3460:209 Assignment Dos

# **Assignment Dos myCode: friends, overloading operators, linked lists, and templates**

**Overview**

The purpose of this assignment is to make sure that you know how to write a program that uses friends, overloading operators, linked lists, bitwise operations, templates and namespace. This part of the project’s task list is exclusive to what we are learning in class. Do not use advanced techniques, for example, do NOT use STL templates if we have not yet covered them. If this occurs it will invalidate your program and it will not receive consideration ☹

**HOW THESE ASSIGNMENTS WORK**

You will notice that, in the code that has been provided for you, there are sections at the top and the bottom of the program (where it says PLACE CODE HERE FOR TASK…) that are made to hold each coding effort for that particular task. They are enumerated and sequentially listed in the program. You will need to put your classes, structs, and any standalone function prototypes above main, where it says to place SPECIFICATION HERE. Also, you must place your actual code at the bottom of the program where it says place APPLICATION HERE. The specifications and applications should be within the area that has been created for each task. So, for example, when working on the code for task A you will place your definitions or code along with anything related, such as function calls to other related functions, after PLACE CODE HERE FOR TASK A and before TASK B. Declarations at the top and code at the bottom will go between those TASK comments for all tasks. You need to use block comments to test out tasks. If you work in order, the block just moves downward.

DO NOT REMOVE ANYTHING ALREADY IN THE DRIVER b/c it results in a zero :(

What happens is the program is graded by examining each task’s area for the appropriate code and declarations/definition. **IMPORTANT: If your code is misplaced the grade will be zero for the task.** Also, **do not inline** any methods unless asked to. Write your code as it appears in the order provided in the instructions. Ask any questions about this process.

**PROGRAM SPECIFICATION**

For the assignment, we will build out a variety of code to exercise the central parts of what we have learned in the current Unit. Download the cpp file on the Brightspace repository called myCodeDos. Please see above as to how to use this program. Driver code is there and we write the parts that make the driver of each task work. You shouldn’t change the driver unless there’s something that needs to be changed to make it work. There is a menu for testing and a pause after each task (use cin.get()). Our program will be separated into 5 different parts as described. **Get an early start on this one because it is hefty!**

TASKS

1. Creating a string class is one of the most frequently asked C++ related job interview questions (it actually was ranked as the #2 question), so this assignment is something that you should look at as a reflection of your own skill. It’s all about you, and getting prepared for what’s ahead.

This is an inefficient string, but it will provide us with the practice at creating a dynamically allocated string, writing methods to support the string, and other goodies such as copy semantics. You should use as much from the C-String library as possible, and you should not have anything referencing the standard string – std::string. If there is then it will be difficult to get any points for this task, so be careful.

We will write a class called myString. Implementing myString requires you to provide many, many functions, discovering the joy in operator overloading. You will find the defined myString class and all associated functions’ declarations in myCodeDos cpp. Download it. Most of these functions and overloaded operators are relatively straightforward in the specifications. You will write all of those functions in the task A coding area. Review the code in the driver and the class definition before starting. You must make the test cases, which are also given to you, properly work as described in the comments. For task A you need to:

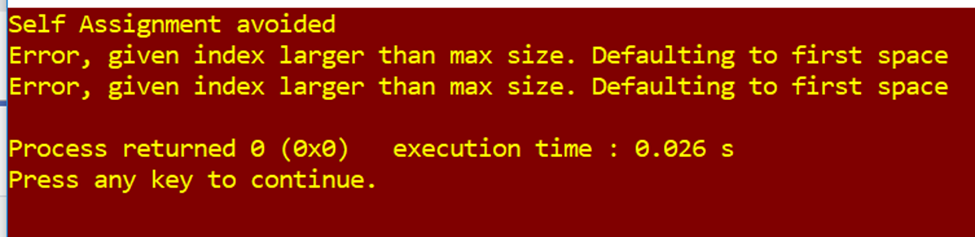
• Create the many supporting functions for the myString class

• Provide overloads as described

Naturally, you will also be required to provide a destructor that releases or deletes any memory allocated to the string in a constructor or assignment operator.

Overloads of streaming operators (`<<`) have been provided for you.

## When you run your test, it will (should) result in something along these lines…



1. Create a set of three functions that take a pointer to an array as an input parameter and reverses the contents of the array. The function should use two pointers, front and rear. The prototypes for functions reverse1, 2, and 3 hath been put in place for you. The front pointer should initially reference the first element in the array, and the rear pointer should initially reference the last element in the array. Reverse the array by swapping the elements referenced by front and rear, then increment front to point to the next element and decrement rear to point to the preceding element, and so on, until the entire array is reversed. So the function must use two pointers, and perform the reverse process entirely with those pointers (front and back).

The program also already contains a constant mySize shared by all with the three prototypes for three different data types held by these arrays. The driver in task B includes the code to call the functions as well as handle the output of the results. The approach to writing these functions is to be able to have one function handle bool type, char type, and short type with the same logic. So the goal for writing these first three reverse functions should not only be to write the solution, but also to universally accomplish it with the functions being clones.

## For task C we will take these functions and overload them into one, same-named function called reverse. If all has gone well thus far, you should be able to simply cut and paste. This is going to be bulky in the amount of repeated code, but that is exactly what we want to exploit here, so no worries.

## For task D, create a template function and call it reversal. You should be able to take any one of the functions that you have written thus far, and use it for the template version. Remember to include the prototype.

1. For the task, we will design an ascending order arranged linked list class. This linked list class should hold a series of Employee objects. The linked list class should have member functions for displaying, appending, inserting, and deleting nodes. Also, it will need a destructor that destroys the list.

For this assignment, the Employee class and all of the methods have been provided for you. You will obviously interface with them in your linked list code. Do not incorporate or hard code anything other than the parameters and data type (Employee) into your linked list class. Note we could attempt a more generic version of the linked list class by use of templates, but that would also make this a longer engagement than desired…

For this assignment, you will need to provide these public member functions that perform the following tasks:

1. **appendNode** appends a node containing employee info, to the end of the list. The method supports either an Employee object or the Employee member values passed as arguments. See the myCodeDos for a concrete example.
2. **insertNode** this function inserts a node with the employee info logically into the list. This function must insert based on the last name values in ascending order. So, the list is ordered by last name, lexicographically, with duplicates permitted. The method supports either an Employee object or the Employee member values passed as arguments. See the myCodeDos for a concrete example.
3. **deleteNode** this function searches for a node with last name as its parameter’s value as the element to find. The node, if found, is deleted from the list and from memory. If two or more nodes with the same value are found, only one is removed (your choice which one). If none are found, no action is taken.
4. **displayList** this function traverses through the linked list and produces output as follows:

Appending to the list

Lisa Able 4500

Mark Cork 4000

Yoo Hoo 720

Boo Hoo 16060

Sally Sea 22250

Zippy Zip 150

Now inserting into the list

Lisa Able 4500

Mark Cork 4000

Yoo Hoo 720

Boo Hoo 16060

Estoy Rico 75000

Sally Sea 22250

Chuck Ster 5050

Zippy Zip 150

Now the reduction in force :(

Lisa Able 4500

Mark Cork 4000

Yoo Hoo 720

Boo Hoo 16060

Estoy Rico 75000

Sally Sea 22250

Chuck Ster 5050

Make sure that your programs follow good documentation standards and follow the requirements for assignments. Use pre and post condition comments. Do not use namespace standard. Reference the rubric standards on Brightspace.

Submission Instructions – for programming solutions

On Brightspace go to the matching Assignments for the ASSGN@, where @ is the number or character of the problem assigned and submit the program (cpp) file.

*Last updated 8.22.2020 by Will Crissey.*

*Be aware that programming falls under all of the rules of plagiarism. Be careful when using any coding found in the outside world that is not your own. Any evidence of plagiarism is subject to sanctions like forfeits, suspension, and even ejection, as determined by the Department of Student Conduct and Community Standards.*

**COPYRIGHT © 2020 Will Crissey, Jr., Associate Professor, All rights reserved.**

This document is intended exclusively for use by The University of Akron's Department of Computer Science. Do not distribute document copies to the public in any way.

***THIS PROGRAM'S CONTENT IS PROTECTED***

The document and associated content are either the property of, or used with permission by, The University of Akron, Department of Computer Science and may be protected by copyright and other restrictions. Copyrights in the site content may also be owned by individuals and entities other than, and in addition to, The University of Akron, Department of Computer Science. The use of this document by you, or anyone else authorized by you, is prohibited, except for purposes of completing the homework/project. Any unauthorized use of the document may violate copyright laws, trademark laws, the laws of privacy and publicity, and communications regulations and statutes.