// =======================

// Included: HW 4

// =======================

// HW 4

// =======================

// Christian Falucho

// CMPR 121

// =======================

#include <iostream>

#include <iomanip>

#include “Box.h”

using namespace std;

/\*================ CONTENTS FROM Box.h ===================

struct Box{

int ID;

double width;

double height;

double length;

Box\* next;

};

Box\* head;

================ CONTENTS FROM Box.h ===================\*/

/\*================ GLOBAL VARIABLES ===================\*/

void insertBox();

void displayBox();

void deleteBox();

void deleteConfirm(int, Box\*&);

/\*================ GLOBAL VARIABLES ===================\*/

/\*

===========================================================

=== MAIN FUNCTION BEGINS ===

===========================================================

\*/

int main(){

Box\* box = new Box;

char answer;

cout << "Enter the specifications of different types of boxes." << endl;

cout << "Include the number of boxes presently in inventory." << endl << endl;

insertBox();

cout << endl;

cout << "Display the list? (Y/N) ";

cin >> answer;

cout << endl << endl;

if (toupper(answer) == 'Y')

{

system("clear");

displayBox();

}

cout << "Do you want to delete a box? (Y/N) ";

cin >> answer;

while (toupper(answer) == 'Y')

{

deleteBox();

cout << "List after deletion" << endl;

displayBox();

cout << "Do you want to delete another? (Y/N) ";

cin >> answer;

}

cout << endl;

displayBox();

return 0;

}

/\*

===========================================================

=== MAIN FUNCTION ENDS ===

===========================================================

\*/

/\*

===========================================================

=== CODE OUTPUT ===

===========================================================

\*/

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

/\*

===========================================================

=== CODE OUTPUT ===

===========================================================

\*/

/\*

===========================================================

=== HELPER FUNCTION BEGINS ===

===========================================================

\*/

// Create a new node

// Input values into node

// Assign head to the next node

// Assign box to be the head

void insertBox(){

char answer;

while (toupper(answer) != 'N')

{

Box\* box = new Box;

cout << "Enter ID: ";

while (!(cin >> box->ID))

{

cout << "Try again. Enter ID: ";

cin.clear();

cin.ignore();

}

cout << "Enter width: ";

while (!(cin >> box->width))

{

cout << "Try again. Enter width: ";

cin.clear();

cin.ignore();

}

cout << "Enter height: ";

while (!(cin >> box->height))

{

cout << "Try again. Enter height: ";

cin.clear();

cin.ignore();

}

cout << "Enter length: ";

while (!(cin >> box->length))

{

cout << "Try again. Enter length: ";

cin.clear();

cin.ignore();

}

box->next = head;

head = box;

cout << "Add another box? (Y/N) ";

cin >> answer;

}

};

// Loop through the node list

void displayBox(){

Box\* nodePtr = head;

cout << " ------------------------------------------- " << endl;

cout << "| " << "Types of boxes: " << setw(27) << "|" << endl;

cout << "|" << "-------------------------------------------" << "|" << endl;

cout << "| " << "ID# " << setw(10) << "Width" << setw(10) << "Height"

<< setw(10) << "Length" << setw(8) << "|" << endl;

cout << "|" << "-------------------------------------------" << "|" << endl;

while (nodePtr != nullptr)

{

cout << "| " << nodePtr->ID << setw(10) << nodePtr->width << setw(10)

<< nodePtr->height << setw(10) << nodePtr->length << setw(10) << " |" << endl;

nodePtr = nodePtr->next;

}

cout << " ------------------------------------------- " << endl;

cout << endl;

}

// Create two pointers

// One pointer (lead pointer) traverse the list

// Second pointer points to the node before the leading pointer

void deleteBox(){

Box\* previousNode;

Box\* nodePtr; // Traverse the list

char answer;

int iD\_To\_Delete;

cout << "Enter the ID # of the box to be deleted: ";

cin >> iD\_To\_Delete;

cout << endl;

// if the node is the first node to delete

if (head->ID == iD\_To\_Delete)

{

nodePtr = head->next;

delete head;

head = nodePtr;

deleteConfirm(iD\_To\_Delete, nodePtr);

}

else{

// initialize node pointer to head

nodePtr = head;

// Loop through each node

while (nodePtr != nullptr && nodePtr->ID != iD\_To\_Delete)

{

previousNode = nodePtr;

nodePtr = nodePtr->next;

}

// When lead pointer finds the node to delete

// Point the previous node pointer to the pointer ahead to

// then delete the node

if (nodePtr->ID == iD\_To\_Delete)

{

previousNode->next = nodePtr->next;

deleteConfirm(iD\_To\_Delete, nodePtr);

}

}

cout << endl;

cout << "!! Box ID #" << iD\_To\_Delete << " has been DELETED !!" << endl << endl;

}

void deleteConfirm(int delete\_ID, Box\*& nodePtr){

char answer;

cout << "Are you sure you want to delete ID #" << delete\_ID << "? (Y/N) ";

cin >> answer;

if (toupper(answer) == 'Y')

{

delete nodePtr;

}

}

/\*

===========================================================

=== HELPER FUNCTION ENDS ===

===========================================================

\*/