**Exercise on Linked List Using C Programming**

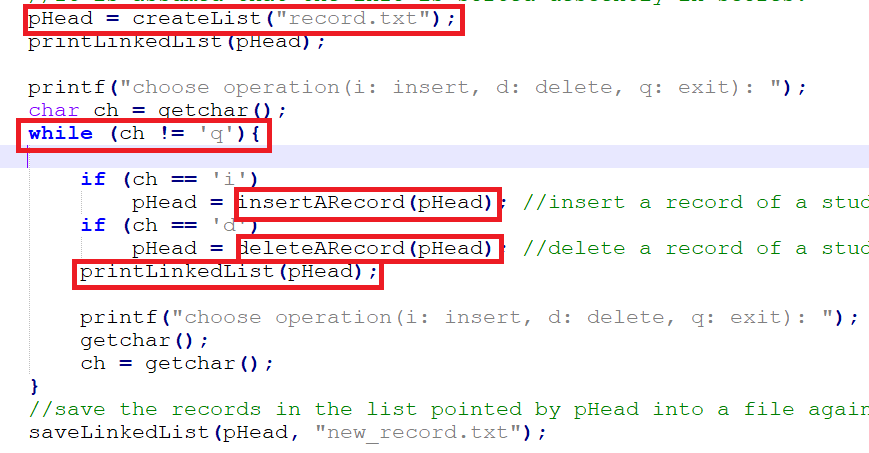
This exercise is to practice the use of C programming language to implement operations on a linked list.

|  |  |
| --- | --- |
| typedef struct node{  char name[20];  int score;  struct node \*next;  } Node;  Node \*pNode, \*head  // create first node, head of linked list  pNode = (Node\*)malloc(sizeof(Node));  strcpy(pNode -> name, "John");  pNode -> score = 100;  head = pNode;  //create second node  pNode = (Node\*)malloc(sizeof(Node);  strcpy(pNode -> name, "Tony");  pNode -> score = 90;  //link them  head -> next = pNode;  pNode -> next = NULL; |  |

In this exercise, we will do the following tasks by filling in the blanks in the given code (LinkedListIncomplete.c):

1. create a list
2. insert a node
3. delete a node
4. save a linked list
5. print a list (done after the create a list task)

To facilitate the testing of the program, the main function has used a *while* statement so that in one run of the program, user can add or delete a node for multiple times.

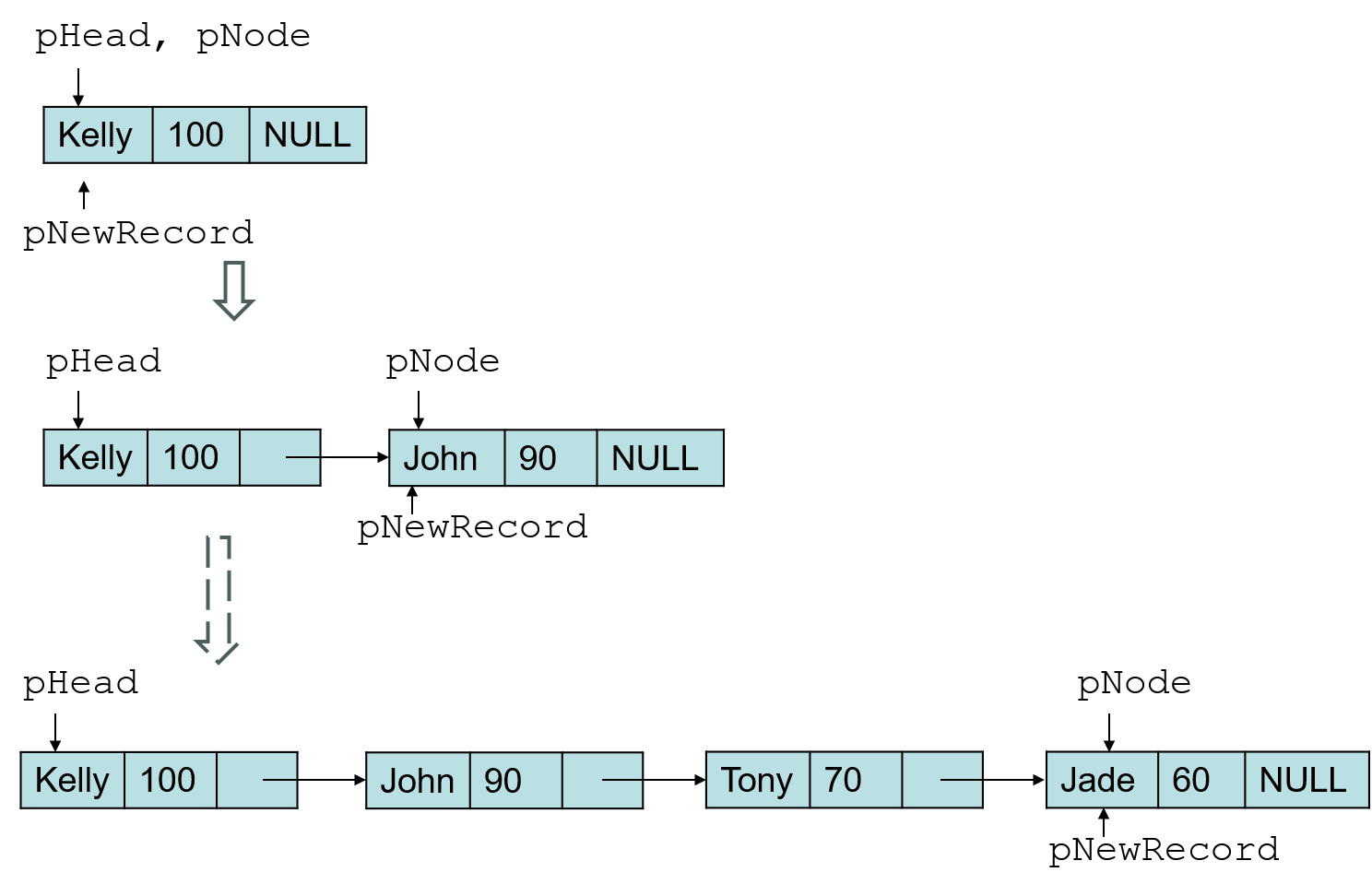


**Create a list**

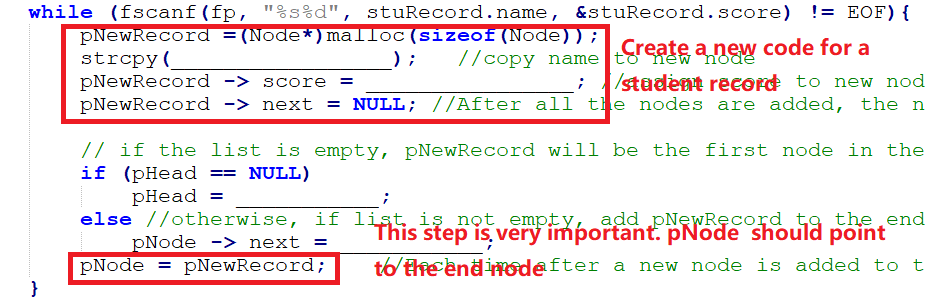
In this task, we are going to create a list using the information from a text file *record.txt*. The content of record.txt is

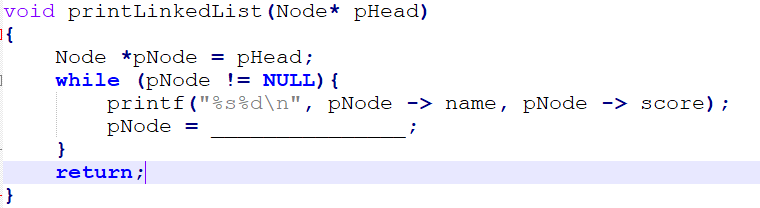
|  |
| --- |
| Kelly 100  John 90  Tony 70  Jade 60 |

The records in the file are sorted already in descendent order in score. When you create the list, this order should be kept. That is, the list should be created as shown here:



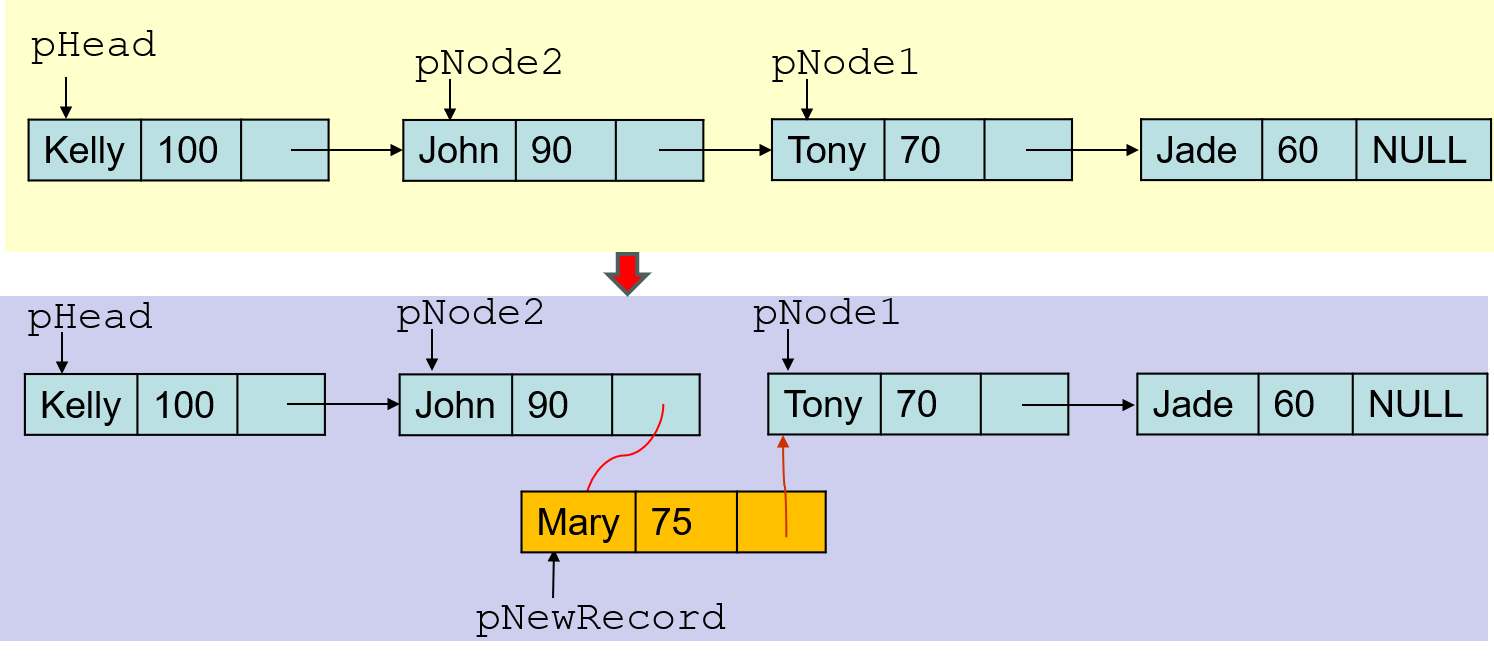
Fill in the blanks in the function ***createList*** in the given code.



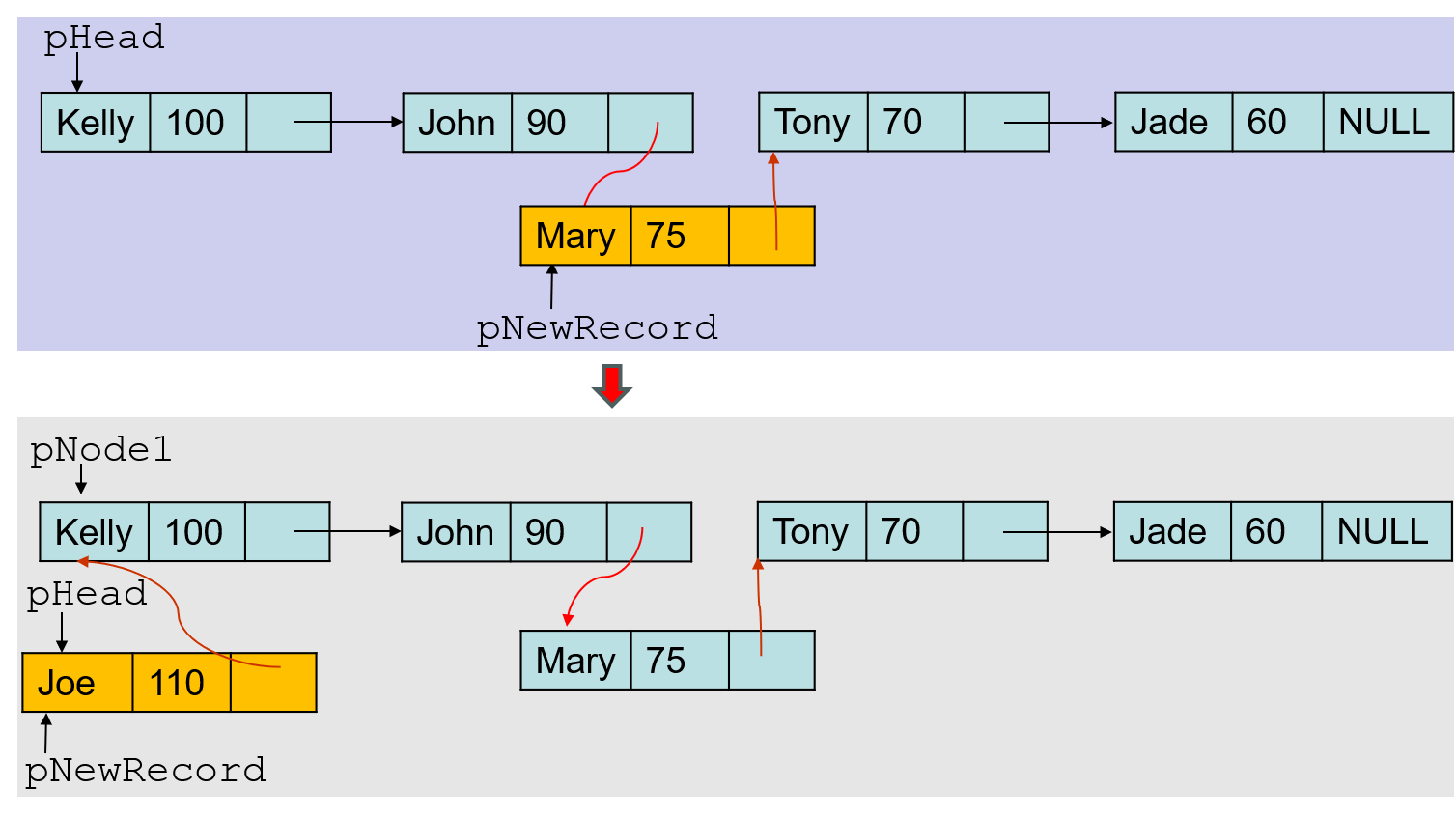
To print out info in the linked list, you also need to fill in the blanks in the function ***printLinkedList***. 

**Insert a node**

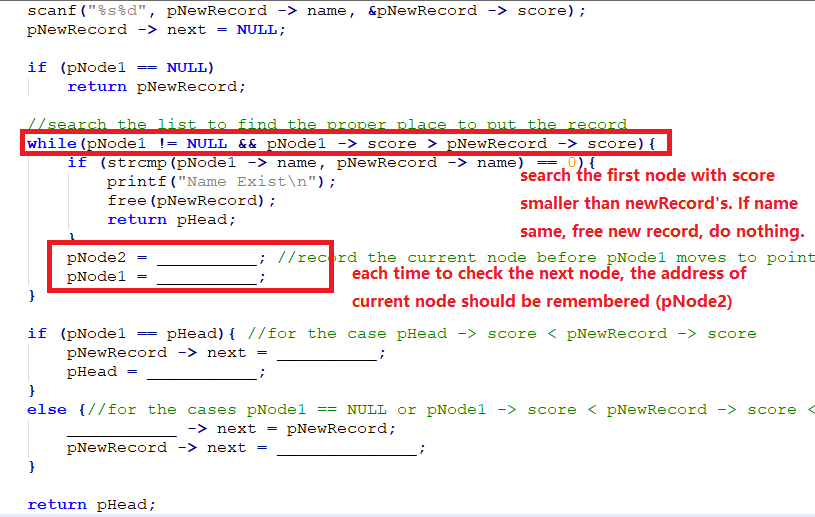
After the linked list is created, new record can be inserted at the proper place. If the name of the new record exists in the list, the new record will not be inserted. For example, if we want to insert a new record (Mary, 75)， the linked list will be changed as follows.



Then if we further add a new record (Joe, 110), the above list will be changed as follows.

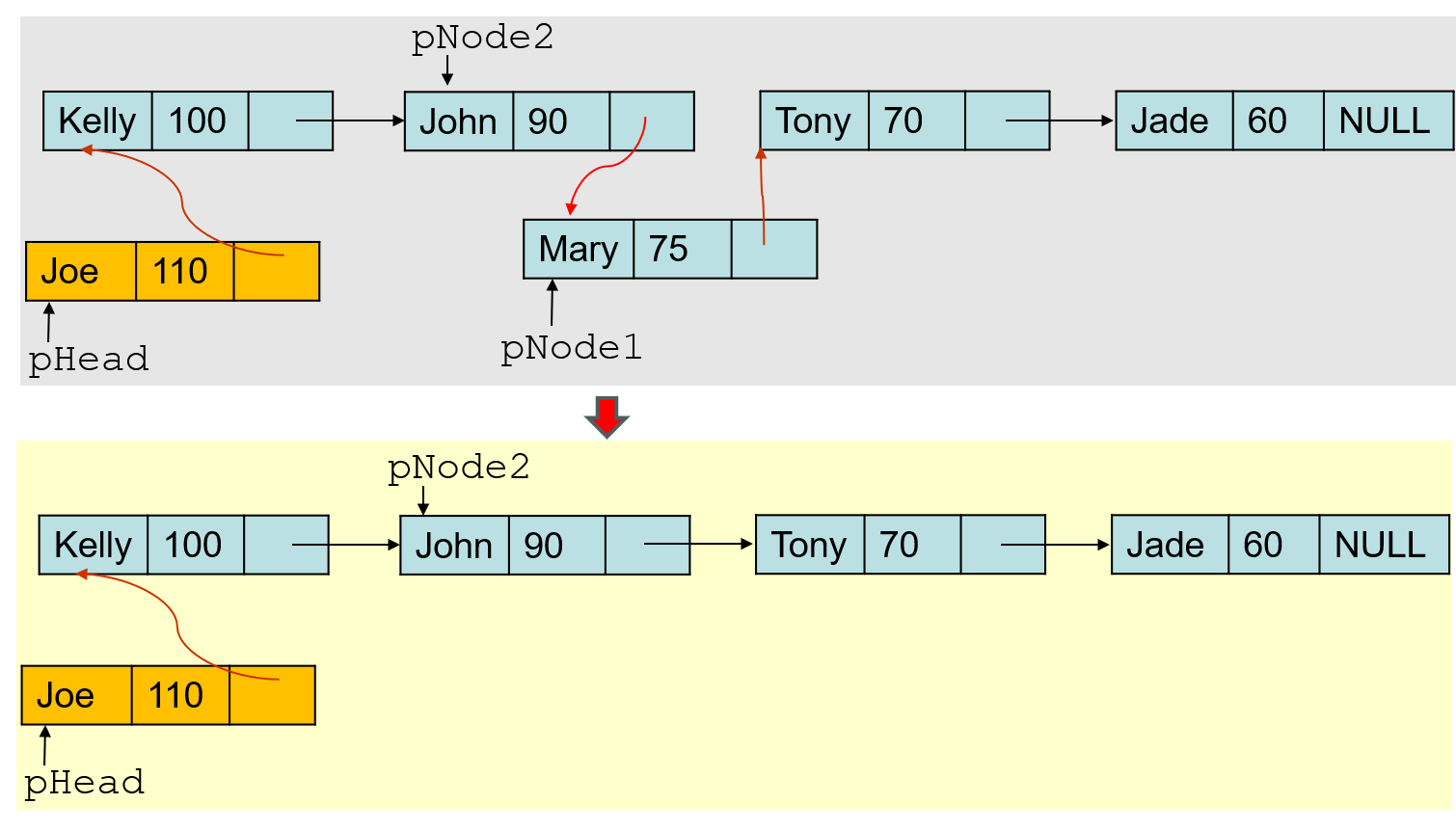


Please fill in the following blanks in the function ***insertARecord*** in the given code.

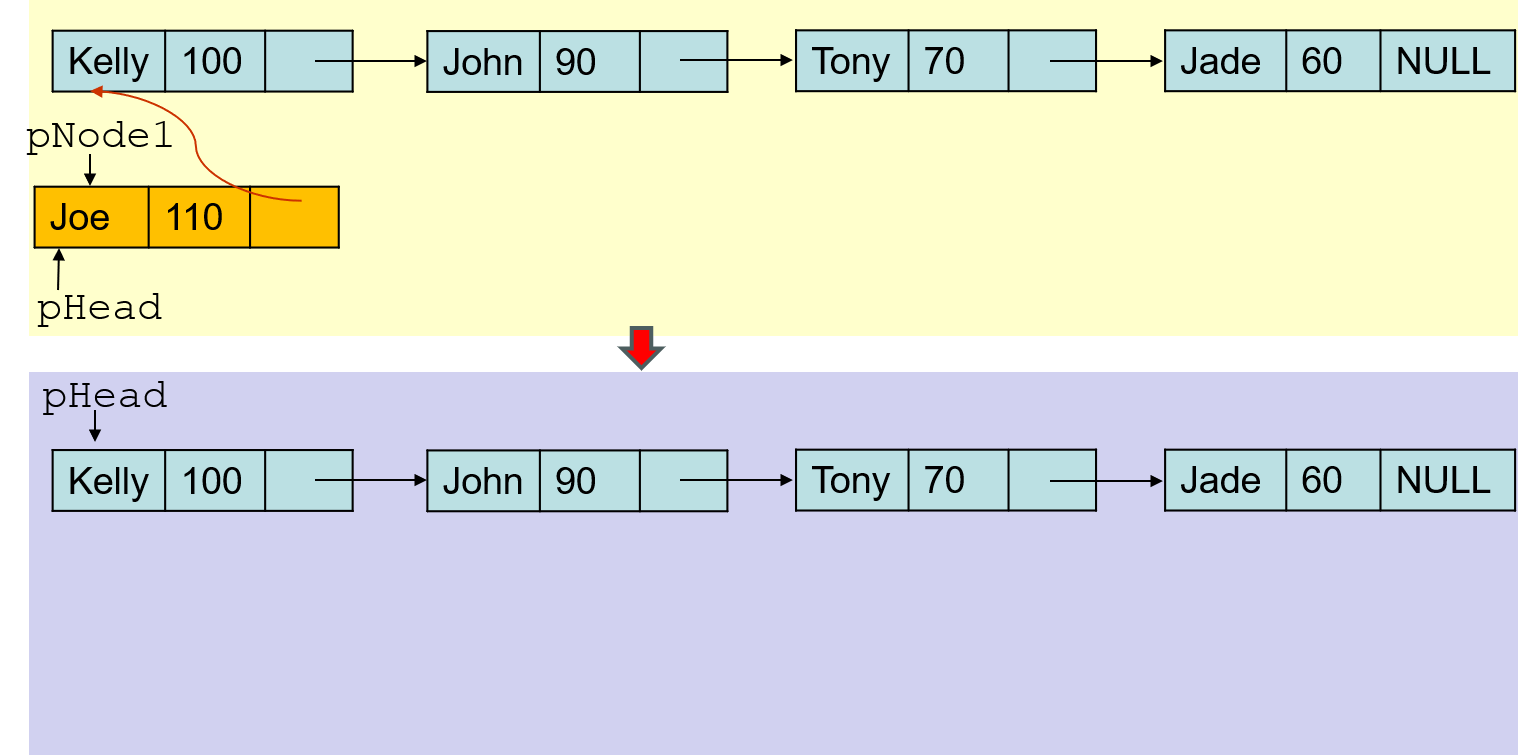


**Delete a node**

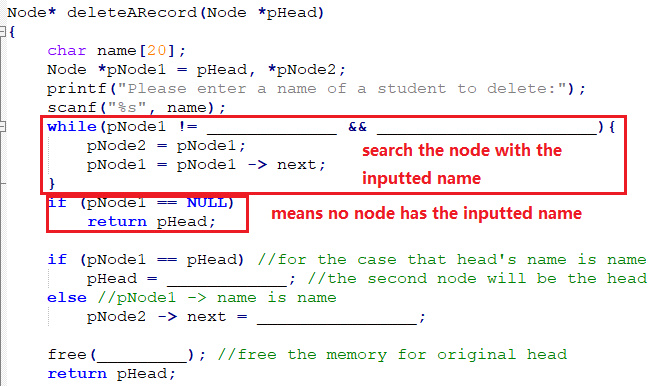
Now if we want to delete a record (Mary, 75), the current linked list will be changed as follows.



Then if we delete another record (Joe, 110), the above linked list will be changed as follows.

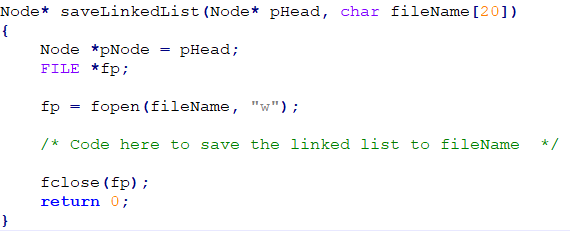


Please fill in the following blanks in the function ***deleteARecord*** in the given code.



**Save linked list**

Finally fill in the code in the function ***saveLinkedList***.

****

Check the content in the file *new\_record.txt* after you quit the execution.

**Sample I/O:**

