Homework Assignment, September 28th: Arrays, Pointers and introduction to Abstract Data Types.

Arrays and vectors can be used to hold lists. For this homework assignment, you will be making a book list program. Write a program to support the following functions on a book list. You can assume that you will not be buying more than 20 books:

Insert new element at the end of the list
Insert new element at a certain position
Find an element (with its ISBN number and list its position) using linear search
Find an element (with its ISBN number and list its position) of a sorted list using binary search
Delete an element that is at a certain position
Delete an element by using its ISBN number
Sort the list by the ISBN numbers (using selection sort) – **Use pointers for accessing array elements**
Sort the list by the ISBN numbers (using bubble sort) – **Use pointers for accessing array elements**
Print out the list

For the sorting functions, do not use the array indices to access the array elements. For example, do not use array[5]. Instead, use pointers to access the array elements (example: *(array + 5)).

Your program should begin with a loop asking the user what they want to do. Clearly, at the beginning since there is no list, they have to begin by inserting a new element at position 1 (note, when calling the function to insert an element at a certain position, you have to check that the position is within bounds of the current book list).

For your implementation, you can assume that the user will enter the ISBN number of the book (so you can use integers for your list). Use arrays or vectors. (If you use vectors; however, you must implement all of the functionality below and not use the built-in functions on vectors for this assignment.) You only need to use either arrays or vectors.

The function headers should look something like the following (variables names and parameters don't need to exactly match):

```
void insert(int mylist[ ], int num_in_list, int new_element);
void insert_at(int mylist[ ], int num_in_list, int at_position, int new_element);
int find_linear(int mylist [ ], int num_in_list, int element);
int find_binary(int mylist [ ], int num_in_list, int element, bool sorted);
void delete_item_position(int mylist [ ], int num_in_list, int position) ;
void delete_item_isbn(int mylist [ ], int num_in_list, int element) ;
void sort_list_selection(int mylist [ ], int num_in_list);
void sort_list_bubble(int mylist [ ], int num_in_list);
void print (int mylist[ ], int num_in_list);
```

Things to note:

1. You can assume that the first digit in the ISBN numbers stored will not be 0.
2. When you insert or delete an element from an array at a certain position, you will have to move up or move down the rest of the items on the list. You can use any algorithm you wish to do this.
3. For both insert and delete, you must check that you do not insert more than 20 items or delete from a list that has no items in it.
4. If there is more than one occurrence of an ISBN you can just delete the first one
5. For the binary search, you will need to check to see if the list is sorted already. If it is, binary search can be used. If it is not sorted yet, you will need to print out that the list needs to be sorted first before performing binary search. You can just use a variable to keep track of if the list has been sorted before using the binary search option.
6. Make sure to use pointers to access array elements for the sorting functions. Do not access the array using indices such as array [5]. This is to make sure you have an understanding of how to access array elements with pointers.

Please see the next page for an example of what the program could look like.

```
SAMPLE EXECUTION OF PROGRAM:

Welcome to the Book list program.
What would you like to do?
        1.  add an element to end of list
        2.  add an element at a location
        3.  find an element by ISBN number (linear search)
        4.  find an element by ISBN number (binary search)
        5.  delete an element at position
        6.  delete an element by ISBN number
        7.  sort the list (using selection sort)
        8.  sort the list (using bubble sort)
        9.  print the list
        0.  exit
your choice ---
1
Please type in the element
1454226834
Your list is now
1. 1454226834
What would you like to do?
        1.  add an element to end of list
        2.  add an element at a location
        3.  find an element by ISBN number (linear search)
        4.  find an element by ISBN number (binary search)
        5.  delete an element at position
        6.  delete an element by ISBN number
        7.  sort the list (using selection sort)
        8.  sort the list (using bubble sort)
        9.  print the list
        0.  exit
your choice ---
2
Please type in the element
2454326534
At what position?
1
Your list is now
1. 2454326534
2. 1454226834
What would you like to do?
        1.  add an element to end of list
        2.  add an element at a location
        3.  find an element by ISBN number (linear search)
        4.  find an element by ISBN number (binary search)
        5.  delete an element at position
        6.  delete an element by ISBN number
        7.  sort the list (using selection sort)
        8.  sort the list (using bubble sort)
        9.  print the list
        0.  exit
your choice ---
6
Which element would you like to delete (by ISBN number)?
1454226834
Your list is now
1. 2454326534
What would you like to do?
        1.  add an element to end of list
        2.  add an element at a location
        3.  find an element by ISBN number (linear search)
        4.  find an element by ISBN number (binary search)
        5.  delete an element at position
        6.  delete an element by ISBN number
        7.  sort the list (using selection sort)
        8.  sort the list (using bubble sort)
        9.  print the list
        0.  exit
your choice ---
```