C++ Programming

- (80 points) Re-write the String class from homework 3, but use a singly-linked list as the representation. Like last week, these strings will be of varying lengths and must grow and shrink as necessary. Implement all the appropriate methods given below.
- Class String declaration:

```
class String
{
public:
  /// Both constructors should construct
  /// from the parameter s
  String( const char * s = "");
  String( const String & s );
  String operator = ( const String & s );
  char & operator [] ( const int index );
  int length() const;
  int indexOf( char c ) const;
  bool operator == ( const String & s ) const;
  /// concatenates this and s
  String operator + ( const String & s ) const;
  /// concatenates s onto end of this
  String operator += ( const String & s );
  void print( ostream & out );
  void read( istream & in );
  ~String();
private:
  bool inBounds( int i )
  {
    return i >= 0 && i < length();
  }
  struct ListNode
  {
    char info;
    ListNode * next;
    ListNode(char newInfo, ListNode * newNext)
      : info( newInfo ), next( newNext )
    {
    }
  };
  ListNode * head; // no other data members!!
};
ostream & operator << ( ostream & out, String str );
istream & operator >> ( istream & in, String & str );
```

- (10 points) Write a main function which tests each function defined in your class String. You may use the same one from last week if you like, but notice I deleted some of the methods.

- (10 points) Give an estimate of the relative efficiency of each of the following two assignments (I mean how many function calls, how many copies are made, etc):

  ```
  String s("Hello");
  String t("There");
  s = s + t;
  s += t;
  ```