Name:

Quiz #1

Assume you construct a single linked list with patient records as nodes. Implement an algorithm to delete a patient record (a node) in the middle of that single linked list, given only access to that patient record (node). For example: Node d should be removed from the linked list a → b → c → d → e. Your algorithm should take one input, i.e. pointer to node d. Your algorithm should return true if the node is successfully removed, i.e. the linked list will look like a → b → c → e. Your algorithm should return false if the input node is not in the middle.

Answer:

The solution to this is to simply copy the data from the next node into this node and then delete the next node.

NOTE: This problem can't be solved if the node to be deleted is the last node in the linked list.

```
boolean deleteNode(LinkedListNode *n)
{
	if (n == null || n->next == null)
	{
		return false;
		// Failure
	}
	LinkedListNode *next = n->next;
	n->data = next->data;
	n->next = next->next;
	free (next);
	return true;
}
```

Name:

Quiz #2

Write a function to swap two adjacent elements in a doubly linked list by adjusting only the pointers (and not the data).

Answer:

here's the code you need to swap two adjacent node pointers assuming that x is previous to y:

```
void swapAdjacent( Node &x, Node &y )
{
	if( x.prev ) x.prev->next = &y;
	if( y.next ) y.next->prev = &x;
	y.prev = x.prev;
	x.next = y.next;
	x.prev = &y;
	y.next = &x;
}
```