Laboratory12A

CS-102

Spring 2022

Laboratory12A: Files Using Data Structure

- Write a program that uses a structure variable to store the data on company divisions into a file called **divisions.dat**:
  - **Division Name** (East, West, North, and South) Should be array of **char**
  - **Quarter** (1, 2, 3, or 4) Should be an **int**
  - **Quarterly Sales** Should be a **double**
  - **City** Should be an **array of char**
- The user may be requested to enter any of four quarters' of sales figures for any of the divisions. This data (a.k.a. "record") for any quarter of any division should be written to a binary file.

Laboratory12A Parts 1&2: Files Using Data Structure

- You will be writing two programs:
  - The first program should be called: *YourName*-Lab12A-1.cpp.
    - This program is similar to Program 12-15, in the text, and records the data given verbally, writing the data, given, into a binary file called: **divisions.dat** .
  - The second program is a companion program which will read in the file produced by the above program (**divisions.dat**), and display it on the screen. Call this program *YourName*-Lab12A-2.cpp.
  - TAKE NOTE: The trickiest part of the assignment is to make sure that the two struct definitions in the two programs are exactly identical. Otherwise, it won't work.
  - When you have each of the two programs up and working, call instructor to demo.
- If you are doing this Lab synchronously, demonstrate both programs to the instructor for credit.
- If you are doing this Lab asynchronously, you will need to write both programs: *YourName*-Lab12A-1.cpp, and *YourName*-Lab12A-2.cpp, submitting both.
- You should find Programs 12-15 and 12-16 very helpful in doing this work.

Laboratory12A Part 3: Counting & Displaying the Words in a Text

- You are given the file Gettysburg.txt, which contains Lincoln's entire Gettysburg address.
- Your first task will be to count all the words in the Gettysburg Address.
- Note: You should not list nor count any non-words (e.g. "--" is not a word).
- Call your program ***YourName*-Lab12A-3.cpp**
- Report your count in a text file, which should be called ***YourName*-Lab12A-3.txt:**
  - If you are doing this Lab synchronously, call the instructor so that you may receive credit for having accomplished this.
  - If you are doing this Lab asynchronously, submit your program and your count in a textfile to Canvas.

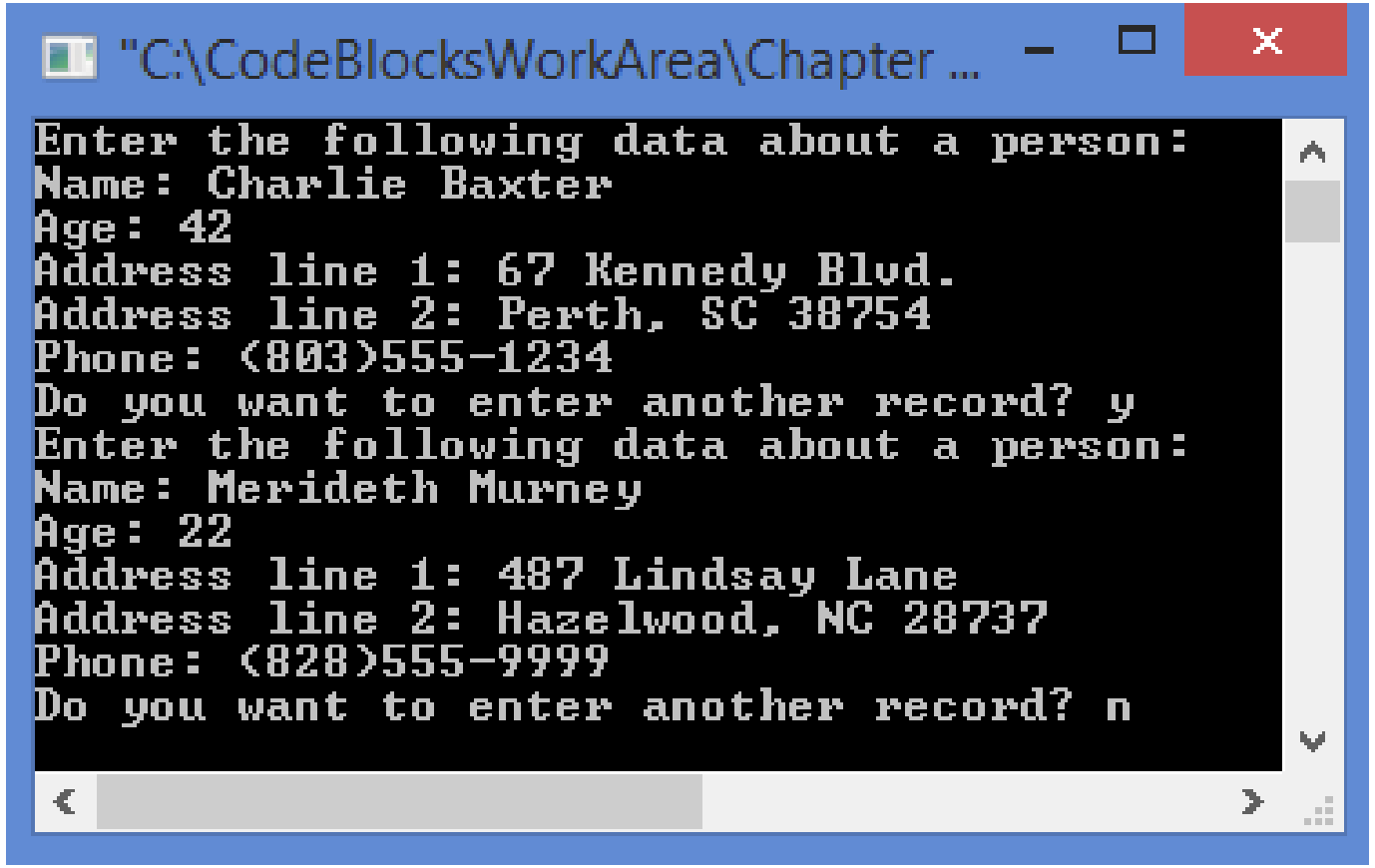
```
// This program uses a structure variable to store a record to a file.
#include<iostream>
#include<fstream>
using namespace std;
// Array sizes
const int NAME_SIZE = 51, ADDR_SIZE = 51, PHONE_SIZE = 14;

// Declare a structure for the record.
struct Info
{
        char name[NAME_SIZE];
        int age;
        char address1[ADDR_SIZE];
        char address2[ADDR_SIZE];
        char phone[PHONE_SIZE];
};

int main()
{
        Info person;        // To hold info about a person
        char again;                 // To hold Y or N

        // Open a file for binary output.
        fstream people("people.dat", ios::out | ios::binary);
```

Program 12-15 Writing a Structure Variable to a File

```
        do
        {
                // Get data about a person.
                cout << "Enter the following data about a “<< "person:\n";
                cout << "Name: ";
                cin.getline(person.name, NAME_SIZE);
                cout << "Age: ";
                cin >> person.age;
                cin.ignore();           // Skip over the remaining newline.
                cout << "Address line 1: ";
                cin.getline(person.address1, ADDR_SIZE);
                cout << "Address line 2: ";
                cin.getline(person.address2, ADDR_SIZE);
                cout << "Phone: ";
                cin.getline(person.phone, PHONE_SIZE);
                people.write(reinterpret_cast<char *>(&person), sizeof(person));
                cout << "Do you want to enter another record? ";
                cin >> again;
                cin.ignore();           // Skip over the remaining newline.
        } while (again == 'Y' || again == 'y');
        // Close the file.
        people.close();
        return 0;
}
```

Program 12-15 Concluded

Program 12-15 Input and Output



Program 12-16 Reading Structure Variables from a file

```
// This program uses a structure variable to read a record from a file.
#include<iostream>
#include<fstream>
using namespace std;

// Array sizes
const int NAME_SIZE = 51, ADDR_SIZE = 51, PHONE_SIZE = 14;

// Declare a structure for the record.
struct Info
{
        char name[NAME_SIZE];
        int age;
        char address1[ADDR_SIZE];
        char address2[ADDR_SIZE];
        char phone[PHONE_SIZE];
};
```

Program 12-16 Reading Structure Variables from a file

```
int main()
{
        Info person;        // To hold info about a person
        fstream people;     // File stream object

        // Open a file for input in binary mode.
        people.open("people.dat", ios::in | ios::binary);

        // Test for errors.
        if (!people)
        {
                cout << "Error opening file. Program aborting.\n";
                return 0;
        }

        cout << "Here are the people in the file: \n\n";
        // Read the first record from the file.
        people.read(reinterpret_cast<char *>(&person), sizeof(person));

        // While not at the end of the file,
        // display the records.
```

```
        while (!people.eof())
        {
                // Display the record.
                cout << "Name: ";
                cout << person.name << endl;
                cout << "Age: ";
                cout <<  person.age << endl;
                cout << "Address line 1: ";
                cout << person.address1 << endl;
                cout << "Address line 2: ";
                cout << person.address2 << endl;
                cout << "Phone: ";
                cout << person.phone << endl;
                // Wait for the user to press the Enter key.
                cout << "\nPress the Enter key to see the next record.\n";
                cin.get();
                // Read the next record from the file.
                people.read(reinterpret_cast<char *>(&person), sizeof(person));
        }
        // Close the file.
        cout << "That's all the data in the file!\n";
        people.close();
        return 0;
}
```

Program 12-16 Reading Structure Variables from a file Concluded

Program 12-16 Output