Question 12

I was unfortunately unable to complete this question because of time constraints. (Had many midterms to study for and these questions took a lot of time). In this paper I will try my best to explain my attempt at this question and the progress I made. I started this problem by creating an entirely new doubly linked list class that was needed for this program. My approach to this problem was to add a few different components to a node which is not parallel to a traditional linked list, however I used this to best adjust to the prompt requirements. The struct node had 4 different data values, the day of the appointment, the length of the appointment, and the time of the appointment and then 2 node pointers, one for the next node and one for the previous node.

The doubly linked list begins with a “addAtEnd” function that takes in a reference pointer to the head, the day, length, time, am or pm, as parameters to this function as do most functions in this class. The function creates a new node and creates a new node pointer that will be used at a later time. Then each part of the node is assigned to what is passed through and the next pointer of this node will be assigned to nullptr because it is being added to the end of the list. Then a while loop will be used to traverse the list using the lastN pointer that was made at the beginning of the function and it will then assign the next pointer of the lastN node point to the new node that was made, then the previous pointer of the new node will be assigned to lastN. Then we have similar functions called “addBefore” and “addAfter”. These functions are very similar and only differ in the assigning of pointers . In addBefore, the previous pointer of the new node is set to the previous pointer of the next node. Then the previous pointer of the next node is set to the new node and then the next pointer of the new node is set to the next node. In addAfter, I set the next pointer of the new node to the next pointer of the previous node. I then set the next pointer of the previous node to the new node. Then to conclude I set the previous pointer of the new node to the previous node. The part that I probably did incorrectly and could not finish was the function called “createAppointment”. At the beginning of the program I create and empty int array with a size of 5 to essentially “remember” which days of the week have been used. So when createAppointment is called and the day is not used, the program will call addAtEnd and add it to the end of the program. If the day has already been used, the program traverses through the list until you reach the day that the function is passing through. Then there are many different conditions in this program that I use to try and sort the array but it does not work properly as it is not checking and sorting the list properly. It more so just looks at the node that is ahead of it rather than the entire list. I was stuck on this for a long time and therefore I didn’t get to complete any of the other program requirements. I apologize for the inconvenience.