Mid-term Review

# Practical Exercise

This exercise is quite large, but completion of this exercise will provide you with a complete, practical review of the material you are expected to know for the exam. This exercise covers the all of practical applications from weeks 1 - 6

## String class

Up to this point, to store a string we have used C-style string. C-style strings are an array of characters, with the string being terminated by the null character ‘\0’. The C-style string can be unsafe, and annoying to work with. We are going to create a class called String to make the handling of strings simpler.

#### Part 1 – Class definition

In a header file, Inside of the “sict” namespace create the definition for a class called “String”. It will contain the following private data members:

* Character pointer named “\_characters”. This will hold the characters of our string object. It is a c-style string, terminated by ‘\0’.
* An integer named \_length. This will hold the length of our “\_character” array.

The String class will have the following public member functions:

* “to\_cstr” It returns a constant character pointer and accepts no arguments.
* “length” It returns an integer and accepts no arguments.
* “print” It is a void function that accepts no arguments.
* An overloaded assignment operator. It returns a reference to a String object and accepts a constant character pointer.
* An overloaded += operator. It returns a reference to a String object and accepts a constant character pointer.
* Default constructor
* Copy constructor
* Single argument constructor that takes a constant character pointer.
* Destructor.

#### Part 2 Class implementation

Inside of an implementation file implement the following functions in the following manner:

1. Default Constructor. Set all data members to a safe empty state.
2. Single argument constructor. Check if the character pointer passed in is a null pointer.
   1. If it is not a null pointer, set the length of the current object to the length of the character array that was passed in. Do not forget to add one for the null character as strlen() does not count the null character. Dynamically create an array and store the pointer to that array in ”\_characters”. Copy the character array that was passed in into the current objects \_character array.
   2. If the character pointer passed in is a null pointer, set the object to a safe, empty state.
3. Copy constructor. Check that the passed in Strings character array is a null pointer.
   1. If it is not a null pointer, set the length of the current object to the length of the String object that was passed in. Dynamically create an array and store the pointer to that array in \_”characters”. Copy the passed in String character array into the currents objects \_character array.
   2. If the character array of the String object is null pointer, set the object to a safe empty state.
4. Destructor. Check if the “\_character” array if a null pointer. If it is not a null pointer, deallocate the array.
5. “to\_cstr”. Return the character array;
6. “length”. Return the \_length;
7. “print”. Send the character array to stdout (use cout)
8. Overloaded Assignment operator. Set length to 0. Check to see if the \_characters array is a null pointer. If it is not a null pointer, deallocate the \_characters array. Check if the character array that was passed in is a null pointer. If it is not a null pointer, set the length of the current object to the length of the character array that was passed in. Do not forget to add one for the null character as strlen() does not count the null character. Dynamically create an array and store the pointer to that array in ”\_characters”. Copy the character array that was passed in into the current objects \_character array.
9. Overloaded += operator. Check if the character array that was passed in is null pointer.
   1. If it is not null, check if the \_characters array is null.
      1. If it is not null, create a temporary character array and allocate enough memory to hold the old array and the new array combined (\_length + strlen()). Copy the old string into the temporary array. Concatenate the string that was passed into the function onto the temporary array so that the old string and the new string have been combined. Deallocate the memory held at the \_characters array. Copy the address from the temporary array into the \_characters array. Make sure to update the length.
      2. If it is null dynamically create an array and store the pointer to that array in \_”characters”. Copy the passed in String character array into the currents objects \_character array. Update the length

#### Part 3 - Helper functions

We will also create some helper functions. Using what you know about the class, and from the instructions above, code the help functions below. Try not to duplicate code:

* An overloaded == operator. It returns a bool, and accepts two String references. It compares if both strings are the same, and returns true if they are.
* An overloaded + operator. It returns a String object, and accepts a String reference as the left operand, and a constant character pointer as the right operand. Takes the string on the right, and adds it on the end of the string on the left, and it returns the result.
* An overloaded + operator. It returns a String object, and accepts a constant character pointer as the left operand, and a constant String reference as the right operand. Takes the string on the right, and puts it on the end of the string on the left, and it returns the result.
* An overloaded + operator. It returns a String object, and accepts two constant string references. Takes the string on the right, and puts it on the end of the string on the left, and it returns the result.

## Extras

* An overloaded assignment operator. It returns a reference to a String object and accepts a constant String reference.
* An overloaded += operator. It returns a reference to a String object and accepts a constant String reference.
* Rewrite all functions to ensure you do not duplicate code.

# Short Answer Questions

**The mid-term will consist of only a few short answer, concept questions. The following questions are not meant to be an indication of what is going to be on the mid-term but as a tool to see if you understand the concepts presented in the text. If you can answer these questions then you will have a deeper** **understanding of C++ and programming. Remember, you are not only taking this class to good grades, but to learn to be a good programmer.**

### Welcome to Object-Oriented

* What is type safety?
* Does C++ stress type safety?
* What is a name space?
* Why do we use namespaces?
* How do you define a namespace scope?
* What are the ways you can use functions within namespaces? There are at least 2.
* How do you use functions within nested namespaces?

### Object Terminology

* What are the 3 CORE concepts in object orientated programming?
* Explain Encapsulation
* Why do we use encapsulation?
* What is a class?
* What is an object?

### Modular Programming

* What two files compose of a “module”
* What part of a class does a header file contain?
* What part of a class does the implementation file contain?
* What are the 3 stages of compilation?
* What does the compiler do during the preprocessor stage?
* What does the compiler do during the compile step?
* What does the compiler do during the linking step?

### Basic Syntax

* What are the three fundamental types?
* List the types that belong to each fundamental type
* Why are compound types called compound types?
* What is the definition between a function definition and a function declaration?
* What does the include statement look like when using a system library, such as iostream?
* What does the include statement look like when you are include a header file that you wrote?
* What does it mean what a function identifier (name) has more than one definition?
* What is the rule when using default parameters in functions with regard to placement of the default parameters?
* What is a benefit of pass by reference?

### Dynamic Memory

* Static memory is allocated when?
* Dynamic memory is allocated when?
* What is the lifetime of static memory?
* What is the lifetime of dynamic memory?
* What is the keyword that is used when allocating dynamic memory?
* What is the return type of the new keyword?
* How do you allocate a single instances of a type(int, double, etc)?
* How do you allocate an array of a type(int, double, etc).
* Can a static array size be determined from user input?
* What is the keyword used to denote an empty/null pointer?
* What is the keyword used to deallocate dynamic memory?
* What is the syntax to deallocate an array that was dynamically allocated?
* What is the syntax to deallocate an instance of a single type?
* What are two issues that you have to be aware of when using dynamic memory?
* How is a memory leak caused?

### Member Functions and Privacy

* What is the difference between the privacy states in structs and classes?
* What are the classifications of member functions?
* What is the purpose of each of these classes of categories?
* What do you call the variables in a class?
* What do you call the functions that belong to a class?
* Do member functions have access to private data member variables?
* Does code outside of class functions have access to private data members?
* What happens when you put const at the end of a member function?
* What is the syntax to define a member function inside of a class definition?
* What is the syntax to define a member function outside of a class definition?
* Can a member function access global functions?
* What are the 2 levels of accessibility in classes and structs?
* How do you define a class?
* Which file do you put the class definition?
* Which file do you put the class definition in?

### Construction and Destruction

* An object is an instance of what?
* What is a constructor and why do we use them?
* How do you define a constructor in a class definition?
* What is the no argument constructor?
* What is a default constructor?
* When is the constructor of an object called?
* What does it mean to put an object in a safe empty state?
* What is a destructor and what is its purpose?
* Can you call an objects destructor?
* When is the destructor called?
* How do you define a destructor in the class?
* When dealing with an array of objects, in what order are the destructors called?
* Can we have more than one destructor?
* How do you define a custom constructor to take several arguments?

### The Current Object

* What does the “this” keyword reference?
* What does the “\*this” keyword reference?
* In what context can you use “this”?
* Can you use “this” outside of a class?

### Member Operators

* What are the 3 classifications of operators?
* What is an example of each?
* What 4 operators are not allowed to be over loaded?
* What is the form that the prototype takes to overload a binary member operator?
* What is the form that the prototype takes to overload a unary member operator?
* What does the single argument constructor define?

### Helper Functions

* What is a helper function?
* What is a helper operator?
* Does a helper operator have access to the private data members?
* How can we give helper functions/operators access to private data members?
* What is friend functions?
* What does it mean to grant friendship to a class?
* Can a classes that you give friendship to reciprocate the friendship?
* What is downside of using friendship?

### Classes and Resources

* What is a resource?
* What is a resource instance pointer?
* What is a shallow copy?
* What is a deep copy?
* When you want to copy the data pointed at by a pointer, do you use a shallow or deep copy?
* What happens when you shallow copy a pointer?
* What is the process when you want to make a deep copy?
* What are the two special member functions used for managing allocations and deallocations?
* What happens if you do not write a copy or assignment operator?
* When is a copy constructor called?
* What form does the declaration of a copy constructor take?
* When is the assignment operator called?
* What form does the declaration of the assignment operator take?