

Quiz #1 Answers

1. Assume that you have a doubly linked list class with a node class inside it. This class uses a sentinel to help simplify the code so it has one special node named end. Draw a picture of this, then write code for adding to the head of the list. (Language doesn't matter here. Pseudocode will work.)

I'm skipping the picture because it's a pain in Word, but you have a doubly linked, circular list. The sentinel has the head element as next and the tail element as prev. To add to the head you create a new node and link it in between the sentinel and what lies on the next side of it.

Code:

```
AddToHead(Data d) {
    Node newNode=Node;
    newNode.data=d;
    newNode.prev=sentinel;
    newNode.next=sentinel.next;
    sentinel.next.prev=newNode;
    sentinel.next=newNode;
}
```

2. The C++ programming language does not have automatic garbage collection. List a few ways that this impacts your programming and the language features that exist to deal with this.

This first significant impact of no garbage collection is that you have to explicitly call delete every time you call new. If you fail to do this you create a memory leak that will eventually take down your program. To help make the delete operator work with objects they also added destructors so that when delete is called on an object you can specify other things that should happen. Normally this means that any pointers in the object will have delete called on them. That is required for memory management, but it actually produces different problems because now it is difficult for objects to share other objects. If an object is immutable you should be able to safely share one copy of it across many objects. However, you can't have all those objects deleting it. As a result, you typically must write copy constructors and overload operator= so that they do deep copies.

Extra Credit: Describe what a hash table is and what it is good for.

A hash table is basically a vector, but you don't look up values in it straight off the key because the key values are very sparse and can be very large. Instead, you pass the key through a hash function, which maps it into the range of the size of the vector. This gives you O(1) insert, search, and delete operations if the size of the hash is proportional to the number of elements that will be stored in it.