Marlex's DSA Exercise

Conversions

Problem Description:

Featured ADTs: List, Queue, Dictionary.

A list of aircraft records is initially represented in internal memory using array implementation. Then it will be converted to cursor-based implementation and sorted in ascending order by their ID.

The list of aircraft records represented using cursor-based implementation is then converted to a queue of aircraft records which is represented in internal memory using linked list implementation.

The queue is converted to a Closed Hash Dictionary. To minimize displacement, a 2-pass loading is implemented, i.e. synonyms are temporarily stored in an array implemented List. In the 2nd round of insertions, synonyms stored in the List are added to the Close Hash Dictionary.

INSTRUCTIONS:

- **Complete the functions according to their specification.**
- **Make a program that will implement and CALL all the functions.**
- **If the function's return type is not void, only 1 return statement only.**
- **No break or continue statement.**
- **Make your code efficient and readable.**
- **Use this Exercise as a practice for DSA. Try not to use any tools aside from your IDE.**
- **Have fun.**

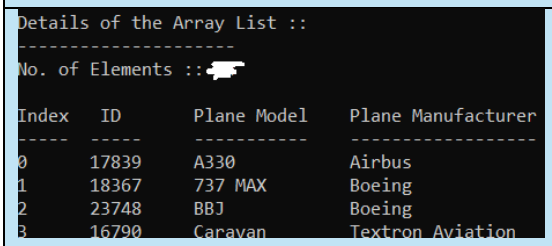
Function Prototypes	Description
Problem #1: Initializes and displays the virtual heap, cursor list and array list	
void initVHeap(VHeap *VH);	The function initializes the virtual heap by linking the nodes and making the last index of the array as the available node. Also initializes the ID to be equal to 4 spaces.
CursorList initCursorList(VHeap *VH);	Given a virtual heap, the function initializes and returns a cursor-based implementation of list. The list contains a pointer to an initialized or existing virtual heap.

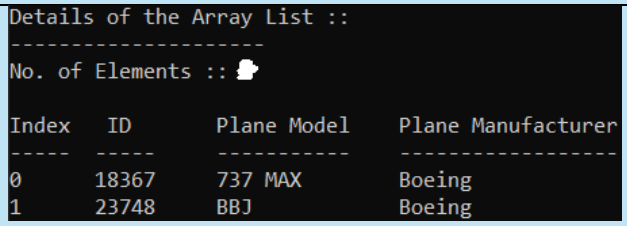
ArrayList initArrayList();	**The function initializes and returns a list implemented using an array.**
void displayVHeap(VHeap V);	**Partially Coded. The function displays the indexes and the next field values of the virtual heap. Included in the display are the values of the available cell and the address of the virtual heap in internal memory. Given below is the expected output. Note: VHeap Address may differ.**
void displayArrayList(ArrayList A);	**Partially Coded. The function displays the contents of the array list. The contents include: the number of elements, the index of elements, the plane's ID, model and manufacturer. Details of each Aircraft are displayed HORIZONTALLY.**
void displayCursorList(CursorList C);	**Partially Coded. The function displays the contents of the cursor list. The contents include: the plane's ID, model, manufacturer, its index in the VHeap and the number of elements. Details of each Aircraft are displayed HORIZONTALLY.**

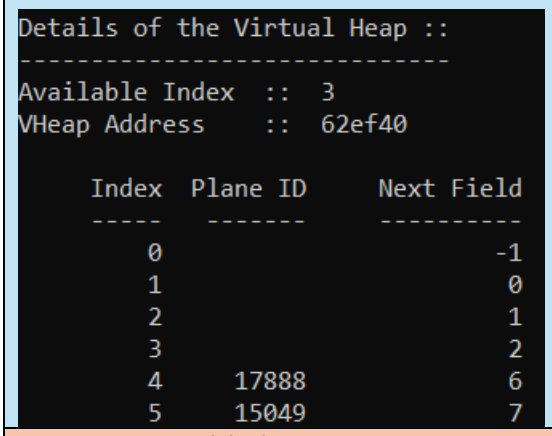
Problem 1 Expected Output

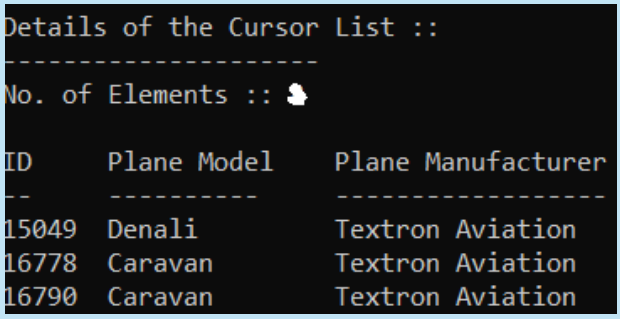
```
Problem #1::
------------

Details of the Virtual Heap ::
------------------------------
Available Index  ::  11
VHeap Address    ::  62ef40

     Index  Plane ID     Next Field
     -----   -------     ----------
         0                       -1
         1                        0
         2                        1
         3                        2
         4                        3
         5                        4
         6                        5
         7                        6
         8                        7
         9                        8
        10                        9
        11                       10
```

```
Details of the Array List ::
--------------------
No. of Elements :: 0

Index   ID       Plane Model    Plane Manufacturer
-----  -----     -----------    ------------------

Details of the Cursor List ::
--------------------
No. of Elements :: 0

ID     Plane Model    Plane Manufacturer       VHeap Index
--     ----------     ------------------       -----------
```

<table>
<tr><td colspan="2">Problem #2: Populates and displays the array list. Then transfers the data to the cursor list sorted in ascending order according to ID from a given array list and displays its contents.</td></tr>
<tr><td>void populateArrayList(ArrayList *A);</td><td>Partially Coded. The function populates the list with list elements provided in the function.</td></tr>
<tr><td>int populateCListSorted(ArrayList *A, CursorList *C);</td><td>Given an array list and a cursor list, the function deletes/removes the elements from the array list and transfers it to the cursor list if the manufacturer of the element is Airbus or Textron Aviation. The function calls allocSpace() for insertion. The function also returns the number of elements inserted to the cursor list.</td></tr>
<tr><td>int allocSpace(VHeap *VH);</td><td>The function removes the first available cell or node in the virtual heap and returns the index of the removed cell or node to the calling function. This is equivalent to the malloc() function in C.</td></tr>
<tr><td colspan="2">Problem 2 Expected Partial Output</td></tr>
<tr><td colspan="2">








</td></tr>
<tr><td colspan="2">Problem #3: Initializes the queue and transfers the data from the cursor list to a queue. Also displays the empty cursor list and the virtual heap.</td></tr>
<tr><td>void initQueue(LinkedQueue *Q);</td><td>Given a queue implemented using linked list, the function initializes the queue to be empty.</td></tr>
</table>

void freeSpace(VHeap *VH, int ndx);	**The function returns back to the heap the node whose index is given by the variable ndx. This is equivalent to the free() function in C.**
void enqueueFromCList(LinkedQueue *Q, CursorList *C);	**Given a queue implemented using linked list and a cursor list, the function deletes/removes all the data from the cursor list and transfers it to the queue. The function calls on freeSpace() for deletion. In addition, for printing purposes, the function also makes the ID equal to a single space.**

Problem 3 Expected Partial Output

```
Details of the Cursor List ::
--------------------
No. of Elements :: 0

ID     Plane Model    Plane Manufacturer       VHeap Index
--     ----------     ------------------       -----------

Details of the Virtual Heap ::
-----------------------------
Available Index  ::  6
VHeap Address    ::  62ef40

     Index  Plane ID     Next Field
     -----   -------     ----------
         0                       -1
         1                        0
         2                        1
         3                        2
         4                       11
```

Problem #4: Converts the queue to a closed hash dictionary and displays its contents.

int closeHash(char ID[]);	**This function returns the hash value of a given ID number by adding its NUMERIC digits and reducing its value appropriate to the size of the hash table. Example: ID numbers "13567" and "17019" have hash values of 0 and 7 respectively.**
void initCloseDic(CloseDic CD);	**The function initializes the close dictionary to be empty using the ID field.**
Aircraft dequeue(LinkedQueue *Q);	**The function deletes the frontmost Aircraft data and returns it to the calling function.**

CloseDic *convertToCloseDic(LinkedQueue *Q);	**The function converts the given queue into a closed hash dictionary using a 2 pass loading (synonyms are temporarily stored in an array implemented list). In the 2nd round of insertions, the elements stored in the list are added in the close dictionary. NOTE: The elements of the close dictionary MUST be UNIQUE. The Aircraft is uniquely identified by their ID.**
void displayCloseDic(CloseDic CD);	**Partially Coded. The function displays the contents of the close hash dictionary.**

Problem 4 Expected Partial Output

```
Details of Closed Hash Dictionary ::
-----------------------------------

Index PlaneID   Plane Model
----- -------   ----------
0     emp
1     16790     Caravan
2     emp
3     emp
```