Nicholas Krisulevicz

COSC 220

Project 1

10/15/2021

Project 1: Doubly and Singly Linked Lists Database for Students’ Books Rented

Files:

main.cpp

booknode.h

booknode.cpp

student.h

student.cpp

Makefile

StudentDlist (executable)

readme.txt

**Source Code: 5 files: booknode.h, booknode.cpp, student.h, student.cpp, main.cpp**

//Nick Krisulevicz

//COSC 220 - Project 1

//booknode.h

//10/15/2021

#ifndef BOOKNODE\_H

#define BOOKNODE\_H

#include <iostream>

#include <string>

using namespace std;

class booknode{ //class for the book objects to go in the list in the student objects

    public:

        string booktitle; //member variable with the title of the book

        string duedate; //member variable with the due date of the book

        string renewable; //member variable to specify if the book is renewable or not

        booknode \* nextbook; //pointer to next book in the list

        booknode \* bookhead = nullptr; //pointer to head of list

        booknode(); //constructor for the book list

        ~booknode(); //destructor for the book list

        void addbook(string t, string d, string r); //add a book to the list of books the student has

        void removebook(string bt); //remove a book to the list of books the student has

        void printbooklist(); // function to print the list of books

        friend class student; //sets the student class as a friend class so it can access the member function of the book class

};

#endif

//Nick Krisulevicz

//COSC 220 - Project 1

//booknode.cpp

//10/15/2021

#include "booknode.h"

#include <iostream>

#include <string>

using namespace std;

booknode::booknode(){ //constructor for booknode objects

    /\*cout << "Enter the title of the book" << endl;

    cin >> booktitle;

    cout << "Enter the due date of the book" << endl;

    cin >> duedate;

    cout << "Is the lending of the book renewable? (Enter y/n)" << endl;

    cin >> renewable;

    nextbook = nullptr;\*/

    bookhead = nullptr;

}

booknode::~booknode(){ //destructor for booknode objects

    booknode \* bookptr;

    booknode \* nextbooknode;

    bookptr = bookhead;

    while(bookptr != nullptr){ //while loop to traverse through list and delete all nodes

        nextbooknode = bookptr->nextbook;

        delete bookptr;

        bookptr = nextbooknode;

    }

}

void booknode::addbook(string t, string d, string r){

    booknode \*newbook;

    booknode \*bookptr;

    newbook = new booknode();

    newbook->booktitle = t; //assign values to the member variables

    newbook->duedate = d;

    newbook->renewable = r;

    newbook->nextbook = nullptr;

    if(!bookhead){

        bookhead = newbook; //if there is no head, make the first book inserted point to head

    }

    else{

        bookptr = bookhead; //if there are books in the list, point the new book to the last book

        while(bookptr->nextbook){

            bookptr = bookptr->nextbook; //while loop to proceed through the list until it finds the end

        }

        bookptr->nextbook = newbook; //insert the new book as the last book in the list

    }

}

void booknode::removebook(string bt){

    //search algorithm to compare bt to booktitle

    //if bt does not equal book title, iterate to the next node in the list

    //if bt does equal book title, remove that node, and point previous to currbook->next

    booknode \* prevbook;

    booknode \* bookptr;

    if(!bookhead){ //if list is empty, nothing will be deleted

        return;

    }

    if(bookhead->booktitle == bt){ //if the first booknode is the target one, it will be deleted

        bookptr = bookhead->nextbook;

        delete bookhead;

        bookhead = bookptr;

    }

    else{ //if the target booknode is not the first one, then iterate through the list until the target node is found and delete it

        bookptr = bookhead;

        while(bookptr != nullptr && bookptr->booktitle != bt){ //while loop to proceed through list

            prevbook = bookptr;

            bookptr = bookptr->nextbook;

        }

        if(bookptr){

            prevbook->nextbook = bookptr->nextbook;

            delete bookptr; //bookptr is a temporary pointer to traverse through the list, so delete it at the end of the function

        }

        else{

        cout << "Book not found" << endl;

        }

    }

}

void booknode::printbooklist(){

    booknode \*bookptr;

    bookptr = bookhead;

    while(bookptr){ //iterate through list and print all books in the list

        cout << bookptr->booktitle << ", ";

        cout << bookptr->duedate << ", ";

        cout << bookptr->renewable << endl;

        bookptr = bookptr->nextbook;

    }

}

//Nick Krisulevicz

//COSC 220 - Project 1

//student.h

//10//15/2021

#ifndef STUDENT\_H

#define STUDENT\_H

#include "booknode.h"

#include <iostream>

#include <string>

using namespace std;

class student{ //class for the student objects

    public:

        string firstname; //member variable for first name

        string lastname; //member variable for last name

        string year; //member variable fir student's year

        int idnumber; //member variable for student id number

        booknode listbooks; //member object of booknode class for the list of books the student has rented

        student \* head = nullptr; //initialize the head of the list to nullptr

        student \* next; //pointer to next node in the doubly linked list

        student \* prev; //pointer to previous node in the doubly linked list

        student(); //constructor for the list

        ~student(); //destructor for the list

        void addstudent(string fn, string ln, string yr, int idn); //member function to add a student node to the list

        void updatestudent(int idn); //member function to search for and update an existing student node

        void deletestudent(int idn); //member function to search for and delete a student node

        void addbooks(int idn); //member function that adds list of book nodes to student node with corresponding id number

        void removebooks(int idn); //member function to that removes books from a student node with corrresponding id number

        void printlist(); //member function to print all of the student nodes in the list

};

#endif

//Nick Krisulevicz

//COSC 220 - Project 1

//student.cpp

//10/15/2021

#include "student.h"

#include "booknode.h"

#include <iostream>

#include <string>

using namespace std;

student::student(){

    head = nullptr; //constructor sets head to nullptr

}

student::~student(){

    student \* nodeptr;

    student \* nextnode;

    nodeptr = head;

    while(nodeptr != nullptr){ //while loop to traverse through list and delete all nodes

        nextnode = nodeptr->next;

        delete nodeptr; //destructor deletes the student nodes one by one

        nodeptr = nextnode;

    }

}

void student::addstudent(string fn, string ln, string yr, int idn){

    student \* newstudent;

    newstudent = new student(); //dynamically allocate the new student node to fill with data before it is inserted

    newstudent->firstname = fn; //assign the values to the member variables

    newstudent->lastname = ln;

    newstudent->year = yr;

    newstudent->idnumber = idn;

    newstudent->next = nullptr; //initially set the prev and next pointers to nullptr before they get assigned their spot in the list

    newstudent->prev = nullptr;

    if(!head){ //conditional for if head does not exist, make this node the head

        head = newstudent;

        return;

    }

    student \* curr = head; //declare the current and previous cursor pointers to iterate through the list

    student \* previous = head;

    while(curr->next){ //conditional for while there is a curr->next, iterate through the list

        previous = curr;

        curr = curr->next;

        if(newstudent->idnumber > previous->idnumber && newstudent->idnumber < curr->idnumber){ //if the idnumber is between current and previous, insert the node here

            newstudent->prev = previous;

            newstudent->next = curr;

            return;

        }

    }

    if(!curr->next){

        newstudent->prev = curr;

        curr->next = newstudent;

    }

}

void student::updatestudent(int idn){

    if(!head){ //if there are no students in the list, print out a message saying there are no students to update

        cout << "There is no student to update!" << endl;

        return;

    }

    string fn;

    string ln;

    string yr;

    cout << "Enter the first name" << endl;

    cin >> fn;

    cout << "Enter the last name" << endl;

    cin >> ln;

    cout << "Enter the student's year" << endl;

    cin >> yr;

    student \* replacement; //declare a node to replace the student being updated

    replacement = new student();

    replacement->firstname = fn; //assign the values from the parameters into the student node

    replacement->lastname = ln;

    replacement->year = yr;

    replacement->idnumber = idn;

    replacement->next = nullptr;

    replacement->prev = nullptr;

    student \* curr = head; //create the current and previous cursor pointers

    student \* previous = head;

    if(head->idnumber == idn){ //if the head is the target node, replace the head with the updated node

        replacement->prev = nullptr;

        replacement->next = head->next;

        head = replacement;

        delete replacement;

        return;

    }

    while(curr->next){ //conditional statement to iterate through list of student nodes

        previous = curr;

        curr = curr->next;

        if(replacement->idnumber == curr->idnumber){ //conditional if the id number is equal to the id number in a node, replace the existing node with the updated one

            replacement->prev = previous;

            replacement->next = curr->next;

            curr = replacement; //assign the value of the replacement node into the old one

            delete replacement; //free memory where the temporary replacement node was

            return;

        }

    }

    if(!curr->next){

        cout << "End of list and student with this id number was not found" << endl; //print message if student node with matching id number is not found

    }

}

void student::deletestudent(int idn){

    student \* previous;

    student \* curr;

    if(!head){

        cout << "There are no nodes to delete" << endl; //print out message if there are no student nodes to delete

    }

    previous = head;

    curr = head;

    if(head->idnumber == idn){ //if the target node to delete is the head of the list, delete the head

        curr = head->next;

        previous = nullptr;

        delete head;

        head = curr;

    }

    else{ //if the target node is not the head, iterate through the list to find the target node

        curr = head;

        while(curr->next != nullptr && curr->idnumber != idn){

            previous = curr;

            curr = curr->next;

        }

        if(curr){ //delete the target node and set the pointers accordingly

            previous->next = curr->next;

            curr->next->prev = previous;

            delete curr;

        }

    }

}

void student::addbooks(int idn){

    student \* previous;

    student \* curr;

    string ti;

    string du;

    string re;

    cout << "Enter the book title" << endl;

    cin >> ti;

    cout << "Enter the due date" << endl;

    cin >> du;

    cout << "Enter if it is renewable (y/n)" << endl;

    cin >> re;

    if(!head){ //if there is no head, print a message saying that there is no student to add books to

        cout << "There is not student to add books to" << endl;

        return;

    }

    previous = head;

    curr = head;

    if(head->idnumber == idn){ //if the head of the student list is the target one, add a book to this student

        head->listbooks.addbook(ti, du, re);

    return;

    }

    while(curr->next != nullptr || curr->idnumber == idn){ //if the head is not the target student, iterate through the list until the target is found

        previous = curr;

        curr = curr->next;

        if(curr->idnumber == idn){

            curr->listbooks.addbook(ti, du, re); //when the target is found, add a book to that student

        return;

        }

    }

    if(!curr->next){ //print out message if the target student was not found

        cout << "End of list and student with this ID number was not found" << endl;

    }

}

void student:: removebooks(int idn){

    student \* previous;

    student \* curr;

    string ti;

    cout << "Enter the book title to be deleted" << endl;

    cin >> ti;

    if(!head){ //if there is no head, print a message saying that there is no student to add books to

        cout << "There is not student to remove books from" << endl;

    }

    previous = head;

    curr = head;

    if(head->idnumber == idn){ //if the head of the student list is the target one, delete a book from this student

        head->listbooks.removebook(ti);

    }

    while(head->next != nullptr && head->idnumber == idn){ //if the head is not the target student, iterate through the list until the target is found

        previous = curr;

        curr = curr->next;

        if(curr->idnumber == idn){

            curr->listbooks.removebook(ti); //when the target is found, delete the book from that student

        }

    }

    if(!curr->next){ //print out message if the target student was not found

        cout << "End of list and student with this ID number was not found" << endl;

    }

}

void student::printlist(){

    student \* curr;

    student \* previous;

    curr = head;

    while(curr){ //iterate through the whole list and print out the data for all the students

        cout << curr->firstname << " ";

        cout << curr->lastname << ", ";

        cout << curr->year << endl;

        cout << "Books rented: " << endl;

        curr->listbooks.printbooklist();

    if(!curr->next){

        break;

    }

    curr=curr->next;

    }

}

//Nick Krisulevicz

//COSC 220 - Project 1

//main.cpp

//10/15/2021

#include "booknode.h"

#include "student.h"

#include <iostream>

#include <string>

using namespace std;

int main(){

    string fname;

    string lname;

    string studyear;

    int idnum;

    int searchid;

    int userchoice = 0;

    student studlist;

    while(userchoice != 7){

        cout << "Input:" << userchoice << endl;

        cout << "Welcome to the student library database!" << endl;

        cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        cout << "Please select and option below:" << endl << endl;

        cout << "1. Add a student" << endl;

        cout << "2. Update an existing student's records" << endl;

        cout << "3. Delete a student's records" << endl;

        cout << "4. Check out a book for a student" << endl;

        cout << "5. Return a book from a student" << endl;

        cout << "6. Print out the list of students" << endl;

        cout << "7. Exit the program" << endl;

        cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        cin >> userchoice;

        cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        if(userchoice == 1){

            cout << "Enter the student's first name" << endl;

            cin >> fname;

            cout << "Enter the student's last name" << endl;

            cin >> lname;

            cout << "Enter the student's year of school" << endl;

            cin >> studyear;

            cout << "Enter the student's ID number" << endl;

            cin >> idnum;

            studlist.addstudent(fname, lname, studyear, idnum);

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else if(userchoice == 2){

            cout << "Enter the ID number of the student to update" << endl;

            cin >> searchid;

            studlist.updatestudent(searchid);

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else if(userchoice == 3){

            cout << "Enter the ID number of the student to remove" << endl;

            cin >> searchid;

            studlist.deletestudent(searchid);

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else if(userchoice == 4){

            cout << "Enter the ID number of the student to check out a book" << endl;

            cin >> searchid;

            studlist.addbooks(searchid);

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else if(userchoice == 5){

            cout << "Enter the ID number of the student to return a book" << endl;

            cin >> searchid;

            studlist.removebooks(searchid);

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else if(userchoice == 6){

            studlist.printlist();

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else if(userchoice == 7){

            cout << "Goodbye!" << endl;

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

        else{

            cout << "Invalid input! Please enter one of the following menu choices." << endl;

            cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

        }

    }

    return 0;

}

**Outputs:**

Case 1: run Makefile

Text

Description automatically generated

Case 2: add one student and one book and print list

Input:0 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the student's first name Nick Enter the student's last name Krisulevicz Enter the student's year of school senior Enter the student's ID number 3086 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:1 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to check out a book 3086 Enter the book title Encyclopedia Enter the due date 10/31/2021 Enter if it is renewable (y/n) y \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:4 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Nick Krisulevicz, senior Books rented: Encyclopedia, 10/31/2021, y \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:6 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Goodbye! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Case 3: add two students and two books and print the list

Input:0 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the student's first name Grant Enter the student's last name Dawson Enter the student's year of school senior Enter the student's ID number 3071 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:1 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the student's first name Jef Enter the student's last name Geff Enter the student's year of school junior Enter the student's ID number 3076 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:1 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to check out a book 3076 Enter the book title Goober Enter the due date 10/20/2021 Enter if it is renewable (y/n) n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:4 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to check out a book 3071 Enter the book title WildernessSurvival Enter the due date 10/30/2021 Enter if it is renewable (y/n) y \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:4 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Grant Dawson, senior Books rented: WildernessSurvival, 10/30/2021, y Jef Geff, junior Books rented: Goober, 10/20/2021, n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:6 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Goodbye! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Case 4: add one student and two books and print the list

Input:0 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the student's first name Dr Enter the student's last name Park Enter the student's year of school graduated Enter the student's ID number 2808 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:1 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to check out a book 2808 Enter the book title COSC220Textbook Enter the due date 12/31/2021 Enter if it is renewable (y/n) y \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:4 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to check out a book 2808 Enter the book title Cookbook Enter the due date 01/15/2022 Enter if it is renewable (y/n) n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:4 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Dr Park, graduated Books rented: COSC220Textbook, 12/31/2021, y Cookbook, 01/15/2022, n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:6 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Goodbye! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Case 5: add two students and delete one student and print list

Input:0 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the student's first name Mike Enter the student's last name Sauer Enter the student's year of school senior Enter the student's ID number 3090 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:1 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the student's first name Andrew Enter the student's last name Metz Enter the student's year of school dropout Enter the student's ID number 3088 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:1 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to remove 3090 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:3 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Andrew Metz, dropout Books rented: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:6 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Goodbye! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Case 6: adding book when there is no student

Input:0 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to check out a book 3000 Enter the book title goober Enter the due date 10/31/2021 Enter if it is renewable (y/n) y There is not student to add books to \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Input:4 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Goodbye! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Case 7: removing student when there is no student in the list

Input:0 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enter the ID number of the student to remove 3086 There are no nodes to delete

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Input:3 Welcome to the student library database! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Please select and option below: 1. Add a student 2. Update an existing student's records 3. Delete a student's records 4. Check out a book for a student 5. Return a book from a student 6. Print out the list of students 7. Exit the program \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Goodbye! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Note:**

Extra credit was not attempted.

**Readme.txt**

readme.txt

Program by: Nicholas Krisulevicz

Welcome to the StudentDlist application!

This program allows the user to manage a library database. The user has complete control over the books that are checked out. The students/ information such as name, year of school, and ID number can be stored, as well as a list of all the books they have currently checked out. The program works via a simple numbered list menu where the user is prompted different actions to take to manage the database. The data structure used is singly and doubly linked lists which allows for efficient processes to add and remove data from the database.

Enjoy your new program to manage the library!