2413 Assignment 4

Due Date
2-22-16 11:59pm

For this assignment you will be implementing both a bubble sort and an insertion sort on doubly linked lists. I recommend that you define the doubly linked list structures and associated functions in its own header so that main and the implementation of those functions can remain separate. Create any and all auxiliary/helper functions you need or want.

Input Pattern:

```
Num_0 Num_1 Num_2 Num_3 ... Num_(Num_0) SortRoutine
```

Sample Input:

```
5 10 8 2 4 6 bubblesort
```

Sample Input:

```
10 10 8 2 4 6 5 3 1 9 7 insertionsort
```

The first integer is how many integers to read in the sequence.

Sample Input:

```
5 10 8 2 4 6 bubblesort
10 10 8 2 4 6 5 3 1 9 7 insertionsort
```

Your input should handle reading in a sequence of sort problems until end of file is reached.

Main Loop PseudoCode

1. While able to input a number D
   (a) Read D numbers and put each number at the back of a doubly linked list .

(b) Read the name of the sort routine from input.

(c) Print the numbers to output in order on the linked list (same order as input).

(d) perform the sort indicated in the sort routine.

(e) Print the numbers to output in order on the linked list (should now be sorted).

(f) Free the nodes on the linked list.

1 Bubble Sort and Output

Bubble Sort Pseduocode (not handling base cases, edge cases, null pointer cases)

1. set flag "notDone" to true
2. while(notDone)
   (a) set notDone to false
   (b) temp = start of list
   (c) while(temp and temp− >next)
      i. if temp− > e > temp− >next− >e
         A. `printf("Swapping %d %d\n", temp->e, temp->next->e);`
         B. swap temp and temp− >next
      ii. temp = temp− >next
      iii. set notDone to true
   (d) fix head and back pointers of doubly linked list of they moved.

2 Insertion Sort and Output

Insertion Sort Pseduocode (not handling base cases, edge cases, null pointer cases)

1. create a new empty sorted list.
2. while(input list not empty)
   (a) remove node T from input list
   (b) `printf("Removed %d\n", T->e);`
   (c) //Insert node T into new list but in sorted order
   (d) basecase 1:
   (e) inserting on empty list

(f) `printf("Inserting %d On Empty List\n", T->e)`

(g) basecase 2:

(h) inserting at front of list

(i) `printf("Inserting %d At Front Of List\n", T->e)`

(j) basecase 3:

(k) inserting at back of list

(l) `printf("Inserting %d At Back Of List\n", T->e)`

(m) Middle Case:

(n) temp = start of new list

(o) while(T− >e > temp− >next− >e) temp = temp− >next;

(p) `printf("Inserting %d Between %d And %d\n", T->e, temp->e, temp->next->e);`

(q) insert T between temp and temp− >next

3. Make input list point to newlists' nodes.

3 Examples

Example 1 Input To Program:

```
5 10 8 2 4 6 bubblesort
```

Example 1 Output:

```
10 8 2 4 6
Swapping 10 8
Swapping 10 2
Swapping 10 4
Swapping 10 6
Swapping 8 2
Swapping 8 4
Swapping 8 6
2 4 6 8 10
```

Example 2 Input To Program:

```
5 10 8 2 4 6 insertionsort
```

Example 2 Output:

```
10 8 2 4 6
Removed 10
Inserting 10 On Empty List
Removed 8
Inserting 8 At Front Of List
Removed 2
Inserting 2 At Front Of List
Removed 4
Inserting 4 Between 2 And 8
Removed 6
Inserting 6 Between 4 And 8
2 4 6 8 10
```

Example 3 Input To Program:

```
5 10 8 2 4 6 bubblesort
5 10 8 2 4 6 insertionsort
```

Example 3 Output:

```
10 8 2 4 6
Swapping 10 8
Swapping 10 2
Swapping 10 4
Swapping 10 6
Swapping 8 2
Swapping 8 4
Swapping 8 6
2 4 6 8 10
10 8 2 4 6
Removed 10
Inserting 10 On Empty List
Removed 8
Inserting 8 At Front Of List
Removed 2
Inserting 2 At Front Of List
Removed 4
Inserting 4 Between 2 And 8
Removed 6
Inserting 6 Between 4 And 8
2 4 6 8 10
```