Question 12

In this implementation of a simulation of appointments being booked using a doubly-linked list, a few parts needed to be created in order to create the linked list.

First the node class. In this class, each node has two pointers, next and prev. Next points to the next node and prev is a pointer from the next node pointing back to the previous node. In addition, each node has three pieces of information, the day, the time the appointment begins, and the time the appointment ends. This will help keep the list organized, and keep track of which appointment is before which appointment.

In our doubly liked class, we have functions that will help us build the linked list according to the needs of the office manager.

First, we have our default constructor setting our list to null

The print function which iterates through the linked list printing the necessary information for each node.

The length function, which returns the length of the linked list.

The insert function, where each node/appointment is inserted one by one building up our linked list. In this function, we have a few different case scenarios of how and where to insert each node, based on the information provided by the one booking the appointment.

Our first case: if this is the first node being inserted in the list

Case two: if there is one node/appointment in the list

Case three: there are already nodes inserted / appointment booked, and we're looking to insert a new appointment, this requires a few checks in order to know where to insert the node.
We want the nodes to be in chronological order, meaning the earlier appointments in the week should come first, therefore adding a node requires checking its time and day in order to see if it can be inserted after a current node.

Our first check in function IsAfter which checks if one appointment is after another by comparing their day and time.

The second check, in function IsTimeAfter, we're looking to see in the case that an appointment is after the previous appointment, is there enough time in the workday (9-5) to fit that appointment in for that day. If there is then we insert it at that spot.

Next, if we see the requested appointment is not after our current appointment, we simply keep moving down the list looking for the correct spot to insert it.

Another case is what happens if we reach the end of the list. First, we check if the last appointment is on Friday then we check if there is time left on that day, if there is we insert the appointment there, if there isn't then we set the appointment to the first appointment in the next upcoming week.

Another case is what happens if a customer requests an appointment that is already taken, then we find them the closest available appointment. This is done by calling function NextApp which checks the next node/appointment and based on that we find the next available appointment. If an appointment cannot be inserted before the next appointment then we simply move down and check the next node/appointment until we find the nearest available appointment.

Our last case is if a customer requests to book an appointment which happens to be before the first node/ appointment on the list, then we simply make that the first appointment by attaching it to the front of the list.

Another function in the class is deleted Appointment, in this function were taking in the day and the time a certain appointment begins, locating it in our list, and then deleting it. Here angina we have a few cases, the first if the appointment being deleted is the first one in the list, if it is in the middle of the list, and if it is the last appointment in the list. In each case, we rearrange the pointers in a specific manner in order to ensure no information is lost.

Another two small functions that are utilized in this class are TimeBetween which checks the time between two appointments, and AppLength which checks the lengths of an appointment.

Note: when calculating the days and hours of operation we assumed Monday to be day 0 up until Friday which is day 4, and the time to go from 9 am - 17pm (9 am-5 pm) with options to book appointments with fractions of hours like 9.5-10.75 (which is 9:30-10:45) which works because type T can be a double. Therefore you will see calculations with those assumptions in place.