wk5Lab\wk5lab2.cpp

```
/*******************************************************************************
Charlie Ritter
cs161 Priestley Spring 2024
Week 5 lab2  cpp

Define a Client struct that has a fixed size. Fields (data members) are id, company name, city, state, customer type.
Customer type can be anything you decide.  explain your choice in the comments inside your code.
should have a main that calls a function called Initialize
The Initialize function attempts to open a file called customers.txt. If the file is not found to open,
then it calls a helper function called WriteFile.
The WriteFile function populates a struct array of size 3 and writes the array to a binary file, then
closes the file and returns Now the "Initialize" function can be certain that it can successfully open
the file, because either it existed or it was just created. Initialize now opens the file and reads in the customer array and returns.
*******************************************************************************/
#include <cstring>
#include <fstream>
#include <iomanip>
#include <iostream>
#include <string>

using namespace std;

const string FILEF = "C:\\Users\\critter\\OneDrive - Mt. Hood Community College\\MHCC\\Spring24\\CS162\\wk5Lab\\customers.txt";
const string FILEG = "C:\\Users\\critter\\OneDrive - Mt. Hood Community College\\MHCC\\Spring24\\CS162\\wk5Lab\\customers.dat";


struct Client
{
    // populated in the write file function
    // First initial lastname
    char id[15] = "";
    // Company name, or none if retail.
    char company[15] = "";
    char city[15] = "";
    char state[4]="";
    // Customer type 'W' wholesale, 'R' retail.
    char custType[2]="";

    // Display struct data overloading ostream
    // takes ostream as argument and returns nothing
    void outP(ostream& ops){
            ops << left << setw(15) << "ID: " << setw(15) << id << endl
                << setw(15) << "Company: " << setw(15) << company << endl
                << setw(15) << "City: " << setw(15) << city << endl
                << setw(15) << "State: " << setw(15) << state << endl
                << setw(15) << "Customer Type: " << setw(15) << custType << endl;
```

```
        };
};


//prototypes
bool Initialize(string file, Client* arr);
Client WriteFile();

int main()
{
    // Declare struct array size 3
    Client customers[3];

    // Initialize file flag

    // Call Initialize function file =
    Initialize(FILEF, customers);

    // Display customer array.
    cout << "The following data is in customers.dat\n";
    for(int idx = 0; idx < 3; ++idx){
        customers[idx].outP(cout << "\n");

        }




    return 0;
}


//  function attempts to open a file called customers.txt if the file is not found to open,
//  then it calls a helper function called WriteFile.
//  takes string file and Client array as arguments and returns a bool.
bool Initialize(string txfile, Client arr[]){

    ifstream file(txfile);

    // Check if file exists
    if(!file)
    {
        cout << "File not found. \nWe will open and create a file for you." << endl;
        // Call WriteFile function to create binary file
        WriteFile();
    }


    fstream fout;
    // Open binary file
    fout.open(FILEG, ios::in | ios::binary);
    // Read file into struct array
    if (fout.is_open())
```

```
    {
        fout.read(reinterpret_cast<char*>(arr), 3 * sizeof(Client));
        fout.close();
    }



    return true;
}

// The WriteFile function populates a struct array of size 3 and writes the array to a binary file,
// then closes the file and returns.
Client WriteFile(){

    // populate array of 3 structs.
    Client clients[3] = {
                          {"jreed", "jcpenny", "Beaverton", "OR", "W"},
                          {"sjones", "none", "Gresham", "OR", "R"},
                          {"gtyler", "none", "Beaverton", "OR", "R"}
                         };

    fstream fout;

    fout.open(FILEG, ios::out | ios::binary);

    if(fout.is_open())
    {
        // write clients[3] array to binary file
        fout.write(reinterpret_cast<char*>(clients), 3 * sizeof(Client));
        fout.close();

    }

    // return Client struct
    return Client();
}


```