```c
/* BY SUBMITTING THIS FILE TO CARMEN, I CERTIFY THAT I HAVE PERFORMED ALL OF
THE WORK TO CREATE THIS FILE AND/OR DETERMINE THE ANSWERS FOUND WITHIN
THIS FILE MYSELF WITH NO ASSISSTANCE FROM ANY PERSON (OTHER THAN THE
INSTRUCTOR OR GRADERS OF THIS COURSE) AND I HAVE STRICTLY ADHERED TO THE
TENURES OF THE OHIO STATE UNIVERSITY'S ACADEMIC INTEGRITY POLICY.
*/#include "lab4.h"
void deleteNode(Node *head, int id) {
        Node *priorNode;
        Node *traversePtr;
        /*If list is empty, nothing to do*/
        /*Checks to see if the node to be deleted is the first node. If so, make head
pointer point to same thing as traversePtr->next and free the traversePtr*/
        if (head->student.student_ID == id) {
                traversePtr = head;
                head = traversePtr->next;
                free(traversePtr);
        }
        /*Non-exception case*/
        else {
                priorNode = head;
                /*Traverse the list to search for the node to be deleted*/
                traversePtr = priorNode->next;
                while (traversePtr != NULL && traversePtr->student.student_ID != id) {
                        priorNode = priorNode->next;
                        traversePtr = traversePtr->next;
                }
                /*If node is found, change next pointer of prior node and free the
node to be deleted*/
                if (traversePtr != NULL) {
                        priorNode->next = traversePtr->next;
                        free(traversePtr);
                }
        }
}
```