Ryan Filgas
CS 163
Karla Fant
Program 2 Design Writeup - Resume

My program will implement a class of resume experiences entered by the user. The class will manage a linear linked list of pointer arrays where each pointer leads to a class object of type experience. The goal of this assignment is to create an abstract data type which implements stacks. Given this, the client program will be able to push new class objects onto the stack, pop them off while displaying the data, peek to display the first item, and display the contents of the stack. With a quicker implementation of this program, It should be possible to make use of the pop function in peak, and display, using const to protect the data, which will allow for reduced overhead, and easier to understand code.

<u>The public section of the resume class will include:</u>

push – push is a function that will add a new experience object onto the stack. To do this it will check if there isn't a list, and if there isn't, create a new node, and copy data to position zero, then increment top. If top is less than five it will copy an experience object from the client into the array at index top inside head. If top is 5, set it to zero, make a temp pointer to hold onto head, make a new node, copy data to the new node, and connect head's next pointer to the hold pointer. It will return 1 for success, and 0 for an empty list.

pop – Is a function that displays data from the top of the stack, and deletes that data from the stack. It will check if the list is empty, and if it's not empty it will call the display function from the experience class on the experience object at index top, delete that object, and decrement top. If there was only one object in the list(top is one, and heads next pointer isn't null), it will delete the list. Pop will return 1 for success, 0 if there is an empty list.

peek – Is a function that displays data from the class object at the top of the stack. The peek function can call the display function from the experience class on the experience class object at the top index pointer - 1. In this case the pop function will check head, and then display the top position if there's a list. It can be assumed by the structure of the other functions, that if a list exists, it will not be empty. Peek will return 1 for success or 0 for an empty list.

display_resume – Is a function to display the contents of the entire stack. display_resume will make use of a display node function that calls a function named display experience on each node, and another function to traverse. This will execute until head is null since its only goal is to display data. In this case the display function will check for an empty list, and execute the code otherwise. The display node will need to display from the top index to index 0, for the first node, and every index for the following nodes since a stack structure mandates the other nodes will be full. The display function can call the sub functions once for the first node, and set top to 5 before calling the sub functions again in a do while loop.

Ryan Filgas
CS 163
Karla Fant
Program 2 Design Writeup - Resume

The private section of the resume class will include:

experience node head – This will hold a node of type experience.
An integer called top_index to track where in the stack to add or remove.

display_node – This function will call the display_experience function on each item in a node starting with index -1. It will reset the index to 5 at the end, so that when it's called for the next node, we can assume it's full and display the contents from each object in the stack.

The public section of the experience class will include:

Display_experience – This function takes in no argument, and outputs the data contents of an experience object.

copy entry – This function takes an experience object as input, and copies it into the classes memory.

input entry – This function allows the client to build an experience object by putting in name, start date, end date, job title, and list of responsibilities as arguments.

The private section of the experience class will include:
Employer, or club name as a char array, start date as month and year ints, end date as month and year ints, job title as a char array, and a list of responsibilities as a char array. All arrays will be dynamically allocated.

Experience node struct
The experience node struct will contain a pointer array where each member is a class object of type experience. It will also contain a next pointer.

Ryan Filgas
CS 163
Karla Fant
Program 2 Design Writeup - Resume

Design Considerations

The main design considerations in this project will be the implementation of stacks to manage data, and using the data structures of a linear linked list, and pointer array in combination to get the best of flexibility with the LLL, and the runtime efficiency of a pointer array. As the program will be an ADT, it strictly must manage data, and operations on that data; this means no error messages to the user, and no expectation of the client to implement any part of the data structure, as it will be hidden through the class. The only return to the calling function for the ADT will be either success, or failure. This project specifically will focus on readability of code, and further data abstraction through the use of helper functions whenever possible. This project may also focus on further runtime efficiencies if time permits by using pointer addition rather than indexing using brackets. Proper allocation and deallocation of dynamic memory will be important, as well as proper testing of each function through a test program and user interface.