Topics on Introductory Data Structures and Algorithms

Ilkka Kokkarinen, TMU

One: Best Kind of Linked Lists

Linked List Variations

- Singly linked vs. Doubly linked
- Linear vs. Cyclic
- Separate `List` object vs. The first `Node` object simultaneously represents the entire list
- Head pointer vs. Header node
- Sixteen possible combinations, only two relevant

Billion Dollar Mistake of Software Engineering

- Null pointers are highly problematic
- Must constantly check that the pointer is not null before dereferencing that pointer
- Tony Hoare's "billion dollar mistake" in 1965
- Modern languages help with non-null constraints enforced at compile time

Null Object Pattern

- Design pattern to help mitigate problems with null pointers
- In a structure of nodes that point to each other, define an otherwise unused node object called `NULL`
- Whenever in code you would make a pointer to be null, assign that pointer to point to NULL instead
- NULL is an object with fields, so dereferencing a pointer with that value does not result in a runtime crash

Null Pointers In Linked Lists

- Usually textbooks advocate having the `List` object contain a head pointer to the first node object
- Many list algorithms become needlessly painful
- For example, remove nodes with given key
- Special case of removing from empty list
- Special case of removing the first node
- General case of removing later nodes

In a Cyclic Doubly Linked List, Always Use Header Node!

- Instead of `head` pointer, use a separate header `Node`
- Don't even think about this, but just do it! Do it now!
- Header node carries no payload
- If the entire list has *n* elements, it has *n*+1 nodes
- In a doubly linked cyclic list, this header node gives quick access to the first and last element
- In an empty list, header node is its own successor and predecessor

No Edge Case Situations Occur With Header Nodes

- A cyclic doubly linked list with header node does not contain any null pointers anywhere ever!
- Basically all algorithms for such lists are massively simpler compared to same algorithms for other list variations
- For example, removing any node is handled with two trivial assignments, regardless of the position of that node
- (The header node itself is never removed from the list)