Linked List Implementation

Problem Description:

A list of product records is represented in internal memory using singly-linked list implementation. The product ID is the basis for uniquely identifying product records. Given below is the data structure definition.

```
typedef struct {
  unsigned int prodID;    /*  product ID, uniquely identifies an element */
  char prodDesc[15];      /*  product description*/
  float prodPrice;        /*  product price*/
  int prodQty;            /* product count or quantity  */
}product;                 /* product record */

typedef struct cell {
   product item;
   struct cell *next;
}*listType;              /*  Definition of the ADT List */

typedef enum {
      TRUE, FALSE
}boolean;
```

The Codes of the following functions are provided:

Function Prototypes	Description
void populateList(listType *L);	**Creates a list of 10 elements by calling insertFirst() 10 times. This function is called in main only if the function insertFirst() is already created.**
void populateSortedList(listType *L);	**Creates a sorted list of 10 elements by calling insertSorted(). Suggestion: Reuse data in populateList().**
void displayList(listType L, char * listName);	**Displays the information of each of the elements of the given list. The 2nd parameter is the name of the list.**

Write the code for the following functions:

Function Prototypes	Description
void insertFirst(listType *L, product X);	**Inserts a new element (product) at the first position of the list.**
void insertFirstUnique(listType *L, product X);	**Inserts product X at the first position in the list ONLY if product X is NOT yet in the list; else add the quantity of product X to existing element with same ID and make product price of X the new price.**
int insertSorted(listType *L, product X);	**Inserts a new element in the list ONLY IF the list is sorted in ascending order according to ID. If the new element is successfully inserted in its proper position in the sorted list, then return 1 to the calling function to denote "success"; otherwise return 0. If malloc() fails, 0 is also returned.**
product delete(listType *L, unsigned int prodId);	**Deletes the element with the given ID and returns the deleted element. If the element does not exist, return a dummy product record, i.e. record with an "XXX" for the product description and 0 for the rest of the fields.**
void displayProduct(product X);	**Display the information of a given product in 1 single line, with a header Suggestion: Reuse statements in displayList().**

FINISH UP THE PARTIAL PROGRAM FOUND IN TEACHERFILES. FILENAME: chocoList.c