Linked lists are an ordered collection of data elements. Each element is usually referred to as a node in the Linked List. Each of these nodes contain two features the first being the data of the node and the other being a pointer. The pointer is what “links” our nodes together essentially. Linked Lists are similar to arrays; however, they are not stored in memory in sequence next to each other. They can be stored almost anywhere in memory.

With a linked list we have certain properties that include:

our successive nodes which are linked by a pointer,

a head node

the last node points to null

are dynamically sized

Another great benefit in comparison to an array is that with an array the size must be declared ahead of time, but a Linked List is able to add and remove elements making them more dynamic. The downfall however can be that, due to its random storage in memory, we do not cannot access an element at any point or index like we can in an array. In addition, they often use more total memory all things being equal due to the storage of the pointer.

There are three major types of Linked Lists:

Singly Linked Lists

Doubly Linked Lists

Circular Linked Lists

Singly Linked Lists are the most common and have been what I have referred to in this post so far. They contain two parts, the data, and the pointer. They work in one (single) direction and end at the last node that contains the null pointer.

Double Linked Lists

These are similar however they contain three parts, the data and two pointers. One points to the next node and one to the previous node if it exists.

Lastly the Circular Linked List is almost identical to the Singular Linked List, the difference being that it does not contain the last node of a null pointer. Instead, it points back to the first node. In these we can traverse them in any direction