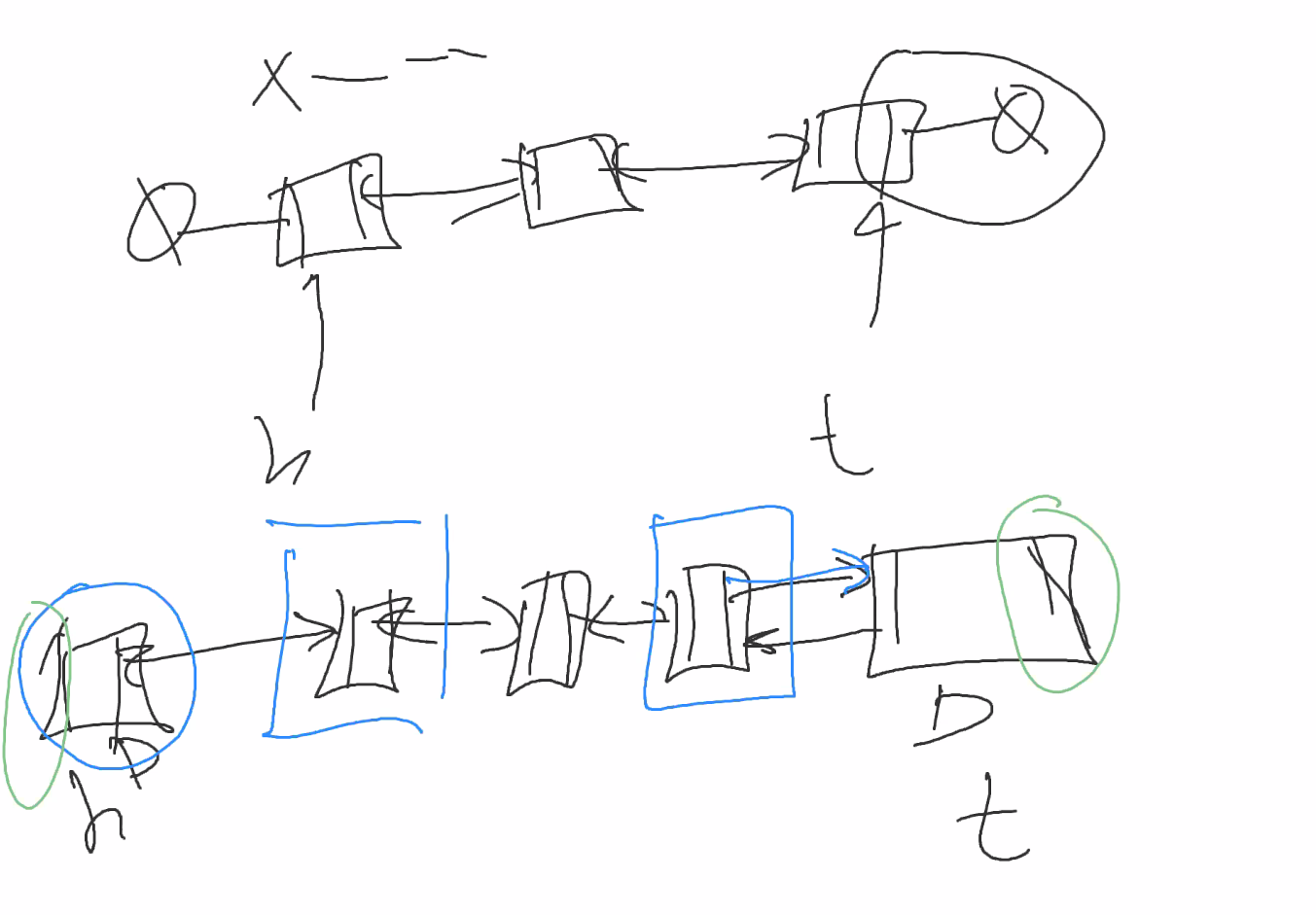
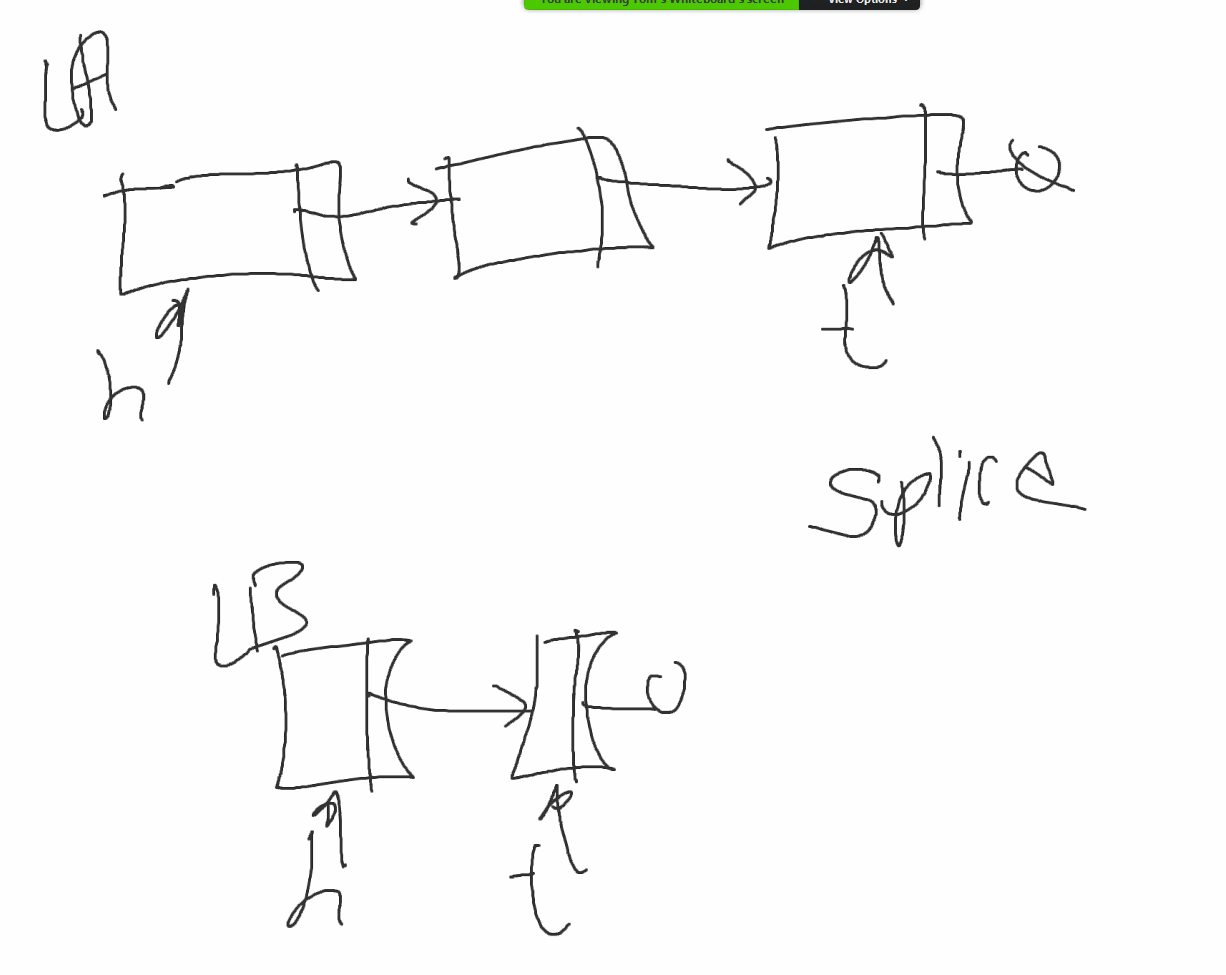
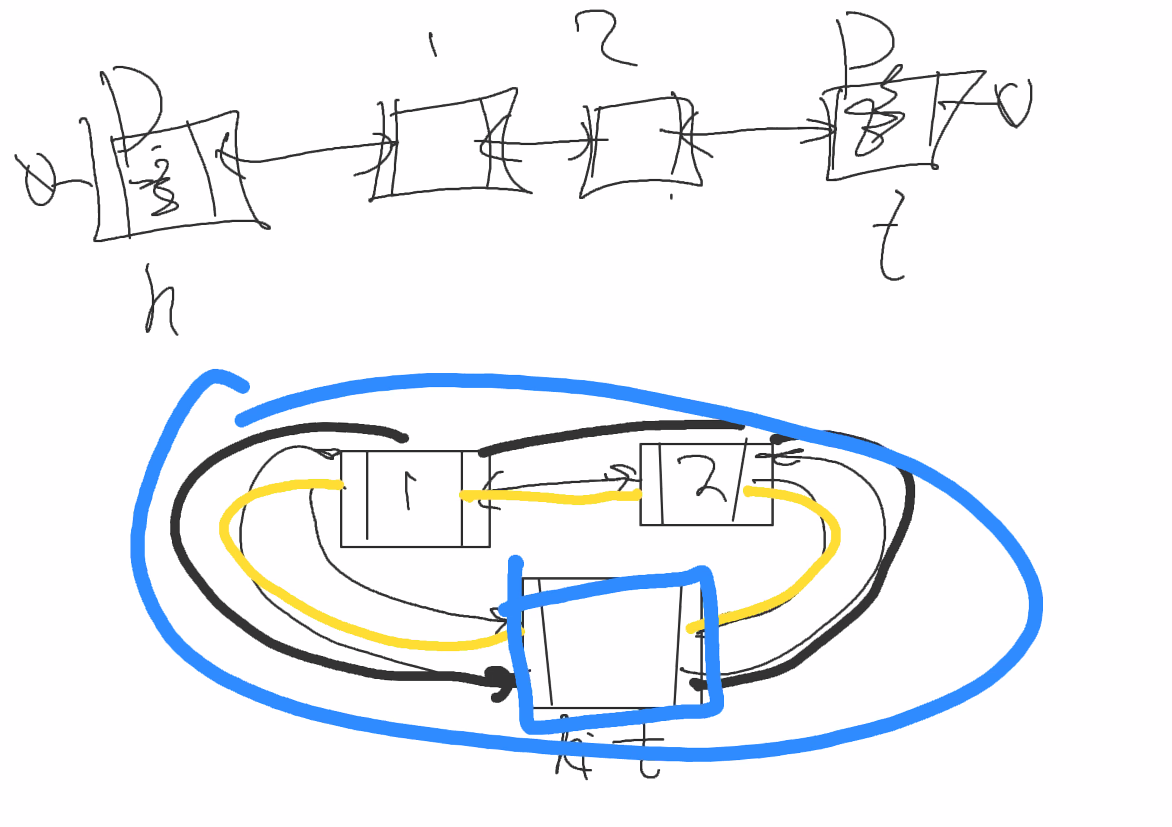
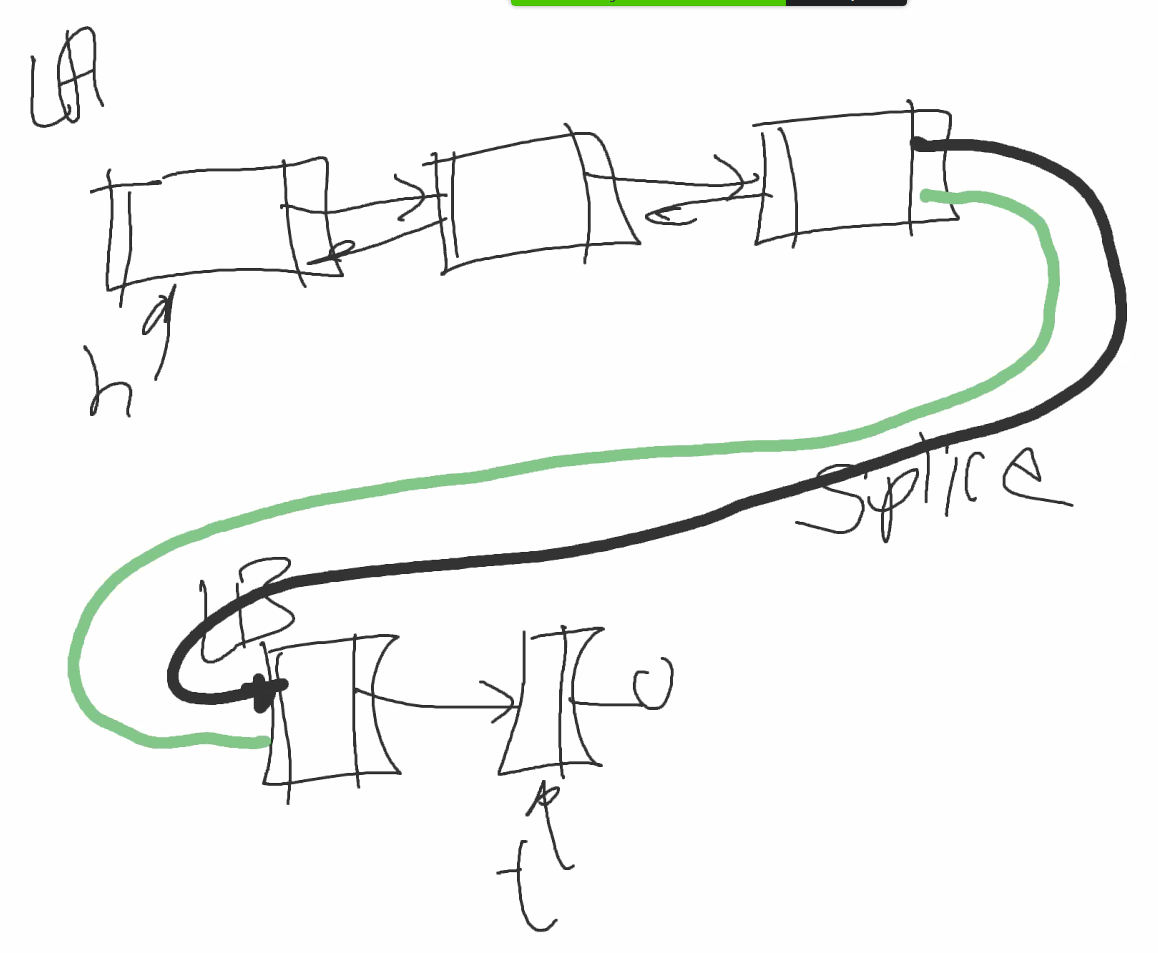
Lecture 9

9/23/2020

CPSC 131

1. Singly-Linked list
   1. Find()
      1. We look for an element of a particular value we are interested in.
      2. We walk the list until we find the node that matches what we’re looking for
      3. But if hwe don’t, we return a nullptr
   2. Iterator find(const Data\_t & data);
      1. The purpose of this function is to kick off the recursion
      2. We expose the public function, get the data
      3. 
      4. We want to start at the beginning/head of the list
      5. To stop recursing in a list  
         
      6. 
      7. So basically, stop when you’re empty or when you’ve found what we’ve been looking for
   3. Reverse()
      1. How to reverse without altering any data?
      2. 
      3. As discussed earlier, this particular functions only purpose is to recursion  
         
      4. This one returns nullptr  
         
      5. Insertafter LinkedLIst  
         
      6. Removeafter  
         
2. Lecture
   1. Singlylinked and doubly linekd list: Reverse Order
      1. 
      2. So in the doubly-linked list node, the node needs room for two pointers.
      3. We are ignoring the size of the data atm
      4. Doubly-linked node bigger than singly-linked
      5. Interface is very similar
   2. Doubly linked lsits
      1. Key ideas  
         
         1. First one: line, it ends for good
         2. Second one: circular, the tail points back to the head
         3. Dummy node: the heads next pointer is the head of the data, the useful data is between these two.
   3. Doubly Linked list Node
      1. 
      2. Constructor
         1. Sketching it as an (almost) empty  
            
            1. The sentinels are the dummies
            2. IT has only two dummy nodes
   4. Interface: Differences between Singly and Doubly
      1. Singly
         1. What we can do this with is giving the value at the front or the back
         2. In a singly linked, only the front because the standard template does not maintain the tail ptr
            1. How to insert something at the head: before\_begin
         3. Doubly
            1. We can easily do so at the front
   5. Referring to the Sequence Container Exmaple Codes
      1. Prepend(): Insert at front of list
      2. Append(): Insert at end
      3. But once we started looking deeper into the code, we strted to focus on fuctions related to inserting
         1. For instance, insertBefore() which, once we create a new node, enables us to insert a node before the tail.  
            
      4. To enable a tail to point to the head,   
         
         1. It is circular
      5. How to start at tail: rbegin()
         1. How to start at end: rend()
         2. R = reverse
   6. Visualizing insert()
      1. 
      2. To break the chain
         1. We take the next ptr not from the next but from the new node [x] and then we take the previous pointer and set it to x instead of e
         2. We broke the chain and linked the new ptr at the same time
      3. So we get a new doubly linked list  
         
         1. Sort of, it’s the same list but with a new element added in
   7. 
      1. With splice  
         
   8. Do we need two dummy nodes?  
      Short answer: yes  
      Long answer:  
      
      1. This enables us to find the beginning and end of a doubly linked list
      2. Blue: Circular Doubly Linked
      3. Yellow and black: Singular Doubly linked
      4. Splice doubly-linked list  
         
3. Quiz
   1. T/F, multiple choice
   2. There will be code in the form of fill-in-the-blank
      1. Circular Doubly linked list
   3. Look at the find() in our code
   4. Draw data structures
   5. Yes, recursive functions will be on it
   6. Two attempts taken
   7. Shouldn’t take you more than 20/30 max minutes