Data Structures- Assignment 2

For: Dr. Ramazan Aygün

By: Erik Failing

2/24/2019

Table of Contents

1 Title Page

2 Table of Contents

3 System Overview, Referenced Documents, Concept of Execution and Abstract Data Type

4 - 5 Code Outline

6 - 11 Detailed Design

12- 13 Test Plan, Test Procedures, Sample Runs

Section I.1. System Overview

This program handles the reading, organization and output of users in some database using linked lists.

Section I.1.1. Referenced Documents

No references were made.

Section I.1.2. Concept of Execution

A database of users will be read in from a text file and be outputted to the console and another text file.

Section I.1.3. Abstract Data Type

The Linked Lists abstract data type was used.

Section I.2. Code Outline

Class: User

void Initialize(const char* afName, const char* alName, char aGender, const char* aMajor, const char* aEmail, AddressType aAddress, float aGPA, DateType aDateOfBirth);

void GetFirstName(char afname[]) const;

void GetLastName(char alname[]) const;

void GetMajor(char amajor[]) const;

void GetEmail(char aemail[]) const;

void GetGender(char& agender) const;

DateType GetDateOfBirth() const;

void GetDateOfBirth(DateType& aDateOfBirth) const;

float GetGPA() const;

void GetGPA(float& aGPA) const;

AddressType GetAddress() const;

void GetAddress(AddressType& aAddress) const;

void GetAddress(char aStreetName[], int& aStreetNo, char aCity[], int& aZip, char aState[]) const;

void SetFirstName(const char* afName);

void SetLastName(const char* alName);

void SetMajor(const char* amajor);

void SetEmail(const char* anemail);

void SetGender(char aGender);

void SetDateOfBirth(DateType aDateOfBirth);

void SetGPA(float aGPA);

void SetAddress(AddressType aAddress);

void SetAddress(char aStreetName[], int aStreetNo, char aCity[], int aZip, char aState[]);

void Display() const;

void Display(std::ofstream& outFile) const;

RelationType comparedTo(User* aUser) const;

Class: Test Driver

int Populate(const char* input, User users[]);

int Populate(const char* input, UnsortedType& allUnsortedUsers) const;

int Populate(const char* input, SortedType& allSortedUsers) const;

void Test(User users[], int count) const;

void Test(std::ofstream& outFile, User users[], int count) const;

void TestList(UnsortedType& allUnsortedUsers) const;

void TestList(std::ofstream& outFile, UnsortedType& allUnsortedUsers) const;

void TestList(SortedType& allSortedUsers) const;

void TestList(std::ofstream& outFile, SortedType& allSortedUsers) const;

Class: UnsortedType

```
void MakeEmpty();
bool IsFull() const;
int GetLength() const;
void ResetList();
User* GetUser(char fname[], char lname[], bool& found) const;
virtual void PutUser(User* user);
void DeleteUser(User* user);
User* GetNextUser();
void Print() const;
void Print(std::ofstream& out) const;
void PrintPartial() const;
void PrintPartial(std::ofstream& out) const;
```

Class: SortedType (inherits from UnsortedType)

```
void PutUser(User* user);
```

Class: DateType

```
void Initialize(int newMonth, int newDay, int newYear);
RelationType comparedTo(DateType aDate) const;
```

Section I.2.1. Detailed Design

Class: User

void Initialize(const char* afName, const char* alName, char aGender, const char* aMajor, const char* aEmail, AddressType aAddress, float aGPA, DateType aDateOfBirth);

Purpose: Creates a new user

Arguments: Char arrays for the user's first name, last name, major and email. A float for GPA. An AddressType for the user's address and DateType for the user's date of birth.

Return Value: Void

void GetFirstName(char afname[]) const;

Purpose: Gets the user's first name

Arguments: A character array that is the user's first name

Return Value: Void

void GetLastName(char alname[]) const;

Purpose: Gets the user's last name.

Arguments: A character array that is the user's last name

Return Value: Void

void GetMajor(char amajor[]) const;

Purpose: Gets the user's major

Arguments: A char array that is the user's major

Return Value: Void

void GetEmail(char aemail[]) const;

Purpose: Gets the user's email

Arguments: A char array that is the user's email

Return Value:

void GetGender(char& agender) const;

Purpose: Gets the user's gender

Arguments: A char that is the user's gender

Return Value: Void

DateType GetDateOfBirth() const;

Purpose: Gets the user's date of birth

Arguments: None

Return Value: Returns a DateType which is three integers representing the user's date of birth

void GetDateOfBirth(DateType& aDateOfBirth) const;

Purpose: Gets the user's date of birth

Arguments: DateType which is three integers representing the user's date of birth

Return Value: Void

float GetGPA() const;

Purpose: Gets the user's GPA

Arguments: None

Return Value: Returns a float which is the user's GPA

void GetGPA(float& aGPA) const;

Purpose: Gets the user's GPA

Arguments: A float which is the user's GPA

Return Value: Void

AddressType GetAddress() const;

Purpose: Gets the user's address

Arguments: None

Return Value: Returns an AddressType which is three char arrays and two integers making up the user's address

void GetAddress(AddressType& aAddress) const;

Purpose: Gets the user's address

Arguments: An AddressType which is three char arrays and two integers making up the user's address

Return Value: Void

void GetAddress(char aStreetName[], int& aStreetNo, char aCity[], int& aZip, char aState[]) const;

Purpose: Gets the user's address

Arguments: Three char arrays and two integers making up the user's address

Return Value: Void

void SetFirstName(const char* afName);

Purpose: Sets the user's first name

Arguments: A char array of the user's new first name

Return Value: Void

void SetLastName(const char* alName);

Purpose: Sets the user's last name

Arguments: A char array of the user's new last name

Return Value: Void

void SetMajor(const char* amajor);

Purpose: Sets the user's major

Arguments: A char array of the user's new major

Return Value: Void

void SetEmail(const char* anemail);

Purpose: Sets the user's email

Arguments: A char array of the user's new email

Return Value: Void

void SetGender(char aGender);

Purpose: Sets the user's gender

Arguments: A char representing the user's gender

Return Value: Void

void SetDateOfBirth(DateType aDateOfBirth);

Purpose: Sets the user's date of birth

Arguments: A DateType of three integers representing the user's new birthday

Return Value: Void

void SetGPA(float aGPA);

Purpose: Sets the user's GPA

Arguments: A float representing the user's new GPA

Return Value: Void

void SetAddress(AddressType aAddress);

Purpose: Sets the user's address

Arguments: An AddressType of three char arrays and two integers representing the user's new address

Return Value: Void

void SetAddress(char aStreetName[], int aStreetNo, char aCity[], int aZip, char aState[]);

Purpose: Sets the user's new address

Arguments: Three char arrays and two integers representing the user's new address

Return Value: Void

void Display() const;

Purpose: Displays all the data from the user to the console

Arguments: None

Return Value: Void

void Display(std::ofstream& outFile) const;

Purpose: Prints all the data of the user into a file

Arguments: An ofstream representing the file being printed to

Return Value: Void

RelationType comparedTo(User* aUser) const;

Purpose: Compares one user to another based on date type

Arguments: A user pointer to the user being compared

Return Value: Returns A RelationType which can either be LESS, GREATER or EQUAL

Class: Test Driver

int Populate(const char* input, User users[]);

Purpose: Reads in all users from a file into an users array

Arguments: A char array representing the file to read from and a users array to store the data into

Return Value: An integer representing the number of users that were read in

int Populate(const char* input, UnsortedType& allUnsortedUsers) const;

Purpose: Reads in all users from a file into an users linked list

Arguments: A char array representing the file to read from and an unsorted linked list to store the data into

Return Value: An integer representing the number of users that were read in

int Populate(const char* input, SortedType& allSortedUsers) const;

Purpose: Reads in all users from a file into an users linked list

Arguments: A char array representing the file to read from and a sorted linked list to store the data into

Return Value: An integer representing the number of users that were read in

void Test(User users[], int count) const;

Purpose: Tests the user class and its helpers class, printing results to the console

Arguments: An users array to test on and an integer representing the number of users in the user array.

Return Value: Void

void Test(std::ofstream& outFile, User users[], int count) const;

Purpose: Tests the user class and its helpers class, printing results to a file

Arguments: An ofstream representing the file being printed to, an users array to test on and an integer representing the number of users in the user array.

Return Value: Void

void TestList(UnsortedType& allUnsortedUsers) const;

Purpose: Tests the unsorted linked list and its helpers classes, printing results to the console

Arguments: An unsorted linked list full of users to run tests on

Return Value: Void

void TestList(std::ofstream& outFile, UnsortedType& allUnsortedUsers) const;

Purpose: Tests the unsorted linked list and its helpers classes, printing results to a file

Arguments: An ofstream representing the file being printed to and an unsorted linked list of users to test on.

Return Value: Void

void TestList(SortedType& allSortedUsers) const;

Purpose: Tests the sorted linked list and its helpers classes, printing results to the console

Arguments: A sorted linked list full of users to run tests on

Return Value: Void

void TestList(std::ofstream& outFile, SortedType& allSortedUsers) const;

Purpose: Tests the sorted linked list and its helpers classes, printing results to a file

Arguments: An ofstream representing the file being printed to and an unsorted linked list of users to test on.

Return Value: Void

Class: UnsortedType

void MakeEmpty();

Purpose: Deallocates the linked list of all its elements

Arguments: None

Return Value: Void

bool IsFull() const;

Purpose: Checks to see if there is free storage for another node on a linked list

Arguments: None

Return Value: A bool that is true if there is no more space to add onto the linked list

int GetLength() const;

Purpose: Gets the length of the linked list

Arguments: None

Return Value: returns an integer representing the length of the linked list

void ResetList();

Purpose: Resets the currentPos pointer to null so iteration can begin again

Arguments: None

Return Value: Void

User* GetUser(char fname[], char lname[], bool& found) const;

Purpose: Gets a user based on their first and last name from a linked list

Arguments: Two char arrays for the user's name and a bool for if the user was found or not.

Return Value: A pointer to the user that was found

virtual void PutUser(User* user);

Purpose: Adds a user to the unsorted linked list

Arguments: A pointer to a user that is added to the unsorted linked list

Return Value: Void

void DeleteUser(User* user);

Purpose: Deletes a user from the linked list

Arguments: A pointer to the user that is going to be deleted from the linked list

Return Value: Void

User* GetNextUser();

Purpose: Gets the next user in the linked list, iterating by one

Arguments: None

Return Value: A pointer to the user that was just iterated to

void Print() const;

Purpose: Prints the current nodes user to the console in the linked list

Arguments: None

Return Value: Void

void Print(std::ofstream& out) const;

Purpose: Prints the current nodes user to the output file in the linked list

Arguments: An ofstream for the user to be printed to

Return Value: Void

void PrintPartial() const;

Purpose: Prints the current node's user's name to the output file in the linked list

Arguments: None

Return Value: Void

void PrintPartial(std::ofstream& out) const;

Purpose: Prints the current node's user's name to the output file in the linked list

Arguments: An ofstream for the user to be printed to

Return Value: Void

Class: SortedType (inherits from UnsortedType)

void PutUser(User* user);

Purpose: Puts the user into the appropriate spot in the sorted linked list of users

Arguments: A pointer to a user that is being inserted into the sorted linked list

Return Value: Void

Class: DateType

void Initialize(int newMonth, int newDay, int newYear);

Purpose: Creates a new datetype object with filled out integer variables

Arguments: Three integers representing the new date's month, day and year.

Return Value: Void

RelationType comparedTo(DateType aDate) const;

Purpose: Compares dates to see if one is greater than the other or if they are equal

Arguments: a DateType to compare the current DateType to

Return Value: A relationtype that is either GREATER, LESS or EQUAL depending on the compared dates

Test Plan

This software is tested by the exhaustive method. Every method is called and every variable is used to confirm functionality.

Section II.1. Test Procedures

Main Testing

•Purpose - To test most of the methods

•Procedure - Reads in users and then displays them

•Inputs - Users

•Expected Outputs - The same Users

•Success Criteria - Inputs match the outputs

Secondary Testing

•Purpose - To test the remaining redundant methods

•Procedure - Alters current users and then displays alterations

•Inputs - Users

•Expected Outputs - The altered Users

•Success Criteria - Outputs are successfully altered and do NOT match the inputs

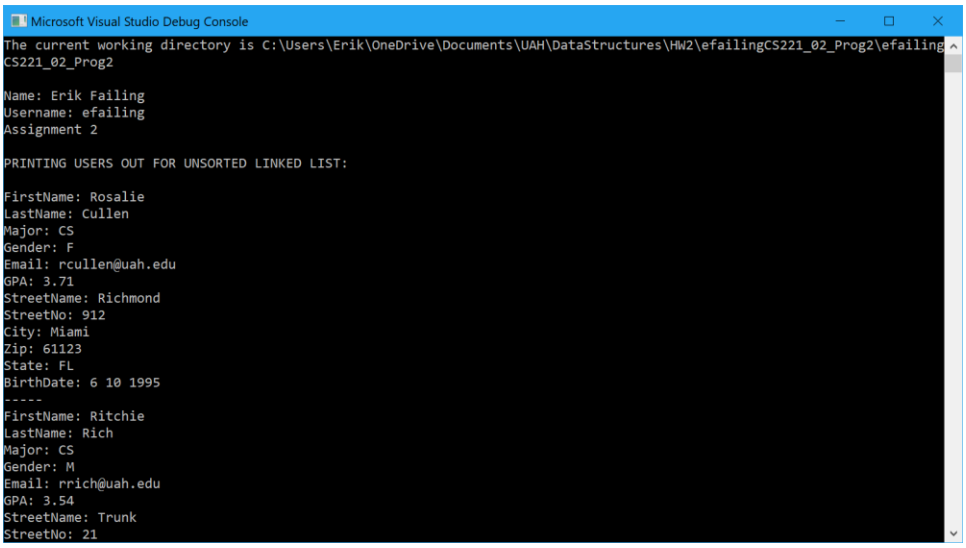
Section II.2. Sample Runs

Sample Run 1:

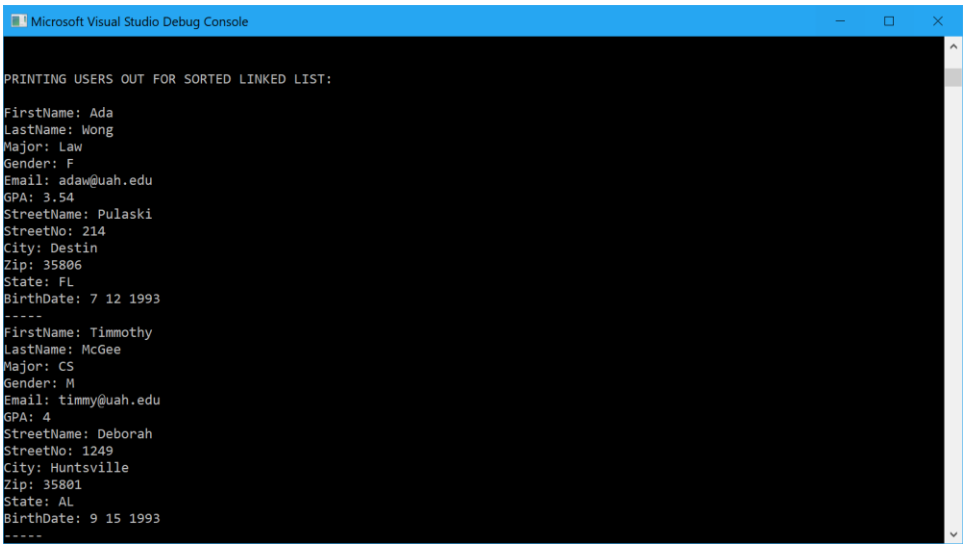
•Inputs - Sample user text file hw1samplefile.txt

•Outputs - efailingHw2Outfile.txt

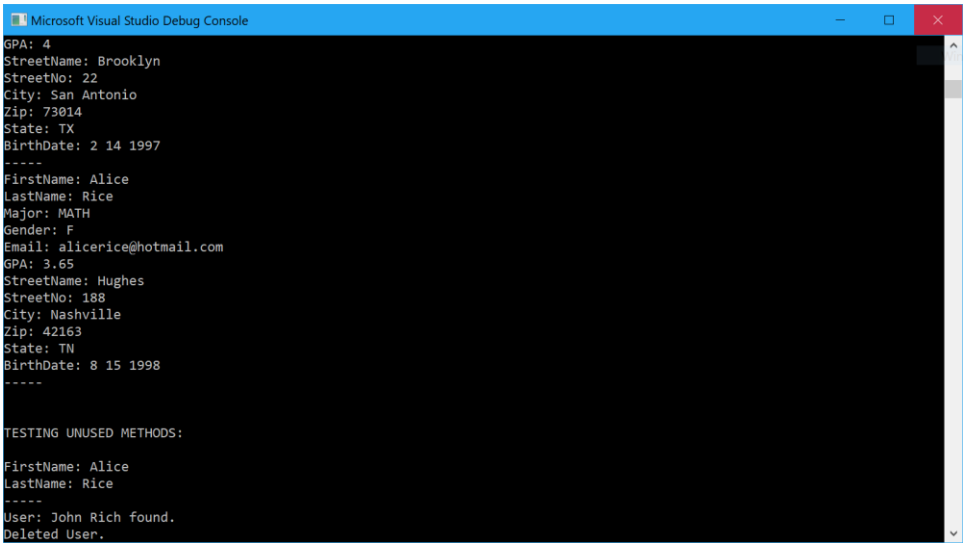
•Snapshots:


Microsoft Visual Studio Debug Console
The current working directory is C:\Users\Erik\OneDrive\Documents\UAH\DataStructures\HW2\efailingCS221_02_Prog2\efailing
CS221_02_Prog2
Name: Erik Failing
Username: efailing
Assignment 2
PRINTING USERS OUT FOR UNSORTED LINKED LIST:
FirstName: Rosalie
LastName: Cullen
Major: CS
Gender: F
Email: rcullen@uah.edu
GPA: 3.71
StreetName: Richmond
StreetNo: 912
City: Miami
Zip: 61123
State: FL
BirthDate: 6 10 1995

FirstName: Ritchie
LastName: Rich
Major: CS
Gender: M
Email: rrich@uah.edu
GPA: 3.54
StreetName: Trunk
StreetNo: 21


Microsoft Visual Studio Debug Console
PRINTING USERS OUT FOR SORTED LINKED LIST:
FirstName: Ada
LastName: Wong
Major: Law
Gender: F
Email: adaw@uah.edu
GPA: 3.54
StreetName: Pulaski
StreetNo: 214
City: Destin
Zip: 35806
State: FL
BirthDate: 7 12 1993

FirstName: Timmothy
LastName: McGee
Major: CS
Gender: M
Email: timmy@uah.edu
GPA: 4
StreetName: Deborah
StreetNo: 1249
City: Huntsville
Zip: 35801
State: AL
BirthDate: 9 15 1993


Microsoft Visual Studio Debug Console
GPA: 4
StreetName: Brooklyn
StreetNo: 22
City: San Antonio
Zip: 73014
State: TX
BirthDate: 2 14 1997

FirstName: Alice
LastName: Rice
Major: MATH
Gender: F
Email: alicerice@hotmail.com
GPA: 3.65
StreetName: Hughes
StreetNo: 188
City: Nashville
Zip: 42163
State: TN
BirthDate: 8 15 1998

TESTING UNUSED METHODS:
FirstName: Alice
LastName: Rice

User: John Rich found.
Deleted User.