Data Structures and Fundamentals of Programming

Problem #1
In C++, implement a string abstract data type using a dynamically allocated array. The array of char should be null terminating. You must implement the following methods:

- Default constructor (allocates 128 characters as capacity)
- Constructor that takes an integer value as the capacity to allocate
- Copy constructor
- Destructor
- Reallocate that changes the capacity of a string (larger or smaller) and preserves as much of the contents of the string as possible.
- Assignment – overload the operator= with standard C++ semantics.
- Constant time swap – swaps two strings in constant time regardless of their capacity
- Length
- Capacity

You can NOT use STL or any other predefined library, built in types (such as std::string), or C functions (such as strcpy).

Problem #2
In C++ implement a binary search tree abstract data type that uses a pointer (dynamically allocated) implementation of the tree structure. Make it a simple tree of integers. Along with the class definition(s), you must implement the following methods for the binary search tree class:

- Default constructor
- Destructor
- Copy-constructor
- insert which takes a parameter of type integer and creates a new node that is added to the tree in the correct position based on the rules of a binary search tree.

Note: Your implementation can NOT use STL or any other libraries (standard or otherwise).

Problem #3
Given a string ADT as defined below, implement (in C++) a method that concatenates two strings and returns the result. You must use the method specification given below. Do all error checking. If you need additional methods or functions to solve this problem they must also be implemented.

```cpp
// CLASS INV: s[length()] == 0
class String {
public:
    String() {s[0] = 0;};
    String operator+(const String&) const;

private:
    char s[256]; //null terminated fixed sized character array
};
```

Note: Your implementation can NOT use STL, any other libraries, or any built in functions.