```cpp
// Justin Dang Student ID: 1148267
/*
Creates an array based stack using push and pop methods

Array is limited to a max size of 10

when going above capacity or trying to remove nothing, an error is thrown
*/

#include <iostream>
using namespace std;

class Node {
public:
    int data;                           // data in each node
    class Node* next;                   // address of next node(or null/0 to define as end of Queue)
    Node(int info, Node* ptr = 0) { // Structure for each node
        data = info;
        next = ptr;
    }
};

class Stack {
private:
    Node* topOfStack;     // since this is a stack, this will be the only var we can interact with
    int stackSize;
public:
    Stack() { topOfStack = 0; }                   // creates empty stack
    bool isEmpty() { return topOfStack == 0; } // returns true if stack is empty
    void push(int info) {
        Node* temp = new Node(info);            // new node created
        if (isEmpty())
            topOfStack = temp;                  // if the stack is empty we set the topOfStack directly to new node
        else {
            temp->next = topOfStack;            // otherwise we grab the address of the new node and set to our top Node
            topOfStack = temp;                  // we then set the top to the new node
        }
        stackSize++;
    }
    int pop() {
        if (isEmpty()) {                        // throws error when attempting to remove nothing
            cout << "Cannot Pop() null.\n\n";
            stackSize++;                        // Offset decrement
            return -999;
        }
        Node* temp;                             // temp Node used to hold top var
```

```cpp
          and later deleted
        int returnInt = topOfStack->data;      // Store the int of the top Node data
        temp = topOfStack;                     // set temp to top of stack to delete data of top node
        topOfStack = topOfStack->next;         // set top Node equal to the next address stored
        delete temp;                           // delete temp
        return returnInt;
    }
    void print() {
        cout << "Top: ";
        for (Node* temp = topOfStack; temp != 0; temp = temp->next)
            cout << temp->data << ' ';
        cout << "\n\n";
    }
};
int main()
{
    Stack* stack;
    stack = new Stack();
    cout << "---------------------------------\n";
    cout << "Working with an Link based stack.\n\n\n"
        << "Testing error when attempting to remove from an empty stack: \n\n";
    stack->pop();
    stack->print();
    cout << "---------------------------------\n";

    cout << "Push 30 onto stack: \n\n";
    stack->push(35);
    stack->print();

    cout << "Push 1 onto stack: \n\n";
    stack->push(1);
    stack->print();

    cout << "Push 5 onto stack: \n\n";
    stack->push(5);
    stack->print();

    cout << "Push 7 onto stack: \n\n";
    stack->push(7);
    stack->print();

    cout << "Push 12 onto stack: \n\n";
    stack->push(12);
    stack->print();
}

```