What is a stack?

1 Key Items

Stacks are based on the concept of linked lists. This means that pointers are key for the operations of a stack. As they are based on linked lists, stacks are comprised of multiple nodes, each of which contains the data for the node as well as a pointer to the next node. The 'top' node of a stack is often times refered to as the 'head'. This is the only node that is easily accessible.

1.1 Data

There is no specific format that is needed for stacks. In fact, multiple pieces of data, such as a point on an x-y plane, can easily be stored within a stack. In the code sample below, we are simply storing an integer within the node, but this is adaptable to whatever data type is needed, including custom data types.

1.2 Node Pointer

This is simply a pointer to another node. There is no special syntax to declaring a pointer as is evident in the code below. Quick review of pointers, a pointer is a variable whose value refers to another memory location within the computer memory.

Figure 1: Code for a node structure

```
struct Node{
    int data;
    struct Node * next;
}node_t;
```

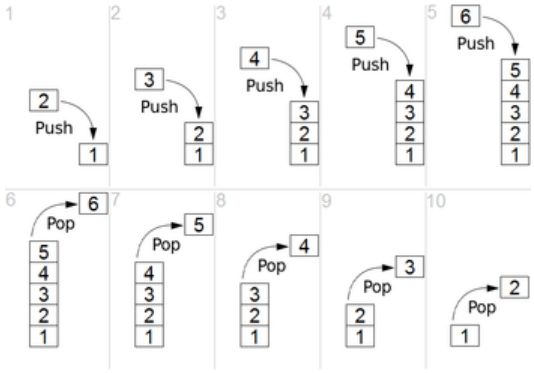
2 Operations

The main operations that are needed for stacks are called push (add) and pop (remove). There operations will usually only impact the first element of the stack. The diagram below is of these operations on a stack.

2.1 Push

This is the name for adding an element to the stack. To do this, we first set up a temporary node that will hold the new value. We then set the value of 'next' to point to 'head'. For this operation, we do not have to see if 'head' is NULL (not declared).

Figure 2: Diagram of Push and Pop operations. Retrievied from:*https* : *//upload.wikimedia.org/wikipedia/commons/thumb/b/b4/Lifo*$_s$*tack.png/*350*px*−*Lifo*$_s$*tack.png*



2.2 Pop

This is the name of the act of removing an element from a stack. Our first step is to see if 'head' is NULL. If it is, return an invalid data value. If not, we set up a temporary node and set it equal to 'head'. We set a 'value' variable equal to the data stored in 'head. Our next step, is to set 'head' to the next element in the linked list and free the temporary variable. Lastly, we return the 'value' variable.