Lab 1: Link List Implementation
Atiya Mirza
CSC 332
9/8/22

Link List vs. Circular Link List

A linked list is a linear data structure made from a series of connected nodes in which each node stores both a data item and an address item for the next node. It essentially forms a sequential chain-like structure with a beginning node and an end node. In singly-linked lists, the last node does not point to any node so its address part contains a NULL value. This means that each node has a link to the next node in linked lists, excluding the last node. Further, only forward traversal is possible in singly-linked lists as each node only has one link. Singly-linked lists are a fundamental and commonly used data structure. For example, they are used to implement stacks and queues, as well as to prevent collision in hash maps. These types of lists are also used to create more advanced data structures like trees and graphs. Some advantages of singly-linked lists are that they are relatively easy to implement and reduce access time. A major disadvantage is that it cannot access nodes randomly since it works sequentially, and so reverse traversal cannot be done on these lists either.

A circular linked list falls under the subset of linked lists, with the biggest difference being that the nodes are connected in a circular cycle-like structure instead. In other words, the last node connects to the first node, so the address part of the last node holds the address of the first node rather than a NULL value. This type of list is also a sequence of nodes (or elements) in which each node is linked to the next node in the sequence, but the last node is linked to the first node as well. Circular linked lists basically have no beginning or end node then, which makes them useful because they can be traversed from any node and in any direction. Furthermore, they can save computing time when having to travel from the last node to the first node, and they can be used to easily reference the previous node which is not possible for singly-linked lists. There are some disadvantages to circular linked lists too, such as the fact that their operations are more complex than singly-linked lists. Also, traversing has to be performed carefully to prevent ending up in an infinite loop since there is no NULL value to stop the traversal. Circular linked lists can be useful in many cases where a regular linked list may not be. For example, if multiple applications are running on an operating system, they can be placed in a circular linked list so the OS will keep on iterating over the applications.