Lab 10

Resources

- Discussion slides (./lab10slides.pdf) - Copy of the presentation given by the TA
- Nemo's Discussion on Operator Overloading (http://www.cise.ufl.edu/~nemo/cop3503/slides/Lecture%2024_my_operator_overload.ppt) a good usage of overloading with Fraction class
- main.cpp (./main.cpp) - The main I will be using to test.

Idea

This lab will help you get familiar with operator overloading in the context of dynamic arrays. As s
you will also need to learn how to write a dynamic array.

In-lab assignment

Description

There are two main parts to this lab, writing a Dynamic Array and overloading some operators. Th
dynamic array should be written based on the discussion and needs to have the specified behavior.
duplicates in the array should be discarded.

Requirements/Deliverables

Dynamic Array Stuff

You will write a dynamic array class called DynamicArray. The elements to add will be strings. Plea
write the following constructors/destructors:

1. `DynamicArray()`

   Creates an empty dynamic array. The intial capacity should be 5. It is okay to make this a mag
   number.

2. `DynamicArray(int capacity)`

   Creates an empty dynamic array with the given capacity.

3. `~DynamicArray()`

   Deconstructor. Should free any memory used.

The array needs to support the following functions (again, name them exactly as they appear here)

1. `bool insert(string)`

   This function will just insert the given string into the array at the end. It should return true up
   successful insertion, false otherwise.

2. `string remove()`

   This function removes and returns the final item in the array. If the array is empty, return "".

3. `string remove(int index)`

   This function should remove the string at the given index from the array. The string removed i
   what should be returned. On failure, return "". Note that the string should not be in the array
   anymore. The relative positions of the other elements should be the same.

4. `void remove(string str)`

   Removes a given string from the array. If the string is not in the array, DO NOT PRINT AN ER
   MESSAGE, just return quietly.

5. `int index_of(string str)`

   This function returns the index of the given element. It returns -1 if the element is not in the ar

6. `string item_at(int index)`

   This function returns the item at a specified index. Note that the item should remain in the ar
   after this function call. If the index is invalid, return "".

7. `int size()`

This function returns the current size of the array. This is the logical size.

8. ```
   int capacity()
   ```

   This function returns the current capacity of the array. This is the physical size.

9. ```
   void clear()
   ```

   This function logically clears the array.

These are the functions that must be public. If you want to write other functions, then that is fine. you need to write the following *private* function:

1. ```
   void grow()
   ```

   This function grows the array as discussed in class. Dont forget about memory management!

In addition to these functions, I **highly** suggest you write the following functions before moving on operator overloading:

1. A copy function that takes in a reference to another DynamicArray and sets the contents of thi DynamicArray to be the same as the one passed in.
2. A function that can take in one or two other DynamicArray objects and add each of there elem to this DynamicArray
3. A toString function that returns a string representation of the DynamicArray in the format spe below.

Operator Overloading Stuff

You must overload the following operators for your DynamicArray class:

1. `=` - The result of equality should be that the array in the first array contains the same exact th as the one in the second. Note that the two arrays *should not share* any memory.
2. `+` - The result should be a new DynamicArray object that contains the elements from the first concatenated with the elements of the second. You should keep the relative order of each array wary of duplicates.
3. `[]` - Should return the item at a given index
4. `<<` - Print out the given DynamicArray to the given ofstream. The format should be as follows: `[element_0, element_1, ....]`. For example: `[apple, pear, tricyle]`. *There should be* newline at the end.

Submission
   ```

Submit a DynamicArray.cpp and a DynamicArray.h file with your implementation of the Dynamic
class. It is due Saturday night. Again, I will use my own main to test your functions as well as revie
code.

Hints

- Get all the functioanlity of your array working before you start on the operator overloading pa
- Be wary of memory management. When you allocate a new array, you need to delete the old on
- The `new string[]` and `delete [] ptr` calls are used for allocating arrays.
- If you are having lots of trouble, try removing the const modifiers and add them back in slowly

Grading Distribution

- 10 point for implementation of each of the functions in your DynamicArray class (~ 1 point ea
function that doesn't work).
- 10 points for correctly overloading the operators

Optional Enhancements

- Overload the `+=` operator as well.
- Make it so your Dynamic Array keeps a sorted order. Is there a way to write the find function s
it runs quicker now?
- Implement an insert(int index, string str) function that inserts a given item at the given index.
that you must keep all of the other elements in the array in the same relative order
- What if we wanted a DynamicArray of floats? ints? doubles? any type? Look up templates and
see how to accomplish this