#include <iostream>

#include <iomanip>

#include <fstream>

#include <string>

#define N 10

using namespace std;

struct Client

{

char name[25];

char lastname[15];

char perscode[25];

char adress[50];

char ordercode1[20];

}client;

int getRequest;

void view\_client();

void filter\_client();

void sort\_client();

void enter\_client();

void delete\_client();

int menu();

int main()

{

while (1)

{

system("CLS");

int req = menu();

switch (req)

{

case 1:view\_client(); system("pause"); break;

case 2:filter\_client(); system("pause"); break;

case 3:sort\_client(); system("pause"); break;

case 4:enter\_client(); system("pause"); break;

case 5:delete\_client(); system("pause"); break;

case 11:return 0; break;

default:

cout << "---------------------------------------------------\n";

cout << "Not correct choose!\n";

cout << "You must choose numbers from 1 to 7 !\n";

cout << "---------------------------------------------------\n";

system("pause");

break;

}

}

return 0;

}

int menu()

{

int er;

cout << left;

cout << "--------------------------[Menu]--------------------------\n\n";

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

cout << setw(35) << "Operations with clients" << "|Operations with service\n";

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

cout << setw(35) << "[1] - View client\n";

cout << setw(35) << "[2] - Filter client\n";

cout << setw(35) << "[3] - Sort client\n";

cout << setw(35) << "[4] - Add client\n";

cout << setw(35) << "[5] - Delete client\n";

cout << setw(20) << "[11] - Exit\n\n";

cout << "\n";

cout << "Your choose : ";

cin >> er;

return er;

}

void view\_client()

{

ifstream file("client.txt", ios::binary | ios::in);

if (!file)

{

cerr << "File could not be opened" << endl;

exit(1);

}

cout << endl;

cout << "Press [1] - to see all clients" << endl;

cout << "Press [2] - to find client using client personal code" << endl;

cout << endl;

cout << "Your choose : ";

cin >> getRequest;

bool Found = false;

int count = 0;

switch (getRequest)

{

case 1:

cout << endl;

cout << "-----------------------------------------------------------\n";

cout << left << setw(13) << "Order code" << setw(10) << "Name" << setw(15) << "Lastname" << setw(15) << "Personal code" << setw(25) << "Adress" << "\n " << "\n";

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

cout << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

count++;

}

cout << endl;

cout << "Current number of clients in data base is : [" << count << "]" << endl;

break;

case 2:

char persCodeTemp[20];

cout << endl;

cout << "Enter client personal code: ";

cin >> persCodeTemp;

cout << endl;

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

if (strcmp(client.perscode, persCodeTemp) == 0)

{

Found = true;

cout << "-----------------------------------------------------------\n";

cout << left << setw(13) << "Order code" << setw(10) << "Name" << setw(15) << "Lastname" << setw(15) << "Personal code" << setw(25) << "Adress" << "\n " << "\n";

cout << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

cout << endl;

}

}

if (!Found)

{

cout << "Client not found.\n";

}

file.clear();

file.seekg(0, ios::beg);

break;

}

file.close();

}

void enter\_client()

{

ofstream file("client.txt", ios::binary | ios::app);

if (!file)

{

cerr << "File could not be open" << endl;

exit(1);

}

cout << endl;

cout << "Please add a new client.\n";

cout << endl;

cout << "Enter order code : ";

cin >> client.ordercode1;

cout << "Enter client name : ";

cin >> client.name;

cout << "Enter client surname: ";

cin >> client.lastname;

cout << "Enter client personal code: ";

cin >> client.perscode;

cout << "Enter client adress: ";

cin.ignore();

cin.getline(client.adress, 35);

cout << endl;

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

file << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

file.close();

}

void delete\_client()

{

char name[25] = "";

char code[25] = "";

int food;

bool Found = false;

cout << "Please enter client personal code to delete client : ";

cin >> code;

fstream file("client.txt", ios::binary | ios::in);

fstream tempfile("tempclient.txt", ios::binary | ios::out);

if (!file || !tempfile)

{

cerr << "One of the File could not be open" << endl;

exit(1);

}

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

if (strcmp(client.perscode, code) > 0 || strcmp(client.perscode, code) < 0)

{

tempfile << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

}

else if (!strcmp(client.perscode, code))

{

strcpy\_s(name, client.name);

Found = true;

}

}

cout << "Client " << name << " with personal code " << code << (Found ? " successfully deleted" : " not found") << endl;

file.close();

tempfile.close();

remove("client.txt");

rename("tempclient.txt", "client.txt");

}

void sort\_object()

{

}

void sort\_client()

{

char code[N][30], cur[30];

int i, j;

bool Found = false;

char\* temp = (char\*)malloc(25 \* sizeof(char));

cout << "Please enter client personal code to sort client : ";

cin >> code[30];

fstream file("client.txt", ios::binary | ios::in);

fstream tempfile("tempclient.txt", ios::binary | ios::out);

if (!file || !tempfile)

{

cerr << "One of the File could not be open" << endl;

exit(1);

}

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

for (i = 0; i < N - 1; i++)

for (j = i + 1; j < N; j++)

if (strcmp(code[i], code[j]) > 0)

{

strcpy\_s(cur, code[i]);

strcpy\_s(code[i], code[j]);

strcpy\_s(code[j], cur);

}

}

if (!Found)

{

cout << "Personal code not found.\n";

}

cout << "Client " << " with personal code " << code << (Found ? " successfully deleted" : " not found") << endl;

file.close();

tempfile.close();

remove("client.txt");

rename("tempclient.txt", "client.txt");

}

void filter\_client()

{

ifstream file("client.txt", ios::binary | ios::in);

if (!file)

{

cerr << "File could not be opened" << endl;

exit(1);

}

cout << endl;

cout << "Press [1] - to filter clients using order code" << endl;

cout << "Press [2] - to filter clients using client name" << endl;

cout << "Press [3] - to filter clients using client adress" << endl;

cout << endl;

cout << "Your choose : ";

cin >> getRequest;

bool Found = false;

switch (getRequest)

{

case 1:

char clientcodeTemp[25];

cout << endl;

cout << "Enter order code : ";

cin >> clientcodeTemp;

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

if (strcmp(client.ordercode1, clientcodeTemp) == 0)

{

cout << "-----------------------------------------------------------\n";

cout << left << setw(13) << "Order code" << setw(10) << "Name" << setw(15) << "Lastname" << setw(15) << "Personal code" << setw(25) << "Adress" << "\n " << "\n";

Found = true;

cout << endl;

cout << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

cout << endl;

}

}

if (!Found)

{

cout << "Order code not found.\n";

}

break;

case 2:

char clientnameTemp[25];

cout << endl;

cout << "Enter client name : ";

cin >> clientnameTemp;

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

if (strcmp(client.name, clientnameTemp) == 0)

{

cout << "-----------------------------------------------------------\n";

cout << left << setw(13) << "Order code" << setw(10) << "Name" << setw(15) << "Lastname" << setw(15) << "Personal code" << setw(25) << "Adress" << "\n " << "\n";

Found = true;

cout << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

cout << endl;

}

}

if (!Found)

{

cout << "Client name not found.\n";

}

break;

case 3:

char clientadressTemp[50];

cout << endl;

cout << "Enter client adress : ";

cin >> clientadressTemp;

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress)

{

if (strcmp(client.adress, clientadressTemp) == 0)

{

cout << "-----------------------------------------------------------\n";

cout << left << setw(13) << "Order code" << setw(10) << "Name" << setw(15) << "Lastname" << setw(15) << "Personal code" << setw(25) << "Adress" << "\n " << "\n";

Found = true;

cout << left << setw(13) << client.ordercode1 << setw(10) << client.name << setw(15) << client.lastname << setw(15) << client.perscode << setw(25) << client.adress << endl;

cout << endl;

}

}

if (!Found)

{

cout << "Client adress not found.\n";

}

file.clear();

file.seekg(0, ios::beg);

break;

}

file.close();

}

}

void sort\_client()

{

Client client;

ifstream file("client.txt", ios::binary);

ofstream tempfile("tempclient.txt", ios::binary);

if (!file)

{

cerr << "File could not be opened" << endl;

exit(1);

}

int count = 0;

while (file >> client.ordercode1 >> client.name >> client.lastname >> client.perscode >> client.adress) ++count;

Client\* cc = new Client[count];

file.clear();

file.seekg(0);

for (int i = 0; i < count; ++i)

file >> cc[i].ordercode1 >> cc[i].name >> cc[i].lastname >> cc[i].perscode >> cc[i].adress;

std::sort(cc, cc + count, [](const Client& a, const Client& b) { return strcmp(a.name, b.name) < 0; });

for (int i = 0; i < count; ++i)

tempfile << cc[i].ordercode1 << cc[i].name << cc[i].lastname << cc[i].perscode << cc[i].adress << endl;

delete[] cc;

}